Bsd - I

Başlarken...

BSD, bir çok bilgisayar kullanıcısının tanımadığı işletim sistemlerinden birisidir. Buna karşılık günlük yaşamda BSD sistemler ile sunulan veya gerçekleştirilen hizmetlerden yararlanıyoruz. BSD ile çalışan kişilerin ortak görüşleri kararlılık, güvenlik, güvenilirlik ve ölçeklenebilirlik açısından uygun bir sistem olduğudur.

BSD, sunucu ve kritik uygulamalar söz konusu olduğunda hemen hemen tüm sistem yöneticilerinin ilk tercihidir. Birçok kuruluşun bilgi işlem bölümünde, sistem odasında kendisine yer edinmiş ve sessiz, sakin çalışmaya devam etmektedir. Bugüne dek birçok sistem yöneticisi tarafından sunucu vb. hizmetler için öncelikle kullanılmış olsa da iş istasyonu çözümlerinde de sıklıkla kullanılmaktadır. Masaüstü ve dizüstü sistemlerde kullanılması görece olarak az olsa da azımsanmayacak ölçüdedir.

BSD, UNIX'vari işletim sistemleri ailesindeki diğer benzerlerine göre kullanım açısından zor olmamakla birlikte "propgandasının" az yapılması nedeni ile az bilinen ve genellikle sistem yöneticileri, geliştiriciler ve mühendisler tarafından kullanılan ve dolayısıyla da "uzman" kullanıcıların tercih ettiği bir sistem olarak kalmıştır. Son yıllarda BSD'nin masaüstü/dizüstü sistemlerde kullanımının yaygınlaşmaya başladığını görüyoruz.

BSD ne kasten karmaşıktır ne de çok basittir. BSD ile çalışmak için BSD tasarımı ile benzerlerinden nerede ayrıldığını bilmek ve gerektiğinde elinizin altındaki kaynaklara ulaşabilmek gerekiyor. Bu ve izleyen yazılarda BSD tasarımı ve işleyişi dışında BSD ile gelen uygulamalar, araçlar ve benzeri konulara yer vereceğiz. Eğer BSD kurulumuna ilişkin ulaştığınız kaynaklar kurulum konusunda sizi tereddütte bırakıyorsa bu durumda DesktopBSD veya PC-BSD kurup üzerinde çalışabilirsiniz. Yazı dizisini hazırlarken kurulum konusunun ilk konu olmasını özellikle atladık. Kurulum aşaması yerine doğrudan hazır bir sistemde çalışmanın sağlayacağı kolaylık ile kullanıcıların BSD öğrenmesini kolaylaştırmayı amaçladık. Bu nedenle kurulum konusunu yazı dizisinin ilerleyen bölümlerine bıraktık. Bu yazı dizisini, BSD adını duymuş veya eniXma'nın eski sayılarında yer verdiğimiz incelemelerden tanıyan ve daha fazlasını öğrenmek isteyen kullanıcılar için yazıyoruz. Yazı dizisini hazırlamak keyifili bir süreç oldu. Aynı keyfi sizin de BSD kullanırken yaşamanızı umut ediyoruz.

Başlarken...

BSD ile ilk tanıştığım günlerde, grafik arayüzler her ne kadar kullanılabilse de işlerin büyük kısmının kabuk ortamında yapıldığına tanık olmuştum. Grafik arayüzler ile çalışmak neyin nasıl yapılacağına dair az çok bir fikir verse de aynı işlemleri kabuk ortamından yapmak hem hızlı hem de grafik arayüzlerin sunduğundan daha fazla bilgilendirici oluyordu.

Örneğin "bad transfer error" hata mesajı gerçek hatayı belirtmemektedir. Oysa aşağıdaki işlem hatanın kaynağı hakkında daha fazla bilgi vermektedir.

```
./unopkg add /usr/home/goksin/zemberek-ooo-1.0_rc2.oxt
/libexec/ld-elf.so.1: Shared object "libuno_sal.so.3" not
found, required by "javaldx"
/libexec/ld-elf.so.1: Shared object "libuno_sal.so.3" not
found, required by "unopkg.bin"
```

İlk BSD sürümünden bu yana tasarımı aynı kalmakla birlikte kullanılan araçların tamamı grafik arayüz olmadan sistemi kullanmanıza olanak verir. Elinizdeki araçlar gereken yapılandırma ve aradığınız araçlara ulaşmanızı sağlayacak şekilde tasarlanmış ve sunulmuştur. Bu nedenle grafik arayüzler kurulu olsa da bazı durumlarda eksik bir paket veya araçtan dolayı çalışmayabilecekleri gibi yapılandırmadaki bir hata nedeniyle de erişilemez durumda olabilirler. Bu durumda ise kabuk ortamında komut satırından gereken işlemleri yapabilir ve sistemi eski durumuna döndürebilirsiniz.

Kabuk ortamında iseniz büyük bir olasılıkla siyah ekranda aşağıdaki işareti göreceksiniz:

```
$
```

Grafik arayüzlerden sonra yanıp sönen bir imleç ve $ işareti ürkütücü olacaktır. Ancak BSD sistemler size gereken araçları sunar. Yapacağınız ise gerekli olan komutları ve araçları birlikte kullanmak olacaktır. Bazı

durumlarda kullanmak istediğiniz komut veya araca ilişkin bir hata mesajı alabilirsiniz. Bu hata mesajı komutun bulunmadığını veya seçeneklerin geçersiz olduğunu belirtiyor olabilir ya da başka bir hata mesajı da olabilir.

Bu tür mesajlar imla hatasından, veya komutun/aracın ortam değişkenlerinden PATH ile tanımlanmış olan dizinlerin içinde yer almadığından ve de en kötüsü ise kurulmamış olmasından kaynaklanır.

Bir komuta/araca ilişkin, sorgulama amaçlı olarak kullanabileceğimiz birçok araç bulunur. Bunlar arasında type, whereis, which en sık kullanılanlardır.

wget BSD'de kurulumu kullanıcıya bırakılmış bir uygulamadır. Kullanmak istediğinizde eğer kurulu değilse hata döndürülecektir. İlk kontrol edeceğimiz PATH değişkeninde tanımlı dizinlerde whet olup olmadığıdır.

```
[goksin@bsd.enixma.org /usr/home/goksin]$ type wget
bash: type: wget: not found
@bsd.enixma.org /usr/home/goksin]$
```

PATH değişkeni içinde tanımlı olan dizinlerde olmayabilir. Bu durumda PATH değişkeni içinde tanımlı olan dizinler içinde nerede olduğunu sorgulamak ikinci adım olacaktır.

```
[goksin@bsd.enixma.org /usr/home/goksin]$ which wget
[goksin@bsd.enixma.org /usr/home/goksin]$
```

Ancak elde ettiğimiz sonuçta wget PATH değişkeni ile tanımlı dizinlerde bulunamamıştır. Bu durumda wget için derlenmiş ve kılavuz sayfalarının bulunduğu dizinleri sorgulamak gerekecektir.

```
[goksin@bsd.enixma.org /usr/home/goksin]$ whereis wget
wget: /usr/ports/ftp/wget
[goksin@bsd.enixma.org /usr/home/goksin]$
```

wget kurulmamış, port ağacında bulunuyor. Bu aşamadan sonra wget'i kurabiliriz. Bunun için pkg_add, make install veya portinstall kullanılabilir.

pkg_add ile kurulum:

```
[goksin@bsd.enixma.org /usr/home/goksin]$ su root
Password:
[root@bsd.enixma.org /usr/home/goksin]# pkg_add -vr ftp/wget
```

make install ile kurulum:

```
[goksin@bsd.enixma.org /usr/home/goksin]$ su root
Password:
[root@bsd.enixma.org /usr/home/goksin]# cd /usr/ports/ftp/wget
[root@bsd.enixma.org /usr/ports/ftp/wget]# make install
```

portinstall ile kurulum:

```
[goksin@bsd.enixma.org /usr/home/goksin]$ su root
Password:
[root@bsd.enixma.org /usr/home/goksin]# portinstall -vpP
ftp/wget
```

Komutlar ve araçlar ile kullanabileceğiniz seçeneklerin hepsine yer vermek bu yazı dizisinin amacı değil. Aradığınız komut, araç ve sisteme ilişkin belgelerde yer alır. Bu belgeler tüm BSD ailesinde olduğu gibi Linux ve diğer UNIX türevi sistemlerde kılavuz sayfaları -man pages- olarak sunulur. Ancak kılavuz sayfaları dışında kullandığımız her araç ve komut için yardım dosyaları de elimizin altında yer alır. Bir diğer belge ise info uygulamasıdır. Kılavuz sayfalarının ortaya çıkmasından sonra info geliştirilmiş ve kullanıcılara sunulmuştur. Kılavuz sayfaları ayı bir yazı konusu olduğu için bu yazı dizisinin kapsamı içinde yer almamaktadır.

Kılavuz sayfalarından en iyi biçimde yararlanmak için kılavuz sayfaları veri tabanında arama yapmak en uygun yöntemdir. Belli bir anahtar sözcük vererek kılavuz sayfaları veri tabanı içinde eşleşen girdileri sorgulamak yeterlidir.

```
[goksin@bsd.enixma.org /usr/home/goksin]$ apropos sysctl
blackhole(4)             - a sysctl(8) MIB for manipulating
behaviour in respect of refused TCP or UDP connection attempts
syncache(4), syncookies(4) - sysctl(8) MIBs for controlling
TCP SYN caching
sysctl(3), sysctlbyname(3), sysctlnametomib(3) - get or set
system information
sysctl(8)                - get or set kernel state
```

```
sysctl(9), SYSCTL_DECL(9), SYSCTL_INT(9), SYSCTL_LONG(9),
SYSCTL_NODE(9), SYSCTL_OPAQUE(9), SYSCTL_PROC(9),
SYSCTL_STRING(9), SYSCTL_STRUCT(9), SYSCTL_UINT(9),
SYSCTL_ULONG(9) - Static sysctl declaration functions
sysctl.conf(5)           - kernel state defaults
sysctl_add_oid(9), sysctl_move_oid(9), sysctl_remove_oid(9) -
runtime sysctl tree manipulation
sysctl_ctx_init(9), sysctl_ctx_free(9),
sysctl_ctx_entry_add(9), sysctl_ctx_entry_find(9),
sysctl_ctx_entry_del(9) - sysctl context for managing
dynamically created sysctl oids
(END)
```

apropos'un döndürdüğü sonuçlar syscntl barındıran kılavuz sayfalarıdır. Parantez içinde yer alan sayılar ise kılavuz sayfasının veya sayfalarının bulunduğu bölümü göstermektedir. Kılavuz sayfaları aşağıdaki listede gösterildiği gibi sınıflandırılmıştır ve BSD'de kılavuz sayfalarının yapısı aşağıdaki gibidir:

UNIX Kılavuz İndisi
Bölüm1 Kullanıcı Komutları
Bölüm2 Sistem Çağrıları
Bölüm3 Alt Rutinler
Bölüm4 Aygıt Sürücüleri
Bölüm5 Dosya Biçemleri
Bölüm6 Oyunlar
Bölüm7 Çeşitli
Bölüm8 Sistem Yöneticisi
Bölüm9 Çekirdek
Bölüml Yerel Belge
Bölümn Yeni

Kılavuz sayfaları veri tabanında arama yapmak yukarıdaki örnekte olduğu gibi çok sayıda sonuç döndürebilir. Bunun yerine eğer komutun/aracın adını biliyorsanız kılavuz sayfası ile birlikte kısa bir tanımını da sorgulamak olanaklıdır. Bunun için whatis kullanmak yeterli olur. whatis ise aprorpos'tan farklı olarak söz konusu anahtar sözcüğü kılavuz sayfaları veritabanındaki isim bölümünde arayarak size sonuçları döndürür. Bazen apropos'a göre daha kesin sonuçlar verir. Bunu whatis cat ve apropos cat ile deneyebilirsiniz. Bir komutun veya aracın kılavuz sayfasına erişmek için doğrudan man <isim> yazmak yeterlidir.

Yukarıda söz ettiğimiz info ise BSD dosya sistemleri, komutları veya araçları hakkında bilgi almak için kullandığımız diğer araçtır. Bazen geliştiriciler kılavuz sayfalarını yeterli görmeyerek daha ayrıntılı bilgi vermek için info ile erişilen sayfalar hazırlar. Bu sayfalar yine kılavuz sayfaları gibi aynı sistemetik içinde info veritabanında kayıtlı olarak bulunur. info sayfaları kullanım olarak kılavuz sayfalarına benzer.

```
info <isim>
```

Yukarıdaki komut <isim> için hazırlanmış olan info sayfasını döndürecektir. Kılavuz sayfalarına göre info sayfaları içinde dolaşmak görece olarak daha kolaydır. Yukarıda anlatıldığı gibi doğrudan belli bir komut veya araç için hazırlanan info sayfasından başlayarak doğrudan tüm info sayfalarına erişebilirsiniz.

info sayfaları içinde dolaşmak için kullanabileceğiniz tuşlar aşağıdadır:

? info ekranında kullanabileceğiniz komutların listesini verir.
L Bir önce görüntülediğiniz info sayfasına dönebilirsiniz.
n, p, u Görüntülediğiniz sayfadan önceki (p), sonraki (n) veya o bölümün başına (u) dönersiniz.
Tab Görüntülediğiniz sayfada yer alan bağlantıyı izlemek için üzerine gelip basın.
Enter İmlecin üzerindeki bağlantıya gitmek için basın.
R Bir çapraz referansı izlemek için basın.
Q İnfo'yu sona erdirir ve çıkış yaparsınız.

FreeBSD, NetBSD, DragonflyBSD ve/veya OpenBSD ile bunlar üzerinde geliştirilen PC-BSD/DesktopBSD kullanırken eğer önceden Linux kullandıysanız bu yazı dizisi içinde yer alacak bazı konular size tanıdık gelecektir. Öte yandan bazı açılardan BSD'nin, Linux sistemlerden ve hatta dergi okuyucuları arasında Windows ve Mac OS kullanaıcıları olduğunu da dikkate alırsak adı geçen sistemlere göre çok daha farklı bir yapıya ve işleyişe sahip olduğunu görebiliriz. BSD'nin işleyişini kavramak ve diğer işletim sistemleri ile karşılaştırıldığında farklı olan yanlarının doğru olarak anlaşılması gereklidir. Bu aşama BSD kullanabilmek ve sistemdan tam olarak yararlanabilmek için öncelikle gerçekleştirilmesi gereken bir aşamadır.

BSD, işleyişini kavramak için öcelikle dosya sistemine göz atmak gereklidir. Burada dosya sistemi ile sözünü ettiğimiz dosyaların dizinler olarak

düzenlenme şeklidir. Her işletim sisteminde dosyalar belirli bir düzen içinde yer alır ve dosya sistemi ile işletim sistemi tarafından bir dizi arayüz üzerinden erişilir, okunur, yazılır ve çalıştırılır. Günümüzün modern işletim sistemlerinin çekirdekleri birden fazla dosya sistemini destekleyebilmektedir. Bu özellikleri nedeni ile aynı disk üzerinde birden fazla dosya sistemi kullanılabilir ve eş zamanlı olarak çekirdek tarafından erişilebilir.

Dosya sistemleri bazen kısaltılmış ve anlaşılması isimler ile anılırlar. örneğin Ext2/Ext3, ReiserFS, NTFS, HFS+, UFS, UFS+S veya FFS gibi. Bu isimlere bakıldığında dosya sisteminin hangi işletim sistemine ait olduğunu anlamak bu sorunun yöneltildiği kişinin bu dosya sistemlerini bilmesi durumunda kolaydır. Dosya sistemleri burada saydıklarımız ile sınırlı değildir. Farklı işletim sistemleri farklı dosya sistemleri kullanırlar ve bu dosya sistemlerinin diğerlerine göre üstün ve zayıf yönleri bulunmaktadır.

BSD, Fast FileSytem -FFS- kullanmaktadır. Eğer elinizin altında bir BSD sistem kurulu ise, ya kurulum aşamasında veya sonradan baktığınızda FFS değil UFS olarak dosya sisteminizi görebilirsiniz. UFS, aslında UNIX gelişim sürecinin ilk günlerinde -ticari UNIX sistemlerin ortaya çıkmasından önceki dönem- tüm UNIX sistemler tarafından kullanılmıştır. Sonraki süreçte FFS geliştirilmiş ve UFS'nin yerini almıştır. Ancak BSD sistemlerin hepsinde dosya sistemi UFS olarak geçer. FFS, UFS yerini almakla birlikte UFS adı ile kullanılagelmiştir. Bugün için FreeBSD, OpenBSD ve NetBSD sistemlerin hepsinde dosya sistemi UFS olarak geçse de dosya sistemi FFS'tir. Buna Mac OS X'in üzerinde geliştirildiği Darwin'i de dahil edilebilir. BSD sistemde kullanabileceğiniz tüm araçlar ve komutlar dosya sistemi olarak size UFS yanıtını döndürecektir.

Kullandığımız dosya sistemi belirleyici olmakla birlikte dosya sistemi ile ilgili bilgiler dışında kullanacağımız araçları ve dizin hiyerarşisini bilmek yerinde olacaktır. İlk olarak dizin hiyerarşisi ile başlayalım. Linux kullandıysanız BSD dizin yapısı size tanıdık gelmekle birlikte bazı noktalarda farklar göreceksiniz.

```
[goksin@bsd.enixma.org /usr/home/goksin]$ ls -Fl /
total 45
-r--r--r--   1 root  wheel  6192  6 Oca  2008 COPYRIGHT
drwxr-xr-x   2 root  wheel  1024  6 Oca  2008 bin/
drwxr-xr-x   7 root  wheel   512 11 Kas 09:53 boot/
drwxr-xr-x   2 root  wheel   512  3 Eyl  2007 cdrom/
lrwxr-xr-x   1 root  wheel    10  6 Oca  2008 compat@ ->
usr/compat
dr-xr-xr-x   4 root  wheel   512  1 Oca  1970 dev/
-rw-------   1 root  wheel  4096 24 Kas 12:28 entropy
drwxr-xr-x  18 root  wheel  2048 22 Kas 22:49 etc/
lrwxr-xr-x   1 root  wheel     8  6 Oca  2008 home@ ->
usr/home
drwxr-xr-x   3 root  wheel  1024 11 Kas 09:56 lib/
drwxr-xr-x   2 root  wheel   512  6 Oca  2008 libexec/
drwxrwxr-t   2 root  wheel   512 24 Kas 11:34 media/
drwxr-xr-x   2 1003  wheel   512 28 Ara  2007 mnt/
dr-xr-xr-x   1 root  wheel     0 24 Kas 18:29 proc/
drwxr-xr-x   2 root  wheel  2560 11 Kas 09:56 rescue/
drwxr-xr-x   6 root  wheel   512 24 Kas 01:08 root/
drwxr-xr-x   2 root  wheel  2560  6 Oca  2008 sbin/
lrwxr-xr-x   1 root  wheel    11  6 Oca  2008 sys@ ->
usr/src/sys
drwxrwxrwt   9 root  wheel   512 24 Kas 16:42 tmp/
drwxr-xr-x  20 root  wheel   512  3 Kas 21:43 usr/
drwxr-xr-x  26 root  wheel   512 24 Kas 15:40 var/
[goksin@bsd.enixma.org /usr/home/goksin]$
```

Windows veya eski MAC OS sürümlerini kullandıysanız yukarıdaki tabloda gördüğünüz dizin isimleri anlaşılmaz gelecektir. UNIX geliştirme sürecinin ilk günlerinde neden bu isimlerin kullanıldığı bilinmekle birlikte artık bugün bu sorunun yanıtını arayan da kalmamış olmalı ki bu isimler ile devam ediyoruz. Şaka bir yana BSD sistemlerde dizin yapısı sistemin bakımını ve yönetilmesini son derece kolaylaştırmaktadır.

Yukarıdaki çıktıda gördüğünüz dizin isimleri kurulum şekline göre değişkenlik gösterebilir. Sonunda / olan ve drw... ile başlayan satırlar dizinleri gösterirken, lrw... ile başlayanlar ve sonunda @ veya -> olanlar ise sembolik bağları göstermektedir. Aşağıda, yukarıda gördüğünüz dizinler içinden çeşitli kurulum senaryolarında görebileceğiniz dizinler kısaca tanıtılmıştır.

bin/

Diğer uygulamalara veya kütüphanelere gereksinim duymayan statik olarak derlenmiş, ikili dosyaların yer aldığı dizindir. Sisteminizde sadece /

dizinine erişebildiğiniz ve diğer disk bölümlerine erişemediğiniz durumlarda dahi bu dizindeki dosyaları çalıştırabilirsiniz. Bu dizin altındaki uygulamalar sistemin bütünü üzerinde etkili olmayan kullanıcıların gereksinim duyacağı uygulamalardır. Sistem yönetimine ilişkin uygulamalar /sbin -system binary- altında yer alır. Bunlar ise sistemin çalışması ve yönetimine dair uygulamalardır.

boot/

Bu dizinde çekirdek -kernel- yer alır. Çekirdek, ağ servislerini, donanımı ve sistem kaynaklarını kontrol eder. Ayrıca açılış sırasında gereksinim duyulan yapılandırma ve diğer dosyalar da bu dizinde yer alır.

compat@

BSD aynı zamanda farklı işletim sistemlerine ait uygulamaları çalıştırmanıza olanak verir. Örneğin: Linux gibi. Eğer Linux uyumluluğunu kurulumda seçmediyseniz bu dizin boş olacaktır.

dev/

UNIX sistemde her şey bir dosya olduğu için donanım da bir dosyadır. Bu dizin bu dosyaların yer aldığı dizindir. Çekirdek tarafından desteklenen donanımlara erişecek olan uygulamalar buradaki dosyaları kullanır. Aynı zamanda /dev, /devfs için bağlanma noktası olarak da görev yapar.

etc/

Eskiden /etc dizini diğer dizinler altında olup olmayacağına karar verilemeyen dosyaların yer aldığı dizin iken, bugün yapılandırma dosyalarının yer aldığı dizindir. Bunlara kullanıcı şifreleri ile açılışta çalıştırılan programlarda da dahildir.

home@

Genel olarak kurulum aşamasında eğer çok sayıda kullanıcıya sahip olacak bir sistem tasarlıyorsanız /home, /usr/home dizinlerini gösteren bir sembolik bağ olarak kalacaktır. Böylelikle / olarak etiketlenen disk bölümü yeterince küçük olabilir. /usr dizini altında yer alan /home ise kullanıcıların dosyalarını barındıracakları dizin olarak hizmet edebilir. Kullanıcı dizinleri /usr disk bölümünde yer alabilir. BSD için /usr ve /var diğer disk bölümlerine göre daha büyük olacaktır. Öte yandan kullanıcı dizinlerinin daha büyük bir alana gereksinim duyacağını düşünüyorsanız bu durumda ayrı bir disk bölümünü /home olarak atayabilirsiniz.

mnt/

Sonradan bir disk veya optik sürücüyü bağlamak için kullanabileceğiniz dizindir. Bu gün bu işler için /media kullanılsa da /mnt bakım ve onarım görevlerinde daha yararlı olmaktadır.

proc/

Süreçlere ait olan dosyaların yer aldığı dizindir. Süreçlere ait olan procfs dosya sistemi bu dizin altına bağlanır ve süreç tablosu yer alır. Yedeklenmesine gerek duyulmayan bir sistemdir.

rescue/

Acil bir durumda, kurtarma işlemlerinde veya kurulumlar sırasında minimalistik bir BSD ortamı sunmak için gereken tüm araçların yer aldığı dizindir.

root/

Sistem yöneticisi olan root kullanıcısının ev dizinidir. Güvenlik nedenleri ile /home altında yer almaz. Bir dizin acil durumda erişilebilir olması gerektiği için / altında yer alır.

sbin/

Sistemin işleyişi için kullanılan, statik olarak derlenmiş -/bin altındakiler gibi- uygulamalardır. /bin içinde yer alanlardan farklı olarak kullandığınızda sistemin işleyişini kontrol edip değiştirmenize olanak veren uygulamalardır.

sys@

Eğer kurulum sırasında sisteme ait kaynak kodları kurduysanız kaynak kodların yer aldığı dizine sembolik bağdır.

tmp/

Geçici dosyaların yer aldığı dizindir. Tüm kullanıcılar bu dizine yazabilir. Eğer /etc/rc.conf dosyasına clear_tmp_enable="YES" satırını eklediyseniz sistemi her başlattığınızda /tmp dizininin içeriği silinecektir.

usr/

Uzun adı ile UNIX System Resources dizini yani UNIX Sistam Kaynakları dizinidir. Bu dizin altında dinamik olarak derlenmiş uygulamalar, kullanıcı dizinleri ve sonradan kurulmuş olan tüm uygulamalar yer alır. Uzun sözün kısası, ömrünüzün geri kalanını geçireceğiniz yer burasıdır. Gelecek bölümlerde bu dizin altıda yer alan dizinler ve programlar ile fazlasıyla çalışacağız. /usr alt dizinleri, kullanım amacına göre hazırlanmış dizinleri kapsar. BSD kullanıcıları ve sistem yöneticileri için en önemli dizin olan /usr/local burada yer alır.

var/

Çalıştırıldığında veya diğer uygulamalar tarafından değiştirilen veya sürekli değiştirilen dosyaların yer aldı dizindir. Bunlar arasında runtime dosyaları, log dosyaları, spool dosyaları vb. burada yer alır.

Linux kullanıcıları için BSD dizin hiyerarşisi ilk bakışta aynı gibi görünse de aslında önemli farklar bulunmaktadır. BSD'de dizin hiyerarşisi sıkı biçimde kontrol edilir. Temel kural, sistem yöneticisi -root- tarafından kurulan her uygulama /usr/local altında olmalıdır. Diğer işletim sistemlerinde bu kural esenetilebilir veya dikkate alınmayabilir. BSD sözkonusu olduğunda ise dizin hiyararşisi katı biçimde kontrol edilir. BSD'ye aktarılan uygulamalar da dahil olmak üzere bu kural uygulanır. Buna karşın, bazen bir uygulama kütüphanelerini /var/lib altına, yapılandırma dosyalarını /etc vb. kopyalabilir. BSD yamaları -ileride göreceğimiz üzere- kurulum programlarını değiştirip, dosyaların /usr/local/lib ve /usr/local/etc dizinlerine kopyalanmasını sağlar. Aslında kuracağınız her program, /usr/local/etc altına kendi yapılandırma dosyalarını kopyalayacaktır. Eğer açılışta çalıştırılması gereken bir kabuk programı da kuruluyorsa o da /usr/local/etc/rc.d altına kopyalanır. Bu dizinde yer alan tüm dosyalar asıl sistem başlatıldığında çalıştırılacak olan /etc/rc.d dizinindeki dosyalar çalıştırıldıktan sonra çalıştırılır.

Sıkı bir şekilde kontrol edilen dizin hiyerarşisinin yararları sistem üzerinde kesin bir kontrole izin vermesidir. Bunun yanında sistemin bakım ve yönetiminde önemli kolaylıklar sağlayacaktır.

Elinizdeki donanımı değiştirmek durumunda olduğunuz bir senaryo düşünelim. Sisteminizin /usr dizinini tuttuğunuz diskinizi yeni ve daha büyük bir disk ile değiştirmek istediğinizde kuramsal olarak düşünürsek, tüm /usr/local dizinini bir tar.bz2 dosyası yapabilir, sonra yeni diskinizi takıp, bölümleyip etkiketledikten sonra tar.bz2 dosyasını açıp eskiden olduğu gibi devam edebilirsiniz. Bunu yapmak ise söylemekten daha zordur. Bu işlemi gerçekleştirirken bazı öngörülmeyen sorunlar ile karşılaşabilirsiniz. Eski sisteminizde bazı ince ayarlar yapmış olmanız da olasıdır ve yeni diske dosyaları kopyalamak bu ince ayarların aynen geçerli olacağı anlamına gelmeyecektir. İdeal bir dünmyada BSD sistemler bunu yapar, inanın bana!

BSD'de uygulanan bu katı kontrol mekanizması bazı sıkıntılar da yaratabilir. Linux veya Solaris için hazırlanmış bir programı BSD'ye aktarmak durumunda iseniz, programın kullanacağı dizinleri yapılandırmak düşündüğünüzden daha zor olması olasıdır. Bu durum en çok Python ile Linux için yazılmış uygulamalarda ortaya çıkmaktadır. Linux dağıtımlarında Python yorumlayıcısı /usr/bin/python altında yer aldığı için BSD'de bu uygulamalar kesinlikle çalışmayacaktır. Zira Python, BSD için sistemin temel bir bileşeni değildir ve kurulması durumunda da /usr/local/bin/python altında yer alacaktır. Bu sıkıntıyı yorumlayıcının bulunduğu dizini tanımlayan satırları değiştirip çözebilirsiniz. Bu düzenleme işlemi bir veya iki program için sıkıntı yaratmayabilir ama on, yüz veya daha fazla uygulama söz konusu olduğunda ise işler sarpa saracaktır.

Eğer farklı platformlarda çalışacak bir program yazıyorsanız, yorumlayıcının bulunduğu dizini tanımlarken yapacağınız bir hata BSD dizin hiyerarşisi kurallarını çiğnemek olacaktır. Sonuç olarak programınız çalışmayacaktır. Bunun önüne geçmek için /usr/bin/python bir sembolik bağ ile /usr/local/bin/python yapmak sorunu çözeceği gibi yorumlayıcı satırını #!/usr/bin/env python olarak yazmak da sorunu çözecektir.

Burada bir noktaya da dikkat çekmek gerekiyor. Dizin hiyerarşisinin nasıl olması gerektiğine dair hazırlanmış belgeler bulunmaktadır. Bu çalışmalar genelde Linux sistemleri dikkate alarak hazırlandığı için BSD de dikkate alınmayan bazı dizinleri de kapsamaktadır. Örneğin sunucu dosyalarının /srv altında olması gibi. Bu kaynakları kullanıyorsanız dikkat etmeniz

gerekecektir.

Disk Alanı Kullanımını İzlemek

BSD, dosya sistemi tek bir dizin olarak görünür. Başlangıcı / olan bu dosya sisteminde her bir disk bölümü kendisine ait olan tanımlanmış dizin altına bağlanır. Disk veya disklerinizdeki alan kullanımını izlemek ve gereken düzenlemeleri yapabilmek için her bir diskin bağlandığı dizin altında bağlanmış olan dosya sistemine göz atmak gerekir.

UNIX'de bir dizin altına bağlanan dosya sistemi fikri garip gelebilir. Windows sistemlerde her bir disk C,D vb. bir harfle gösterilir ve kendine özgü dosya sistemine sahiptir. UNIX'de ise sadece bir tane dizin vardır ve bu dizin altında farklı dosya sistemleri yer alırken tüm dizinler tek bir dizin altında yer alır. Konuyu daha iyi açıklayacak bir örnek vermek gerekirse Windows sistemler için dosya sistemlerini terzi, manav, bakkal, kasap, tuhafiyeci ve manifaturacı vs. gibi ufak dükkanların yan yana veya dağınık olarak sıralandığı bir semt gibi düşünürsek, UNIX sistemler için tüm dükkanların tek bir çatı altında yer aldığı büyük bir alışveriş merkezi olduğunu düşünebiliriz.

Bu çeşit bir yapılanmanın yararı, sisteme gerektiğinde yeni bir veya daha fazla diski eklemenin son derece kolay olmasıdır. Dizin hiyerarşisinde bir bölümdeki disk alanını arttırmak için o dizin altında bağlanan disk sayısını arttırmak yeterlidir. Bu konuya ileride ayrıntılı olarak değineceğim. Öte yandan sakıncası ise bunun gibi bir dosya sisteminde bir dizinin tüm içeriğinin eski bir diskten yeni bir diske aktarılmasının MAC OS veya Windows'a göre daha zor olmasıdır. Bu yapılanma sunucu sistemler için daha uygun olmaktadır. Zira sunucularda terfiler ve benzeri gereksinmeler daha uzun bir döneme ve daha seyrek aralıklar ile gerçekleşirken bir masaüstü sistem için donanım veriye göre daha kısa zamanda "eskimektedir".

df: Diskinizde Ne Kadar Boş Alan Var?

Sistemde yer alan disklerin ne kadar dolu/boş olduğunu belirleyebilmenin en kolay yolu df -disk free- aracını kullanmaktır. BSD kullanıyorsanız veya birden fazla BSD sistemden sorumlu iseniz df günlük olarak bazen de gün aşırı kullanmak durumunda olduğunuz bir araç olacaktır. df ile kullandığınız her bir disk ve disk üzerindeki dosya sistemi ile diğer gerekli olan bilgiler edinilir. Kendi sisteminizde de aşağıdakine benzer bir sonuç elde edeceksiniz.

```
[goksin@bsd.enixma.org /usr/home/goksin]$ df
Filesystem  1K-blocks     Used    Avail Capacity  Mounted on
/dev/ad4s1a    253678    73536   159848    32%    /
devfs               1        1        0   100%    /dev
/dev/ad4s1d    507630   249252   217768    53%    /var
/dev/ad4s1e  74416072 57198284 11264504    84%    /usr
/usr/tmp     74416072 57198284 11264504    84%    /tmp
procfs              4        4        0   100%    /proc
linprocfs           4        4        0   100%
   /usr/compat/linux/proc
[goksin@bsd.enixma.org /usr/home/goksin]$ df -h
Filesystem     Size    Used   Avail Capacity  Mounted on
/dev/ad4s1a    248M     72M    156M    32%    /
devfs          1.0K    1.0K      0B   100%    /dev
/dev/ad4s1d    496M    243M    213M    53%    /var
/dev/ad4s1e     71G     55G     11G    84%    /usr
/usr/tmp        71G     55G     11G    84%    /tmp
procfs         4.0K    4.0K      0B   100%    /proc
linprocfs      4.0K    4.0K      0B   100%
   /usr/compat/linux/proc
[goksin@bsd.enixma.org /usr/home/goksin]$
```

Her bir disk bölümü veya BSD jargonuna göre dilim-slice- tanımlı olduğu dizin altında bağlı ise görüntülenecektir. Çıktıda BSD sistemde /, /var ve /usr disk bölümleri görülmektedir. / bölümü 156MB, var 213MB ve /usr ise 71GB olarak disk üzerinde bölümlenmiştir. Bu kurulum BSD disk bölümleme uygulaması tarafından tüm diskin BSD'ye ayrılması durumunda varsayılan disk bölümleme şeklidir. /usr dışında kalan dizinlere yapılandırma gereği çok az dosya kopyalanacak veya oluşturulacaktır. /var disk bölümünün /usr'a göre daha küçük olması nedeni ile kurulum sırasında daha küçük disk bölümleri kullanılması hatalı bir yaklaşımdır. Log dosyaları /var altında tutulduğu ve bu dosyaların "hızlı büyümek gibi bir özellikleri olduğu düşünüldüğünde yeterince büyük olmasına dikkat edilmesi gerekir. Bazı sistem yöneticileri bu durumu dikkate alarak /var disk bölümünü bir sembolik bağ olarak tanımlayıp /usr/var dizinine bağlamayı tercih etmektedir.

df'nin döndürdüğü sonuçlar, Linux veya Solaris kullandıysanız, beklediği-

nizden daha farklı olabilir. df, hesapladığı değerler farklı uygulamaların ve süreçlerin diske yazması ve okuması sırasında oluşabilecek değişimler sonucunda disk üzerindeki boş alanın kullanımını birebir yansıtmaya-caktır. Bu nedenle kesin bir değer olarak düşünülmemelidir.

du - disk usage- Disk Kullanımı

df kesin olmayan ama yaklaşık değerler veren bir araç olduğu için daha kesin bilgi verecek bir araca gereksinim vardır. Bu iş için du biçilmiş kaftandır. Özellikle birçok kullanıcının erişebildiği bir dosya sunucusu vb. bir servis çalıştırıyorsanız kullanıcılar bazı büyük dosyaları paylaşılan dizinlere yüklemekte terreddüt etmeyecektir

```
[goksin@bsd.enixma.org /usr/home/goksin]$ du -h -d 1
/home/goksin/Belgeler/
390K    /home/goksin/Belgeler/AMD
7,3M    /home/goksin/Belgeler/Pdf
 28M    /home/goksin/Belgeler/Post_Script
 64M    /home/goksin/Belgeler/Resimler
 85M    /home/goksin/Belgeler/Tux-Magazine
2,3G    /home/goksin/Belgeler/Bilgisayar
 88M    /home/goksin/Belgeler/Dergi_yazilarim
1,6G    /home/goksin/Belgeler/Dersler
4,1G    /home/goksin/Belgeler/
[goksin@bsd.enixma.org /usr/home/goksin]$
```

du aracı, belirttiğiniz dizinden başlayarak alt dizinleri de kapsayacak şekilde -d 1 ile 1 adet alt dizin ile sınırlandırdık- disk kullanım bilgilerini görüntülemektedir. -h ise dosya boyutlarını KB, MB ve GB cinsinden daha okunabilir şekilde döndürmektedir. Eğer tüm dizine ait tek bir sonuç elde etmek istesek -s seçeneği ile çalıştırmak yeterli olur. Bu durumda sadece tek bir satırlık çıktı döndürülecektir.

```
goksin@bsd.enixma.org /usr/home/goksin]$ du -h -s
/home/goksin/Belgeler/
4,1G    /home/goksin/Belgeler/
[goksin@bsd.enixma.org /usr/home/goksin]$
```

du'nun diğer seçenekleri sembolik bağları vs. nasıl değerlendirdiğini tanımlamaktadır. Disk alanı kullanımı konusunda bir sınırlama getirmek durumunda, örneğin aynı bilgisayarı birden fazla kişi paylaşıyorsanız bu durumda disk alanı kullanımını sınırlamak uygun olacaktır. Bu konuyu ileride ele alacağız.

BSD'de Dosya Sistemlerini Bağlamak ve Çözmek

UNIX türevi sistemlerin en güzel özelliklerinden birisi tek bir disk üzerinde kurduğunuz sisteminizin artan gerekesinmeleriniz sonucunda yeni bir disk ekleyip günlük çalışmanıza aynı şekilde olduğu gibi devam etmenize olanak vermesidir. Diskinizi anakartınıza taktıktan ve bölümleyip etiket-ledikten sonra tüm diskin üzerinde bir veya daha fazla dosya sistemi oluşturup uygun gördüğünüz dizin altına bağlayarak kullanabilirsiniz.

Mount Aracı

Aynı bilgisayarı birden çok kullanıcı ile paylaşmak durumunda olan birisi iseniz, sisteme eklediğiniz her yeni kullanıcı ile /usr dizininin giderek şiştiğini hatta, diskinizin de dolmaya başladığını görebilirsiniz. Bunun nedeni /usr/home dizininin giderek büyümesinden başka bir şey değildir. Bu durumda /usr/home'u barındıracak yeni bir disk eklemek uygun bir çözüm olacaktır. Bu işlemi yaparken sistemi açılış menüsündeki 4. seçenek olan 'single user mode' ile başlatmanız ve işlemlerinizi grafik arayüz kullanmadan sadece kabuk üzerinden yapmanız gerekecektir.

Yeni bir disk aldığınızı ve bu diske sisteminizi taşıyacacağınız varsayalım. Yeni diskinizde /dev/ad1s1e, /dev/ad1s1f ve yeni ev dizini olarak kullan-mak üzere /dev/ad1s1g disk bölümleri oluşturdunuz. Eski sistemdeki /home dizininizi yeni diskteki /dev/ads1s1g bölümüne taşıma işlemine bağlayalım.

Eğer var olan sistedeki /home dizini, /usr/home'a işaret eden bir sembolik bağ ise yeni disk bölümünü bu bölüme veya uygun gördüğünüz bir dizin altına bağlayabilirsiniz. Eğer tersi bir durum söz konusu ise /home sembolik bir bağ değilse bu dizin altına yeni diski başladığınızda eski /home dizinin içeriğine erişmeniz olanaklı olmayacaktır. Bu durumda

```
# mv /home /home.yedek
```

ile başka bir isim vermek en uygun çözüm olacaktır. Kişisel tercihim bu işlemler için /mnt dizinini kullanmaktır. Yeni disk bölümünü /mnt altına bağlayıp eski dizinlerdeki dosyaları buraya taşımak bana göre en kolay yöntem olmaktadır. Öte yandan aynı düğüm -node- yani bağlanma

noktası olarak tanımladığınız dizin altındaki dosyalara aynı zamanda erişirken yeni bağladığınız diske de erişmek istiyorsanız bu durumda unionfs dosya sistemi kullanılmalıdır. Ancak unionfs dosya sistemi konusunda bilgim az olduğu için burada sadece adını vermekle yetiniyorum. /dev/ad1s1g /home altına bağlamak için aşağıdaki işlemi yapıyoruz.

```
# mount /dev/ad1s1g /home
```

Eğer verdiğiniz komut hata mesajı döndürmeden sonuçlandıysa df ile kontrol ettiğinizde aşağıdakine benzer bir çıktı görebilirsiniz.

```
# mount /dev/ad1s1g /mnt
Filesystem  1K-blocks     Used    Avail Capacity  Mounted on
/dev/ad0s1a    128990    74314    44358    63%    /
/dev/ad0s1f    257998       44   237316     0%    /tmp
/dev/ad0s1g   8027686  1990866  5394606    27%    /usr
/dev/ad0s1e    257998    30586   227694    14%    /var
procfs              4        4        0   100%    /proc
/dev/ad1s1g  39245453        0 39245453     0%    /home
#
```

Sonuca bakılırsa, disk bölümünü bağladık ve artık /usr/hone altındaki dosyaları taşımaya başlayabiliriz. Burada yukarıda söz ettiğim gibi bu işlem sorunsuz gerçekleşmeyebilir. Bağlama işlemi sırasında "Incorrect super block" veya "Invalid argument" gibi bir hata mesajı alabilirsiniz. mount ile bağlamak istediğiniz dosya sistemi veya disk bölümünün hatalı tanımlanmış olması veya dosya sisteminde bir hata olması sonucu ortaya çıkabilir. Bu durum özellikle de BSD'de kullanılan isimlendirmeye yabancı olmanızdan kaynaklanır. Bir dosya sistemi /dev/ad1s1e ve /dev/ad1s1 ve yukarıda verdiğimiz örneklerde olduğu gibi dilim modu - slice mode- veya adanmış mod -dedicated mode- olarak disk bölümlemesi yapmanıza bağlı olarak değişkenlik gösterir.

Eğer bağlarken bir hata mesajı aldıysanız ve isimleri hatalı veya eksik yazmadıysanız bu durumda mount -f kullanmak akla ilk gelen durumdur. -f seçeneği ile zorlayarak bağlama işlemi yaparsınız. Ancak zorlamak yerine sorunun kaynağını bulmak ve çözmek daha doğru olacaktır. Örneğin disk üzerinde dosya sistemi oluşturmuş olabilir ama formatlamamış olabilirsiniz. Sistem doğru biçimde kapatılmamış olabilir, tanınmayan bir dosya sistemini kullanan bir diski bağlamaya çalışıyor olabilirsiniz. Eğer doğru biçimde kapanmamış bir sisteme ait diski bağlamaya çalışıyorsanız bu durumda fsck ile dosya sistemini kontrol etmek gerekir. Bu işlemden sonra dosya sistemi bağlanabilir.

Bağlama işlemi sırasında sadece -f seçeneği elimizde bulunmamaktadır. mount kılavuz sayfasında dosya sistemlerinin bağlanması işlemi sırasında kullanabileceğimiz bir çok seçenek olduğu görülür. Bunun için kılavuz sayfalarına man mount ile bakmanız uygun olacaktır.

umount Aracı

Dosya sistemine kopyalama işlemi yaptıktan sonra dosya sistemini çözmek gerekir. Bunun için de dosya sisteminin bağlı olduğu düğüm ayrılır. Yukarıdaki örnekte /home altına bağladığımız diskin ilgili bölümünü aşağıdaki komutla çözüyoruz

```
# umount /home
```

Aynı işlemi

```
# umount /dev/ad1s1g
```

ile de yapabilirsiniz. umount -a aynı işlemi gerçekleştirir. -a seçeneği ile tüm bağlı dosya sistemleri ayrılır. Sadece root dosya sistemi bağlı kalır.

Bir dosya sistemini ayırmak, bağlamaya göre daha kolay bir işlemdir. Ancak dikkat edilmesi gereken nokta, bağlanmış olan dosya sisteminin çekirdek, bir daemon veya bir kullanıcı süreci tarafından kullanılmıyor olması gerektiğidir. Bir diğer deyişle /var ve /usr disk bölümlerini ancak tek kullanıcı modunda ayırabiliriz. /home dizini için de aynı durum söz konusudur. Kullanıcıların sistemde olmadıklarında /home bölümü ayrılabilir. Bu durum özellikle kullanıcıların ilk defa mount ve umount kullanmaya çalıştıklarında "Device busy" hata mesajının kaynağıdır. Sisteme kendi kullanıcı hesabınız ile giriş yaptıysanız ve bulunduğunuz dizinin üzerinde yer aldığı dosya sistemini ayırmak istediyseniz dosya sistemi meşgul yanıtını alırsınız. Ayırmanız gerekiyorsa / bölümüne geçmeniz ve işlemi yeniden yapmanız gerekir.

Eğer dosya sistemini ayıramıyorsanız mount ile olduğu gibi -f kullanabilirsiniz. Bu seçeneği kullanmak özellikle sistemin üzerinde kurulu olduğu dosya sistemine zarar verebileceği için istenemeyen bir durumdur. Olabildiğinde kaçınmak gerekir.

Diğer İşletim Sistemlerine Ait Dosya Sistemlerini Bağlamak ve Ayırmak

BSD sisteminiz tek işletim sistemi olarak kurulu olmayabilir. BSD yanında diğer işletim sistemleri de kurulu ise bu işletim sistemlerine ait olan dosya sistemlerini de mount/umount ile ayırabilir ve çözebilirsiniz. Aşağıdaki listede BSD'nin GENERIC, yani varsayılan çekirdeği tarafından desteklenen dosya sistemlerinin bir listesi verilmiştir. Dosya sistemlerinin hem İngilizce hem de Türkçe adlarını vermeyi uygun bulduk. Türkçe kaynaklarda bu dosya sistemlerine ait İngilizce isimler ağırlıklı olarak kullanıldığı ve önerilen Türkçe isimler aynı dosya sistemine ait olsa da aralarında farklılık gösterdiği için bazılarına yer verilmemiştir.

Dosya Sistemi	Name
FFS	Berkeley Fast Filesystem
MFS	Memory Filesystem
NFS	Network Filesystem- Ağ Dosya sistemi
MSDOSFS	MS-DOS Filesystem - MS-DOS dosya sistemi
CD9660	ISO 9660 (CD-ROM) Filesystem - CD-ROM dosya sistemi
PROCFS	Process Filesystem -Süreç dosya sistemi

Bunların dışında desteklenen diğer dosya sistemleride şunlardır:

Dosya Sistemi	Adı
FDESC	File Descriptor Filesystem
HPFS	OS/2 Dosya sistemi
NTFS	Windows NT Dosya Sistemi
NULLFS	NULL Dosya sistemi
NWFS	NetWare Dosya Sistemi
PORTAL	Portal Dosya sistemi
SMBFS	SMB/CIFS Filesystem (Windows paylaşılan dosyalar için)
UMAPFS	UID Map Dosya Sistemi
UNION	Union Dosya sistemi
CODA	CODA Filesystem
EXT2FS	Ext2/Ext3FS Dosya sistemi (Linux)
REISERFS	ReiserFS Dosya sistemi (Linux)

Burada adı geçen dosya sistemlerinin bazıları için sadece okunur olarak bağlanabilmektedir. Bazıları görece olarak daha kararlı olan dosya sistemleri olsalar da bazı nedenlerden ötürü GENERIC çekirdekte yer almazlar.

BSD için çekirdekte yer alamayan dosya sistemleri aslında çekirdek kodu içinde yer alan ancak modül olarak derlenmemiş veya derlenmiş ama açılışta yüklenmemiş modüller olarak bulunur. Bu yazı hazırlandığı sırada Ext2FS, ReiserFS, FDESC, CODA, PORTAL, NWFS, NULL, NTFS, UNION, SMBFS modülleri GENERIC kernelde yer almakta ancak açılışta yüklenmemekteydi. /boot/kernel içinde bu modülleri bulabilirsiniz.

Dosya sistemini kendiniz elle bağlamak ve ayırmak isterseniz /sbin altında bu dosya sistemleri için hazırlanmış olan mount araçlarını bulabilirsiniz. Bunların bazılarını aşağıda listeledik. Kullandığınız BSD sürümüne göre bu isimler değişiklik gösterebilir.

```
mount_cd9660     mount_linprocfs   mount_nfs4
      mount_portalfs    mount_std
mount_devfs      mount_mfs         mount_ntfs
      mount_procfs      mount_udf
mount_ext2fs     mount_msdosfs     mount_nullfs
    mount_reiserfs    mount_umapfs
mount_fdescfs    mount_nfs         mount_nwfs
      mount_smbfs       mount_unionfs
```

Yukarıda adı geçen dosya sistemlerini mount ile bağlarken

```
# mount -t nfs /mnt
```

ile işlem yapmak yerine

```
# mount_nfs /mnt
```

kullanabiliriz. -t ile tanımlayacağımız dosya tipi yerine doğrudan ilgili dosya tipi için hazırlanmış olan mount_* kullanmak daha kolaydır. Ancak bu araçları kullanarak dosya sistemini bağlamadan önce çekirdek tarafından desteklenen işlemler ile dosya sistemine ilişkin kısıtlamaları bilmekte yarar var. Bu nedenle işlem yapmadan önce ilgili mount_* kılavuz sayfasına göz atınız.

Windows Dosya Sistemlerini Bağlamak/Ayırmak

Windows kurulu olan disk bölümlerini bağlamak için mount_msdosfs veya mount_ntfs kullanabiliriz. Ancak her iki araç için yerel, yani dosya isimleri için varsayılan dil kodu iso-8859-1'dir. Bu durumda bazı dosya isimlerini görüntülerken veya yazarken olası sorunların önüne geçmek için -L seçeneği ile dil tanımlaması yapılaması gerekir. Microsoft'un kullandığı dil ayarlarını BSD'de tanımlayarak ilgili disk bölümlerini bağlayabilirsiniz.

Micrsoft dil kodları

http://www.microsoft.com/globaldev/reference/oslocversion.mspx

MSDOSFS için yazma ve okuma desteği tüm BSD sürümlerinde bulunurken NTFS yazma desteği son sürümlerde bulunmaktadır. Bu nedenle NTFS disk bölümlerini oku-yaz olarak bağlamadan önce ilgili kılavuz sayfasına göz atınız.

Buradaki örneklerde root olarak bağlama işlemi yaptığımız için normal kullanıcı hesaplarından erişilemediği düşünülebilir. Tersine olarak MSDOSFS dosya sistemi normal kullanıcılar taarfından da bağlanabilir. Bunun için UID/GID ve umask değerleri tanımlanabilir. Ayrıca kısaltılmış dosya isimleri yerine uzun dosya isimlerini de kullanabiliriz. Bunun için mount_msdosfs kılavuz sayfasında ayrıntılı bilgi bulunmaktadır.

Linux Dosya Sistemlerini Bağlamak

Linux dağıtımlarının büyük bir bölümü Ext2fs/Extfs dosya sistemini kullanır. Bu dosya sistemleri BSD çekirdeğince desteklenir. Bazı yapılandırmalarda ext2fs.ko, çekirdeğin ext2fs dosya sistemine desteği veren modülü yüklenmemiş olabilir. Bu durumda ext2fs.ko modülünü elle yüklemek ve ardından mount -t kullanmanız gerekir. Bunun yerine mount_extfs kullanarak gereken modüller otomatik olarak yüklenir ve Linux dosya sistemini tercih ettiğiniz dizin altına bağlayarak işlem yapabilirsiniz. Bu işlemin tek sıkıntısı yüklenen ext2fs dosya sisteminin işlem bittikten sonra yüklü olarak kalmasıdır. Eğer sisteminizde sürekli olarak Linux disk bölümlerini bağlayıp ayırmak vs. üzerinde işlem yapacak iseniz GENERİC kernel yerine ext2fs dosya sistemi desteğine sahip bir çekirdek derleyerek kullanmayı tercih edebilirsiniz. çekirdek derleme konusunu ileride ele alacağız.

```
# mount_ext2fs /dev/ad1s1 /mnt
```

Burada BSD disk bölümlendirme konusunda bilgi vermek yararlı olacaktır. Windows ve Linux için disk bölümleri -partition- aynı anlama gelir. BSD için ise disk bölümleri yerine dilimler -slices- kullanılır. Disk isimleri ile her bir disk üzerindeki bölüm disk adı, dilim, alt_dilim-adı olarak geçer. Windows disk bölümü örneğindeki ad1s1 gibi. BSD'de de diğerlerinde olduğu gibi bir disk diliminde -slice- alt dilimler yaratılabilir. Bu alt dilimler a,b,c, vb. isimler ile anılır. Örneğin /dev/ad1s1g gibi.

MSDOS ve Linux için birincil bölümde sadece dört adet disk bölümü oluşturulabilir. Daha sonra oluşturulacak olan bölümler uzatılmış bölümlerde oluşturulur. BSD sisteminizden bu uzatılmış -extended- disk bölümlerini bağlamak isterseniz "Invalid argumanet" hatası alabilirsiniz. Uzatılmış bölümler 5 ile başladığı için /dev/ad1s5 yerinde olur.

Eğer tam olarak bilemiyorsanız fdisk kullanarak disklerdeki bölümleri listeleyebilir ve gereken bilgiyi edinebilirsiniz. Aşağıdaki örnek çıktının benzerini kendi sisteminizde de görebilirsiniz

```
# fdisk /dev/ad1
******* Working on device /dev/ad1 *******
parameters extracted from in-core disklabel are:
cylinders=1247 heads=255 sectors/track=63 (16065 blks/cyl)

Figures below won't work with BIOS for partitions not in cyl 1
parameters to be used for BIOS calculations are:
cylinders=1247 heads=255 sectors/track=63 (16065 blks/cyl)

Media sector size is 512
Warning: BIOS sector numbering starts with sector 1
Information from DOS bootblock is:
The data for partition 1 is:
sysid 131,(Linux filesystem)
    start 63, size 2104452 (1027 Meg), flag 0
        beg: cyl 0/ sector 1/ head 1;
        end: cyl 130/ sector 63/ head 254
The data for partition 2 is:
sysid 130,(Linux swap or Solaris x86)
    start 2104515, size 787185 (384 Meg), flag 0
        beg: cyl 131/ sector 1/ head 0;
        end: cyl 179/ sector 63/ head 254
The data for partition 3 is:
```

```
sysid 131,(Linux filesystem)
    start 2891700, size 17141355 (8369 Meg), flag 0
        beg: cyl 180/ sector 1/ head 0;
        end: cyl 1023/ sector 63/ head 254
The data for partition 4 is:
<UNUSED>
#
```

Her bir disk bölümündeki/dilimdeki dosya sistemi kendine ait olan bir sysid'ye sahiptir. EXT2FS dosya sistemi için bu sayı 131 iken BSD disk bölümleri için 165'dir. Eğer kurulumda BSD kaynak kodlarını kurduysanız disk bölümlerine ait olan sayılar hexadesimal olarak /usr/src/sbin/fdisk/fdisk.c altında yer almaktadır.

Optik Sürücüleri (CD-ROM) ve Disketleri Bağlamak ve Ayırmak

Dosya sistemlerini, ayırma ve bağlama işlemlerinin nasıl yapılacağını biliyoruz. Bunu diğer dosya sistemleri içinde yapabilirsiniz. Başlığa bakınca disket mi kaldı diyenler olabilir. Ancak bazı acil durumlarda sisteminizi normal olarak başlatamıyorsanız, disket sürücü ile bir tane 3.5" disket işinizi çok kolaylaştıracaktır. BSD, 3.5" disketler kullanılarak boot edebilir.

Optik sürücüler için kullanılan dosya sistemi CD9660 dosya sistemidir. Disket ise MSDOS veya BSD sisteminizde biçimlendirdiyseniz FFS dosya sistemine sahiptir. Eğer Linux makinada aynı işlemi yaptıysanız Ext2FS kullanılmıştır.

Optik sürücüleri bağlama/ayırma işleminin püf noktası kullanacağınız donanımın adını bilmektir. IDE sürücülere sahipseniz optik sürücünüzün adı /dev/acd0c veya SCSI bir sürücü ise /dev/cd0c olarak geçer. Bunun dışında standart olmayan bazı sürücüler, örneğin Sony gibi, farklı isimler ile kullanılırlar. BSD ile ilgili çeşitli kaynaklarda yukarıda adı geçen sürücüler farklı notasyonlarda gösterilebilir. BSD için bir sürücünün tamamen o dosya sisteminde formatlandığını anlatmanız yani son harf olarak "c" kullanmanız gerekir. BSD için o sürücüsünün o dosya sistemine atanmış olması anlamına gelir. Bu konuda en güvenilir kaynak FreeBSD elkitabıdır (1).

Bir örnek:

```
# mount_cd9660 /dev/acd0c /cdrom
```

Masaüstü ortamlarında ise bu işlemler kullandığınız masaüstü ortamına ait araçlar tarafından yapılmaktadır. DesktopBSD bu işler için desktopbsd-tools kullanırken, PCBSD düşük kapasiteli diskler için masaüstü ortamının araçları ile HAL kullanmaktadır. 128GB ve daha büyük disklerde bu işlem elle yapılmalıdır.

Disketleri bağlamak ve ayırmak ise daha kolay bir işlemdir. Disket /dev/fd0 olarak tanımlıdır. Eğer kasanızda yer varsa ve iki tane diset sürücüsü kullanıyorsanız bu durumda ikinci disket sürücüsü /dev/fd1 olacaktır. Disket dizini altına bağlamak için

```
# mount /dev/fd0 /disket
```

MSDOS -Windows sistemde- hazırlanmış bir disket için ise

```
# mount_msdosfs /dev/fd0 /disket
```

Burada bir uyarı yapmakta yarar var. BSD için bağladığınız optik sürücü ile disketin yazılabilir veya yazma koruması olup olmadığını bilmesi mümkün değildir. Bunu bağlama işlemini yapan kişinin tanımlaması gerekir. Disketin arka tarafındaki tırnağı kullanarak yazma korumalı hale getirdiğinizde disketinizin yazma koruması olduğu yani salt okunu olduğunu belirtmeniz gerekir. Aksi halde disketinizi bağlarken "I/O error" mesajını alırsınız. Bu olabilecek olan bir çok senaryo içinde en iyimser olanı. Zira hatanın olası sonuçları diskete erişmek isteyen uygulamaya bağlı olarak değişkenlik gösterecektir. Bu işlem CD-ROM sürücüler içinde geçerlidir.

Yazma korumalı bir disket için

```
# mount -r /dev/fd0 /disket
```

veya

```
# mount -rdonly /dev/fd0 /disket
```

Optik sürücüleri ve disketleri bağladıktan sonra ayırmak ilk bakışta kolay gibi gelse de bazı püf noktalarına dikkate etmek gerekir. Bu sürücüler fiziksel olarak sürücüden -sisteme ayırma işlemi yapmadan- çıkartılabilir.

Windows kullanıcıları için bu işlem sistem tarafından sürücü bağlandıktan sonra dinamik olarak sürücünün durumu izlenir ve güncellenir. Eğer disketi veya optik sürücüyü çıkartmış iseniz Windows bunu fark edecektir. Mac OS X için ise bu kontrol işlemi yazılım tarafına aktarılmıştır. Bu işlemler için ayrılmış bir yazılım ile sürücülerin durumları kontrol edilir. Kullanıcının yapacağı her işlem bu yazılıma bağımlı olarak gerçekleşir. BSD için ise bu derece karmaşık çözümlere gerek yoktur. Bu sorumluluk kullanıcıya bırakılmıştır.

Güncel optik sürücüler ise bu konuda önlem alarak olası bir hatanın önüne geçecek mekanizmaya sahiptir. Sürücüde bulunan bir kilit mekanizması, sürücünün bağlı/ayrılmış durumunu işletim sisteminden öğrenerek sürücüyü kilitler. Bu kilit mekanizması ise sürücü ayrılana dek çalışır durumdadır. Sürücüyü ayırdığınızda çıkartma düğmesine basarak çıkarabilirsiniz. Aksi durumda ise olası her türlü hataya karşı hazırlıklı olmak gerekir.

USB Diskler

USB bağlantısı ile kullanılabilen yüksek kapasiteli diskler giderek yaygınlaşmakta. Bu tür sürücüleri kullanmak isteyenler için diskin dosya sistemi ile sürücünün adının doğru girilmesi gerekir. Bir de buna ek olarak bağlayacağınız dizini de tanımlamanız gerekir. USB sürücüleri "umass device" mesajı ile çekirdek mesajında görebilirsiniz. Disket ve cd-rom bağlama veya ayırma işlemleri ile bu tür donanımları bağlayabilir, ayırabilir ve kullanabilirsiniz. DesktopBSD tools bu işlemi sizin için kolaylaştırmaktadır. FreeBSD ile kullanabilirsiniz.

Dosya Sistemlerinin Bağlanacağı Dizinler: /etc/fstab

Bağlama ve ayırma işlemlerini daha önceden grafik arayüzlere sahip araçlar ile yapageldiyseniz BSD işleyişi garip gelecektir. Bu konuda en sık karşılaştığım soru bunları nasıl bilebilirim olmuştur. BSD ve diğer UNIX türevi sistemlerde ağ üzerinden NFS dosya sistemlerine erişim, sisteme ikinci veya üçüncü bir disk takmak, optik sürücünüzü kullanmak gibi işlemler için tüm bağlanma noktalarını ve komutları bilmek kolay değildir. Bunu bizim için /etc/fstab dosyası yapar.

Dizüstü bilgisayarımdaki yapılanma

```
[goksin@bsd.enixma.org /usr/home/goksin]$ cat /etc/fstab
# Device      Mountpoint  FStype          Options Dump    Pass#
/dev/ad4s1a   /           ufs             rw      1       1
/dev/ad4s1b   none        swap            sw      0       0
/dev/ad4s1d   /var        ufs             rw      2       2
/dev/ad4s1e   /usr        ufs             rw      2       2
/usr/tmp      /tmp        nullfs          rw      0       0
procfs        /proc       procfs          rw      0       0
linprocfs     /usr/compat/linux/proc  linprocfs  rw  0    0
[goksin@bsd.enixma.org /usr/home/goksin]$
```

masaüstü bilgisayarımdaki yapılanma

```
[goksin@bsd1.enixma.org /usr/home/goksin]$ cat /etc/fstab
# Device      Mountpoint      FStype  Options         Dump  Pass#
/dev/ad0s1b   none            swap    sw              0     0
/dev/ad0s1a   /               ufs     rw              1     1
/dev/ad1s1f   /tmp            ufs     rw              2     2
/dev/ad0s1g   /var            ufs     rw              2     2
/dev/ad0s1e   /usr            ufs     rw              2     2
/dev/acd0     /cdrom          cd9660  ro,noauto       0     0
/dev/fd0      /floppy         msdos   rw,noauto       0     0
proc          /proc           procfs  rw              0     0
/dev/ad1s1g   /home           ufs     rw              2     2
[goksin@bsd1.enixma.org /usr/home/goksin]$
```

fstab dosyası sisteminizdeki disklere ve kuruluma göre değişkenlik gösterecektir ama yukarıdaki örneklere yakın çıktılar görebilirsiniz. fstab dosyası, hangi donanımın hangi dizin altına bağlanacağını ve dosya sisteminin ne olacağını, bağlanırken hangi seçeneklerin kullanılacağını tanımlar. Bunlara ek olarak da sistem açılışı sırasında dosya sisteminin hangi sırayla kontrol edileceği tanımlıdır.

fstab dosyasının temel işlevi, sisteme, açılış sırasında bağlanacak olan dosya sistemlerini tanımlamaktır. Tüm bağlama dizinleri tanımlı olduğu için bağlanacak olan donanımları bağlamak ve ayırmak son derece kolay olacaktır.

Sistem açılışı sırasında fstab dosyasında yer olan dosya sistemleri "mount -a" ile bağlanır. Bu işlemin ardından ise "fsck -p" çalıştırılır ve dosya sistemleri kontrol edilir. Dosya sistemleri "temiz-clean" olarak

işaretli ise sistem olağan açılış sürecini tamamlar. Aksi halde fsck ile dosya sistemi kontrol edilerek hatalar giderilir ve açılış bu işlemden sonra devam eder. Bu aşamada sistem dosya sistemlerini bağlamak için yeniden "mount -a -t nonfs" çalıştırır. Bu komut, ağ üzerinden erişilen NFS vb. dosya sistemleri hariç fstab'ta tanımlı dosya sistemlerini bağlar.

fstab, sadece açılışta dosya sistemlerinin kontrolünde görev almaz. Yukarıda yer verdiğimiz bağlama ve ayırma işlemlerini gerçekleştirmek için işlerimizi kolaylaştırır. Sadece bağlame işleminde kullanılacak olan dizinin adını kullanarak ilgili dosya sistemini bağlayabilir ve ayırabiliriz.

Masaüstü sistemimde ev dizini /home dizine bağlandığı için bakım vb. işlemlerde bu disk bölümünü bağlamak için

```
# mount /home
```

yeterli olmaktadır. Yukarıda kullandığım komut, bağlanma işleminin gerçekleştirilmesi için gereken tüm bilgileri /etc/fstab dosyasından okuyarak gerçekleştirmektedir. Yine aynı sistemde CDROM sürücüyü bağlamak için benzer biçimde

```
# mount /cdrom
```

kullanmak yeterli olmaktadır.

Yukarıdaki masaüstü sistemine ait olan fstab dosyasında dördüncü sütunda bulunan noauto seçeneği, sözkonusu dosya sisteminin açılış sırasında bağlanması içindir. Optik sürücüler, disketler ve ağ üzerinden erişilebilen dosya sistemleri sistemin açılışında erişilibilir olmayabilir. Eğer noauto seçeneğini kullanmasaydık, sistem söz konusu dosya sistemlerini bağlamaya çalışacaktı. Bunun yerine bu dosya sistemlerini daha sonra bağlamak/ayırmak için bu seçenek kullanılmalıdır. Bağlamak için yukarıdaki örneklerde olduğu gibi bağlanacağı dizinin adını kullanarak dosya sistemini bağlayabilir, ayırabilirsiniz. Dördüncü sütünda yer alan değerler, mount kılavuz sayfasından öğrenilebilir. Bunun dışında yukarıda adı geçen diğer işletim sistemlerine ait olan dosya sistemlerine ait olan değerler, mount_* kılavuz sayfasında yer alır ve ilgili kılavuzdan öğrenilebilir.

Beşinci sütunda yer alan değer dump komutuna ilişkin değerlerdir. Dump aracı farklı seviyelerde yedekleme yapmak, bir başka deyişle bellekte yer alan verinin diske veya bir başka ortama yazılmasına ilişkin bir kontrol değeridir. dump, tanımlanan öncelik, yani kılavuz sayfasındaki düzeye göre çalışır. fstab dosyasında tanımlanan değer 1 ise, bu durumda dump ancak ve ancak 1 veya daha alt seviyelerde işlem yapar. 0 tanımlarsanız bu durumda o dosya sistemine ait olan bilgi dump ile yedeklenmez. Takas, CDROM ve disket için bu değer 0 alınmıştır.

En sonda yer alan altıncı sütün ise fsck için tanımlanmış değerlerdir. Sıfırdan büyük olan sayılar fsck'in dosya sistemlerini hangı sırayla kontrol edeceğini gösterir. Root / dosya sistemi için bu değer daima bir olur. Bunun anlamı / dosya sisteminin ilk olarak kontrol edilecek dosya sistemi olduğudur. 2 yazan dosya sistemleri 1 değerine sahip olanlardan sonra kontrol edilir. Eğer sıfır değeri varsa bu dosya sistemleri kontrol edilmez. Bunlar optik sürücüler, disketler, ağ üzerinden erişilebilen dosya sistemler ve benzerleridir.

Dosya Sistemini fsck ile Kontrol Etmek

fsck (Filesystem Consistency Check) diğer işletim sistemlerinden de tanıdığınız, bildiğiniz veya kullandığınız disk bakım/kontrol programları ile benzerdir. Açılış sırasındaki çalışması ve de işleyişini düşündüğümüzde %100 aynı işi yapar diyemesek de, fsck açılış sırasında sisteme dosya sistemleri bağlanmadan önce çalışarak dosya sistemlerinin "sorunsuz" olup olmadığını kontrol eder. "Sorunsuz" veya bazı Türkçe kaynaklarda belirtildiği gibi "temiz" olan dosya sistemleri sonradan bağlanır. Bu işlem ingizice "preen" olarak adlandırılır ve sözlük anlamı olarak kuşların tüylerini düzeltmesi, kontrol etmesi, temizlemesi işlemine karşılık gelir. Bilgisayar açısından ise dosya sistemini kontrol edip sorunların giderilmesinden sonra kullanılmasıdır. Doğanın yaptığı bir işi kopya ediyoruz. Elle bu işlemi yapmak istersek fsck -p kullanırız. fsck kontrol işlemi sırasında dosya sisteminde olabilecek sorunları kontrol eder. Eğer bilgisayarı doğru biçimde kapatmadıysanız, reset tuşuna basarak veya elektirk kesintisi vb. durumlarda disk üzerindeki dosya sisteminde hatalar oluşacaktır. fsck bu durumu kontrol ederek "temiz/sorunsuz - clean" olmayan dosya sistemlerindeki hataları giderir.

fsck'i BSD kullanırken çalıştırdım diyebileceğiniz durum sayısı belki sistemi kullanırken -sunucularda kullandığımız bir BSD sisteme bir elektrik kesintisi sırasında oluşan hatayı gidermek için kullandığımız bir olay hariç- bir elin parmaklarını geçmeyecektir. Öte yandan fsck geri planda açılış sırasında zaten çalışmaktadır. Bu sürece ait görebileceğiniz

yegane çıktı ise açılış sırasında görebileceğiniz aşağıdakine benzer bir mesaj olacaktır:

```
/dev/ad4s1a:FILE SYSTEM CLEAN; SKIPPING CHECKS
/dev/ad4s1a,clean,88773 free (93 frags,11085 blocks 0,1%
fragmentation)
/dev/ads4s1d:FILE SYSTEM CLEAN; SKIPPING CHECKS
/dev/ads4s1d,clean,129590 free (12606 frags, 14623 blocks
5,0% fragmentation)
/dev/ads4s1e:FILE SYSTEM CLEAN; SKIPPING CHECKS
/dev/ads4s1e,clean,8539462 free (610054 frags,991176 blocks
1,6% fragmentation
```

Bu okuduğunuz çıktıda "fragmentation" gördüğünüzde endişelenmeniz için bir neden yok. Zira BSD, dosya sisteminde diğer işletim sistemlerinin dosya sistemlerinde karşılabileceğiniz "fragmentation/parçalanma" aynı anlama gelmemektedir. UNIX dosya sistemleri, disk üzerindeki ilgili sektörlerin bir arada kalmasını sağlayacak biçimde tasarlanmıştır. Bu işlem dosya sisteminde birleştirme işlemine gerek duyulmadan diskinize okuma yazma işlemleri yaparken gerçekleşir. Dolayısıyla da UNIX dosya sistemleri disk birleştirme gibi işlemlere gerek duymaz. Disk üzerindeki verinin saklanam şekli üzerinde burada bilgi vermek yararlı olacaktır. Disklerde bir data block -veri bloğu- verinin saklanması için geliştirilmiş olan bir birimdir ve boyutu 8192 bayttır. Blok aynı zamanda 1024 baytlık sekiz parçadan oluşur. Bir dosya bu bloklar üzerindeki parçalara kayıt edilir. Dosya boyutu büyük olduğunda da tek bir blok yeterli olmaz ve birden fazla blok kullanılır. haliyle de dosya boyutuna bağlı olarak her bir blok tam olarak doldurulamayacaktır. fsck açılış sırasında parçalanma olduğunu verirken aslında bu doldurulamayan bölümlere ve bloklara atıfta bulunmaktadır. Sonradan kayıt edilen yeni bir dosyada ilk boş blok veya bölümden itibaren yazılmaya başlar. Eğer var olan dosyaya yeni eklenen veriler ile dosya büyürse ve kullandığı alana sığmaz ise bu durumda başka bir yere taşınır. BSD için ise dosya boyutu 8192 bayttan az ise bunu daima tek bir blok üzerinde saklar.Daha büyük boyutlu dosyaları da daima ardışık bloklar üzerinde saklayarak okuma/yazma performansını korur. Diğer işletim sistemlerinde ise sıradaki bloktan başlanarak yazıldığı için okuma/yazma verimi düşer.

Bir çok kişinin aklında fsck'in sadece açılış sırasında karşımıza geleceği sorusu gelebilir. Yukarıda verilen örneklerde olduğu gibi dosya sistemi doğru biçimde kapatılmamış bir bilgisayarda açılış sırasında fsck hata mesajları döndürecektir. Bunun nedeni ise UNIX dosya sistemleri, dosyalara ilişkin bilgileri metadata olarak eşzamanlı olarak kayıt ederek saklaması ve bunun da diske yazma işlemlerinin tekrarlanması anlamına gelir. Eğer bu yazma işlemi sırasında bilgisayar doğru biçimde kapatılmadıysa dosya sisteminde yazma hatası olmuştur.

Eğer açılışta dosya sisteminiz "unclean" olarak karşınıza gelirse fsck, diskinizdeki tüm blokları teker teker kontrol etmeye başlar. Her bir blok üzerinde yer alan metadata doğruluğu kontrol edilir. Bu işlem donanımınıza bağlı olarak biraz zaman alacaktır. Fsck bu kontrol işlemi sırasında hatalı olan metadatayı düzeltmeye çalışır. Eğer bu işlem otomatik olarak yapılamıyorsa, kullanıcıdan onay istenir. Burada bir noktayı açıklamak gerekir. Bazı kullanıcılar fsck onaramadıysa neden ben bu soruya evet demeliyim diyerek itiraz eder. fsck bu işlemi gerçekleştirdiğinde ilgili blokta yer alan veride kayıp olma olasılığı söz konusudur. Bu veri kaybı ise sadece ve sadece kesintiye uğrayan yazma işlemine ait olan veriyi kapsar, diğerlerini kapsamaz. Eğer düzenli olarak yedek alıyorsanız disk üzerindeki veri kayıplarının da bu işlem sonucunda çok çok az olacağı için size ciddi bir kaybı olmayacaktır. Disk üzerinde bozuk metadata verisi bir sonraki açılışta yine size merhaba diyebilir. fsck tarama işlemi başladığında

```
#
```

ekranı görebilirsiniz. fsck işini bitirdiğinde ise yine

```
#
```

görünümüne geri dönülür. Bu aşamadan sonra sistemi açmak için

```
# continue
```

yeniden başlatmak için ise

```
# reboot
```

kullanılır. Yeniden başlatmak dosya sisteminde ortaya çıkabilecek olan olası bir sorun olup olmadığını görmek için uygun olacaktır. Sistem açılışa devam eder diyip, makinanın başından ayrılmayın.

fsck'i normal bir çalışma sırasında da kullanmak isteyebilirsiniz. Her ne

kadar bu işlemi root olarak yapmak isteseniz de yapamazsınız. Bunun yerine sistemi yeniden başlatmalı ve tek kullanıcı girişi -single user- ile başlatmanız gerekir. Ancak bu şekilde fsck'i elle çalıştırabilirsiniz. Bu işlemi de uzaktan yapmanız olanaklı değildir. Eğer uzaktan erişerek bu işlemi yapmak isterseniz kasanın başına gitmekten başka seçeneğiniz yoktur.

fsck'i tek kullanıcı modu ve açılış sırasında kullanmak dışında sisteme ikinci bir disk takarken kullanabiliriz. Bunun püf noktası ise fsck'in /etc/fstab tanımlı dosya sistemleri üzerinde işlem yapmasıdır. Yeni bir diski sonradan eklediğimizde ve bunun üzerinde yeni bir disk bölümü oluşturan işlemi vb. yaptığımızda söz konusu ikinci diski fsck ile kontrol edebiliriz. Bu işlemi doğrudan

```
# fsck -p /dev/yeni_disk/
```

ile yapabiliriz. Bu diski henüz sisteme eklemediğimiz ve üzerinde kimse işlem yapmadığı için fsck kullanılabilir.

Günlüklü Dosya Sistemleri ve Soft Updates

Disklere ait metadatanın eş zamanlı olarak yazılması sonucunda ortaya çıkan bölünme ve kaybolan verilerin önüne geçilmesi için bir çok çözüm geliştirilmiştir. Bunun için en iyi bilinen örnekler Ext3FS dosya sistemidir. Bu dosya sisteminde yapılan tüm yazma vb. işlemler gerçekleştirilmeden önce kayıt edilir. Kayıt işlemi tamamlandıktan sonra ise gerçekleştirilir. fsck bu kayıt dosyalarına dayanarak diskteki ilgili bölümdeki hatayı giderir, tüm diski baştan sona taramaz.

BSD, günlüklü dosya sistemi kullanmaz. Onun yerine aynı işlevi yerine getiren Soft Updates kullanır. Soft Updates'in tam olarak Türkçe karşılığını bulamadığım için terimi İngilizcesi ile kullanmayı tercih ettim. Soft Updates default veya GENERIC çekirdekte freeBSD 4.5 sürümünden bu yana yer alır. Yeni oluşturacağınız tüm dosya sistemlerinde otomatik olarak aktifleştirilir. Soft Updates yazma işlemlerinin kaydını tutarken bir kayıt dosyası kullanmaz, onun yerine özel bir teknik kullanarak metadata verisinin bütünlüğünü kontrol edilmek üzere saklar. Bu işlemin yararı, günlüklü dosya sistemlerine göre açılış süresini oldukça kısaltmakta oluşudur. Bunun dışında dosya sisteminin bütünlük kontrolü ise geri planda ve diske ait "snapshot-görüntü" üzerinde gerçekleştirilir.

Soft Updates özelliği kurulum sırasında Disk Label Editor -diskinizi bölümlendiren sysinstall uygulaması- tarafından gerçekleştirilir. Soft Updates özellikle sık sık okuma-yazma işlemi yapılan /var ve /usr gibi disk bölümlerinde çok yararlı olur. Soft Updates ile kullanabileceğimiz uygulamalar ports üzerinde yer almaktadır. /usr/ports/sysutils'e göz atın. Soht updates hakkında daha fazla bilgi için (2)'ye bakınız.

fsck ile Super Block Onarılması

Yukarıda sözünü ettiğimiz diskin bütünlük kontrolü sürecinde fsck diğer işletim sistemlerine göre daha etkili ve kolayca kullanılmaktadır. BSD için dosya sisteminde ciddi bir hata olması durumunda sistemin açılmaması olasılığı çok çok düşük olsa da, karşı karşıya kalmayacağımız anlamına gelmez. Bu tür durumlara karşı hazırlıklı olmak gerekir.

Dosya sistemlerinde olabilecek olası hatalardan birisi diskinizdeki dosya sistemini tanımlayan bloğun bir nedenle okunamaması durumudur. Diğer bir deyişle superblok hatası. Adı son derece ürkütücü olan bu durum, diskinizdeki dosya sistemini tanımlayan 16. ve 31. sektörler arasında kalan bloklardaki veri ile ilgili bir durumdur. BSD bu duruma önlem olarak her bir silindir grubu için yedek bir süper blok bulundurur. Bir süper blok hatasının oluşması durumunda fsck bu yedekleri kullanarak onarımı gerçekleştirir. İlk yedek 32. blokta yer alırken, diğer yedekler disk üzerinde farklı bloklarda yer alır.

Olası bir süper blok hatasında /dev/disk /mnt: Incorrect super block hata mesajını görebilirsiniz. fsck'i çalıştırıp hata mesajını aldığınız disk bölümünü kontrol etmeniz durumunda fsck ilgili hataları gidermek için onayınızı isteyecektir.

```
# fsck /dev/ad0s1e
 ** /dev/ad0s1e
 BAD SUPER BLOCK: MAGIC NUMBER WRONG
LOOK FOR ALTERNATE SUPERBLOCKS? [yn] y
USING ALTERNATE SUPERBLOCK AT 32
** Last Mounted on /home5
** Phase 1 - Check Blocks and Sizes
** Phase 2 - Check Pathnames
** Phase 3 - Check Connectivity
** Phase 4 - Check Reference Counts
** Phase 5 - Check Cyl groups
```

```
148 files, 15660 used, 7038840 free (208 frags, 879829
blocks, 0.0%
fragmentation)
UPDATE STANDARD SUPERBLOCK? [yn] y
***** FILE SYSTEM WAS MODIFIED *****
```

Süper blok hatası giderilmiş durumda. fsck, BSD sistemlerde var olan yedeklerden birisini kopyalayarak sorunu çözer. Ayrıca yedek süper blok yerini tanımlamanız gerekmez.

Eğer diskinizde yer alan yedek süper blokları görüntülemek isterseniz newfs -N kullanın.

```
[root@bsd.enixma.org /usr/ports]# newfs -N /dev/ad4s1a
/dev/ad4s1a: 256.0MB (524288 sectors) block size 16384,
fragment size 2048
        using 4 cylinder groups of 64.02MB, 4097 blks, 8256
inodes.
super-block backups (for fsck -b #) at:
 160, 131264, 262368, 393472
[root@bsd.enixma.org /usr/ports]# newfs -N /dev/ad4s1d
/dev/ad4s1d: 512.0MB (1048576 sectors) block size 16384,
fragment size 2048
        using 4 cylinder groups of 128.02MB, 8193 blks, 16448
inodes.
super-block backups (for fsck -b #) at:
 160, 262336, 524512, 786688
[root@bsd.enixma.org /usr/ports]# newfs -N /dev/ad4s1e
/dev/ad4s1e: 75036.6MB (153674880 sectors) block size 16384,
fragment size 2048
        using 409 cylinder groups of 183.72MB, 11758 blks,
23552 inodes.
super-block backups (for fsck -b #) at:
 160, 376416, 752672, 1128928, 1505184, 1881440, 2257696,
2633952, 3010208, 3386464, 3762720, 4138976, 4515232,
4891488, 5267744, 5644000, 6020256,
 6396512, 6772768, 7149024, 7525280, 7901536, 8277792,
8654048, 9030304, 9406560, 9782816, 10159072, 10535328,
10911584, 11287840, 11664096, 12040352,
 12416608, 12792864, 13169120, 13545376, 13921632, 14297888,
14674144, 15050400, 15426656, 15802912, 16179168, 16555424,
16931680, 17307936, 17684192,
 18060448, 18436704, 18812960, 19189216, 19565472, 19941728,
20317984, 20694240, 21070496, 21446752, 21823008, 22199264,
22575520, 22951776, 23328032,
 23704288, 24080544, 24456800, 24833056, 25209312, 25585568,
25961824, 26338080, 26714336, 27090592, 27466848, 27843104,
28219360, 28595616, 28971872,

.....

 133947296, 134323552, 134699808, 135076064, 135452320,
135828576, 136204832, 136581088, 136957344, 137333600,
137709856, 138086112, 138462368, 138838624,
 139214880, 139591136, 139967392, 140343648, 140719904,
141096160, 141472416, 141848672, 142224928, 142601184,
142977440, 143353696, 143729952, 144106208,
 144482464, 144858720, 145234976, 145611232, 145987488,
146363744, 146740000, 147116256, 147492512, 147868768,
148245024, 148621280, 148997536, 149373792,
 149750048, 150126304, 150502560, 150878816, 151255072,
151631328, 152007584, 152383840, 152760096, 153136352,
153512608
[root@bsd.enixma.org /usr/ports]#
```

Eğer fsck b işlemi otomatik olarak yapmasını istemiyorsanız sizin tanımlayacağınız bir yedeği kullanmasını sağlayabilirsiniz. Kullanım şekli ise

```
# fsck -b YEDEK_SUPER_BLOK_ADRESİ /dev/disk_dilimi
```

Notlar:
(1)http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/disks-naming.html
(2)http://www.mckusick.com/softdep/ ve
http://www.ece.cmu.edu/~ganger/papers/CSE-TR-254-95/

Gökşin Akdeniz
<goksin@enixma.org>

Bsd - II

FreeBSD ile ilk adımlar

BSD ile ilgili yazı dizinin ilk yazısı ile ilgili en önemli eleştiriler yazının ileri düzey kullanıcılara yönelik olarak yazılmış olduğu idi. Bu konuya katılmakla birlikte yazı içinde kısaca BSD kullanmak isteyen kullanıcılara rahat bir başlangıç yapmak amacıyla FreeBSD'yi temel alan PC-BSD veya DesktopBSD kullanmalarını önermiştik. DesktopBSD, PC-BSD ile karşılaştırıldığında FreeBSD+desktopbsd-tools olarak özetleyebileceğimiz daha az özelleştirilmiş, kullanıcı dostu bir FreeBSD olarak karşımıza geliyor. PC-BSD yapısı ve kullandığı uygulamalar ve yapılandırma araçları ile DesktopBSD'ye göre daha fazla özelleştirilmiş olarak karşımıza çıkmakta.

Her iki BSD sisteminin de kendi açısından olumlu ve olumsuz olarak nitelendirilebilecek olan yanları bulunduğunu göz önünde bulundurmak gerekiyor. Bu nedenle BSD yazı dizisi, PC-BSD, DesktopBSD ya da bir diğer BSD sistem kullanıyorsanız yazı dizisinin FreeBSD 6'yı esas almakta ve FreeBSD 7.x ve ilerisi için de temel olabilecektir. Bu yazı FreeBSD kurulumu ve temel yapılandırmasını ele almaktadır.

FreeBSD Kurulumu

Daha önce bir işletim sistemi kurulumu yapmış iseniz, FreeBSD kurulumunu da gerçekleştirebilirsiniz. Kurulum işlemine geçemeden önce kurulum ve yapılandırma basamakları hakkında yeterli bilgi edinmek işleri kolaylaştıracaktır. Kurulum işlemini sabit diskinize yapacağınıza göre öncelikle verilerinizi yedeklemeliniz, kurulum sırasında sistemin temel yapılandırmasına ilişkin tercihlerinizi belirteceğiniz için önceden sistem, ağ ve ilgili diğer bilgilerin yazılı olarak elinizin altında olmasında yarar bulunmaktadır. Eğer disikinizde bir başka işletim sistemi bulunuyorsa, bu sistemin kurulu olduğu disk bölümüne zarar vermeden FreeBSD kurmak için yeterli alanı açmanız uygun olacaktır. FreeBSD bu işlem için kullanılabilecek olan araçlara sahip olsa da işlem sonucunda verilerinizda kayıp olmayacağını garanti etmemektedir. Sorumluluk size aittir.

Donanım Uyumluluğunu Kontrol Edin!

Öncelikle kurulum yapmayı düşündüğünüz donanımın desteklenip desteklenmediğini kontrol edin. FreeBSD geliştirme sürecinin 30 yıla uzanan bir geçmişi olmakla birlikte, güncel donanımların bir çoğunun çekirdek düzeyinde ve de sürücü bazında BSD desteği bulunmamaktadır. Bu nedenle compiz ve benzeri heveslerinizi tatmin edemeyebileceğinizi anımsatmakta yarar var.

FreeBSD kurulum ve yapılandırma aracı donanımı tanıyıp gereken yapılandırmayı otomatik olarak gerçekleştirecektir. Ancak yukarıda değindiğim gibi bazı donanımlar ile ilgili yapılandırma için sizin karar vermeniz gerekecektir. Genel olarak kurulum aşamasında şu bilgilere gereksinim duyabilirsiniz:

* Grafik kartınız ve bellek miktarı
* Monitörünüzün varsa kılavuzu, desteklediği çözünürlük, yatay ve düşey tarama hızı
* Modeminizin kullandığı COM portu, atanan IRQ değeri
* Fareniz ve bağlantı biçimi; PS/2 ya da USB
* Ağ kartınızın, bazı eski kartlar için ise IRQ değeri de gerekebiliyor, markası, modeli ve daha da önemlisi kartta kullanılan yonga seti
* Ağ kartınızda kullandığınız ağ yapılandırması
* Diskinizde kurulum için yeterli boş alan. Eğer grafik arabirim ve bsd-port kurmayı düşünüyorsanız en az 2 GB alan gerekecektir.

Kurulum Disketleri Oluşturmak

Bu kısım eğer CD/DVD ile kurulum yapma olanağınız yoksa veya ağ üzerinden kurulum yapmayı düşünüyorsanız, kurulum disketleri hazırlamak gerekecektir. Kurulum disketleri kernel ve küçük bir temel sistem için gereken bileşenleri barındırmaktadır. Kurulum disketleri için 3 ile 5 adet disket gerekecektir. Bu disketleri kullanarak sistemi disketten başlatın. FreeBSD kurulumu başlayacaktır. Disketler ile kurulum süreci kurulum için gereken tüm dosyaların ftp sunucusundan indirilip kurulmasını kapsamaktadır. Kurulum disketleri aslında ftp sunucularında hazır bulunur ve bunları doğrudan indirip disketlere kopyalayarak oluşturabilirsiniz.

FreeBSD ftp sunucularından indireceğiniz disk imajları /pub/FreeBSD/releases/mimari/sürüm/floppies dizini altında yer alır. 'mimari' kullandığınız işlemci, 'sürüm' ise kurmayı düşündüğünüz FreeBSD sürümüdür. Dizinde göreceğiniz *.flp dosyaları diskete kopyalayacağınız imaj dosyalarıdır. Bu dosyaları yazmak için gereken fdimage.exe aracını ise /pub/FreeBSD/tools altında bulabilirsiniz.

flp dosyalarını doğrudan diskete kopyalamak olanaklı değildir. Bu dosyaların tabir'i caiz ise diske kopyalanması gerekir. Bu işlemi Windows altında yapmak için gereken araçları ftp sunucularından indirmek gerekir. Linux/UNIX sistemlerde ise gereken araçlar elimizin altında yer almaktadır. Disket kullanacak iseniz yeni bir disket kullanmanız yerinde olacaktır. Kopyalama işlemi dosyaları doğrudan diske yazmaktadır. Diskte bir hata vb. kontrol edilmediği için kullanılmış bir diskette hata bulunması durumunda kurulum gerçekleşmeyecektir.

Windows Üzerinde BSD Boot Disketleri Hazırlamak

Windows üzerinde DOS penceresi açıp disketlere ait flp ve fdimage.exe dosyasının bulunduğu dizine geçin. Aşağıdaki örnekte dosyalar BSD_Discs dizininde yer alıyor.

```
C:\BSD_Discs> fdimage boot.flp a:
```

Bu işlemin ardından kern1.flp ve diğerleri için işlemi tekrarlamamız gerekir.

```
C:\BSD_Discs> fdimage kern1.flp a:
```

Linux/UNIX sistemler için ise dd kullanılarak bu işlem kolaylıkla gerçekleştirilebilir. FreeBSD sistemde bu işlem

```
# dd if=boot.flp of=/dev/fd0c
```

yapılabilir. Benzer olarak kern1.flp vd için de

```
# dd if=kern1.flp of=/dev/fd0c
```

yapılarak sırası ile diğer dosyalar da yazılabilir.

Linux sistemlerinde de ise yapılandırma /dev/fd0 veya /dev/floppy olabilir. Bu durumda fd0c yerine uygun aygıt adı yazılarak yukarıdaki gibi disketler hazırlanabilir. Disketler hazırlandıktan sonra kuruluma başlayabilirsiniz.

Kurulum Süreci

FreeBSD kurulumu için yukarıdaki adımda hazırladığımız disketleri kullanabilir veya bu işlemi doğrudan CD/DVD kullanarak da yapabilirsiniz. FreeBSD için disk bölümleri ile bölümlemeye ilişkin terimler farklıdır. Bu nedenle diskinizde eğer başka bir işletim sistemi bulunuyorsa FreeBSD kurulumu için birincil disk bölümünde yeterli alan oluşturmanız gerekecektir. Eğer FreeBSD disk bölümlendirme araçları ile terminolojisine yabancı iseniz, bu işlemi diğer işletim sistemlerinin araçları ile yapmanız veya bu işler için geliştirilmiş yazılımları kullanarak gerçekleştirmeniz uygun olacaktır.

Disketler ile kurulum aşamasında ilk olarak boot.flp dosyasını kopyaladığınız açılış disketi ile sisteminizi başlatın. CD/DVD ile başlatıyorsanız sisteminizi ekranınıza gelen yönergeleri izlemeniz yeterlidir. Disketler ile açılış sırasında çekirdeğin açılması için boot.flp ardından kern1.flp disketini takmanızı isteyen

```
"Insert disk labeled "Kernel floppy 1" and press any key"
```

mesajını gördüğünüzde kern1.flp yer aldığı disketi ve sırasıyla diğer kern2.flp, kenr3.flp disketlerini takarak kurulumu devam ettirmeniz gereklidir. En son olarak yeniden boot.flp disketini takmanız gerekecektir. Çekirdek yüklendikten sonra BSD metin tabanlı açılış logosu sizi karşılayacaktır. Varsayılan açılış seçeneği sizi FreeBSD kurulum aracı sysintall programını çalıştırıp kuruluma geçecektir.

Sysinstall

Sysintall, FreeBSD'nin kurulum ve temel yapılandırma aracıdır. Sysinstall, metin tabanlı bir uygulama olduğu için fare ile kullanılması olanaklı değildir. Onun yerine klavyeden aşağı ve yukarı ok tuşları ile tab, boşluk ve enter tuşlarını kullanmak gerekir. Bir çok kişi için sysinstall ürkütücü gelebilir ancak yapılandırma ve kurulum işlemini kolaylıkla yapmamıza olanak verir. Sysinstall ekran görüntülerine [1] FreeBSD sitesinden erişilebilir.

FreeBSD kurulumunu gerçekleştirmek için "standart - Begin a standart installation"u seçmeniz yerinde olur. Bu size standart bir kurulum sunar. X sunucusu ile grafik arabirim, belgeler, ports vb. dahil olmak üzere masaüstü bir sistem kurmak için gereken tüm bileşenler seçilmiş olur. Bu aşamadan sonra ise kurulum yapılacak diskin seçilmesi aşamasına geçilir.

Eğer kurulum yaptığınız sistemde tek bir sabit disk bulunuyorsa bu durumda kurulum için disk seçimi yapmanıza gerek yoktur. Doğrudan diski bölümleme aşamasına geçersiniz. Ancak sistemde birden fazla disk bulunuyorsa, bu durumda sizden kurulum yapacağınız diski seçmeniz istenecektir.

Disk seçimi aşamasından sonraki işlem disk üzerinde BSD kuracağınız bölümü belirlemek ve ardından da bu bölüme kurulumu gerçekleştirmek olacaktır. Ancak dikkat etmeniz gereken nokta ise BSD disk bölümlerini adlandırması ve kullanmasının diğer işletim sistemleri ile karşılaştırıldığında farklı olduğudur. Eski alışkanlıklarınız ile /, /var, swap, /usr ile /home için ayrı disk bölümleri ayırmak için diski elle bölebilirsiniz. Ancak hatalı işlem yapmanız durumunda diskinizdeki verileri kaybedebilirsiniz. Aşağıdaki kısımda iki ayrı diskiniz olduğu ve bir diskinizi FreeBSD kurulumu için kullandığınızı var sayıyoruz. Bu bölüm ayrıca tek bir disk üzerine FreeBSD'yi tek sistem olarak kurmanız durumunda da işinizi görecektir.

Resim 1'de sistemde iki adet disk bulunmaktadır; ad0 ve ad2. Disklerinizin neden bu şekilde isimlendirildiğini açıklayalım.

ad0: Sisteminizde bulunan ve birinci ide yuvasına takılı olan master diskinizdir. ATA diskler ad* olarak adlandırılır.

da0: ad0 olarak adlandırılan disk ile aynıdır ama SCSI diskler bu şekilde adlandrılır.

ad1: Sisteminizdeki ikinci ATA diskinizdir. Birinci IDE yuvasındaki slave diskiniz olabileceği gibi ikinci ide yuvasındaki master diskiniz de olabilir.

da1: Sistemde yer alan ikinci SCSI disktir.

Sisteminizde birden fazla disk bulunabilir. Bu disklerin yer aldığı yuva veya karta göre 'ad' veya 'da' olarak adlandırlırlar. FreeBSD'de diskler sıfırdan başlanarak numaralandırılırlar. Resimde gördüğünüz gibi bir diski

Resim 1 - Disk ayarları

seçmek için ok tuşları ile aşağı yukarı hareket ederek boşluk tuşu ile seçim yapılmalı ve enter ile sonraki aşamaya geçilmelidir. Burada şu soru akla gelebilir. Bir tek disk yerine birden fazla diski kullanabilir miyim? Bunun yanıtı evet olacaktır. FreeBSD'yi birden fazla diske kurabilirsiniz. Ancak tüm dizinler / altında yer aldığından sadece tek bir disk gibi görünecektir.

Bir diski seçtikten sonra kuruluma başladığınızda disk bölümleri düzenleyicisi karşınıza gelecektir. Seçtiğiniz diskteki bölümlemeyi tamamladığınızda ise önceki ekrana geri dönersiniz. Bu aşamada disk bölümlerinin düzenlenmesi işlemi için FDISK Partition Editor'ün nasıl kullanılacağına ilişkin bilgi vermek gerekir.

```
Partitioning the Disk(s)
Here is what the Partition Editor looks like:
Disk name:      ad0                 FDISK Partition Editor
DISK Geometry:  29795 cyls/16 heads/63 sectors = 234436482
blocks (113664MB)
```

```
Offset   Size(ST)      End  Name  PType   Desc  Subtype  Flags
      0         63       62     -     12   unused      0
     63   30033297 30033359 ad0s1      8  freebsd    165



The following commands are supported (in upper or lower case):

A = Use Entire Disk   G = set Drive Geometry   C = Create
Slice   F = 'DD' mode
D = Delete Slice      Z = Toggle Size Units    S = Set
Bootable   | = Wizard m.
T = Change Type       U = Undo All Changes     W = Write
Changes


Use F1 or ? to get more help, arrow keys to select.
```

Yukarıda disk bölümlerinizi düzenleme aşamasına geldiğinizde karşılaşacağınız ekran görüntüsü az çok yukarıda gördüğünüz gibi olacaktır. Üzerinde işlem yaptığım disk ad0, yani sistemde yer alan birinci ATA disk. Altındaki bölüm ise disk hakkında bilgi veriyor. Diske ait olan silindir, kafa ve sektör sayıları verilmiş. Sonrakiler ise disk üzerinde bulunan disk bölümleri hakkında bilgi vermektedir. Eğer boş bir disk üzerine kurulum yapmış olsaydınız tüm disk "unused" yani boş olarak gösterilecektir. Burada yukarıda gördüğümüz kısımlar hakkında bilgi verelim.

Offset: Oluşturulan disk bölümünün başlangıç sektörü

Size (ST): Bölümün sektör cinsinden boyutu

End: Bölümdeki son sektör.

Name: Sözkonusu bölüm için FreeBSD tarafından kullanılan cihaz adı.

PType: Disk bölümü tanımlayan bir sayı. Partition type kısaltmasıdır.

Desc: Bölümün tanımı

Subtype: Bölüm hakkında ayrıntılı bilgi bu kısımda verilir.

Flags: Bu bölümde ise =,>,R,B,C ve A simgeleri yer alır. =, bölümün düzgün olarak oluşturulduğunu belirtir. >, oluşturulmuş olan bölümün disk üzerindeki 1024'cü silindiri aşmakta olduğunu belirtir. Bu FreeBSD için bir sorun olmaz iken bazı BIOS'ların kısıtlamalarından dolayı problemlere neden olabilir. R, root dosya sisteminin bulunduğu disk bölümünü gösterir.B, BAD144'ü yani bozuk sektöre işaret eder. A, Sistemin açılış yaparken kullanacağı aktif disk bölümünü tanımlar.

Eğer diskinize ait olan bilgileri KB, MG veya GB olarak görmek isterzeniz Z tuşuna basarak yapabilirsiniz. Disk bölümleme işlemi sırasında ekranın alt kısmında yer alan seçenekler çeşitli bölümlendirme seçeneklerini göstermektedir. Eğer tüm diski FreeBSD kurulumu için kullanacak iseniz A tuşun basın. A bastığınızda ekranda disk bölümü tipi olarak 165 görebilirsiniz. Ayrıca yukarıda gördüğünüz gibi, Ayrıca diskin tamamını FreeBSD kurmak için ayırmış olmanıza rağmen halen unused yazdığını görebilirsiniz. Bu durum normaldir. İşlemi tamamlamak için Q tuşuna basıp çıkış yapmanız gerekecektir. W ile işlemi diskinize yazmanıza gerek yoktur çünkü daha diskleri etiketlemediniz.

Sonraki adım olarak size FreeBSD açlış yöneticisini kurmak isteyip istemediğiniz sorulacaktır. Eğer sisteminizde birden fazla işletim sistemi bulunuyorsa kullanmanız uygun olur ancak tek sistem için gerekli değildir. Bu aşamayı geçtikten sonra FreeBSD disk label editor ile disk bölümlerinizi etiketlemeniz gerekecektir. Eğer tek bir disk üzerinde kurulum yapıyorsanız doğrudan disk label editör karşınıza gelir. Ancak birden fazla diski kullanarak bir kurulum gerçekleştirdiyseniz bu durumda etiketleme yapacağınız diski seçip etiketleme işlemine geçmeniz gerekir. Burada birden fazla disk üzerinde kurulum konusunda biraz durmakta yarar var. FreeBSD tek bir disk üzerinde kurabileceğiniz gibi bazı dizinleri farklı disklere dağıtarak da kullanabilirsiniz. Örneğin tek bir takas yerine birden çok diske takas bölümleri atayabileceğiniz gibi /var ve /usr için ayrı diskleri kullanabilirsiniz. Yukarıda anlatıldığı gibi farklı diskleri seçerek yukarıda anlatıldığı gibi işlemleri yaptıktan sonra önce disk seçerek etiketleme işlemine geçebilirsiniz.

Disk Label Editor

FreeBSD için disk bölümleri -partition- slice olarak adlandırılır. Kurulum yapacağınız disk bölümleri üzerinde dosya sistemi oluşturacağımız için bölümleri de kendi içinde yeniden bölümlemek gerekecektir. Bu bölümler

BSD partitions - BSD bölümleri- olarak anılır ve sistemde belli yerlere bağlanır. İlk defa disk label editor çalıştırdığınızda aşağıdakine benzer bir ekran ile karşılaşacaksınız.

```
                FreeBSD Disklabel Editor

Disk: ad0  Partition name: ad0s1  Free: 234436482 blocks
(113664MB)

Part  Mount    Size Newfs   Part      Mount      Size Newfs
----  -----    ---- -----   ----      -----      ---- -----


The following commands are valid here (upper or lower case):
C = Create        D = Delete   M = Mount pt.           W =
Write
N = Newfs Opts    Q = Finish   S = Toggle SoftUpdates   Z =
Custom Newfs
T = Toggle Newfs  U = Undo     A = Auto Defaults        R =
Delete+Merge

Use F1 or ? to get more help, arrow keys to select.
```

Yukarıdaki ekranda kurulum için seçtiğiniz disk veya diskler görüntülenecektir. FreeBSD 5.0'dan bu yana "Auto Defaults" seçeneği hemen hemen her türlü sistem için kullanılabilen bir otomatik disk bölümleme işlemi gerçekleştirmektedir. Debian kullanıcıları için ayrı /usr, /var ve /home bölümü sunan otomatik bölümleme ile hemen hemen aynıdır. Eğer tek tek disk bölümlerini ayırmak ve etiketlemek ile uğraşmak istemiyorsanız A tuşuna basarak varsayılan bölümleme seçeneğinin otomatik olarak diskinize uyarlanmasını seçebilirsiniz. Etiketleme ve bölümleme işlemi bitince Q tuşuna basarak sonraki basamağa geçebilirsiniz.

Elle Bölümleme

Eğer otomatik bölümleme yerine kendiniz bu işlemi yapacak iseniz bu durumda en az iki bölüme gereksinim olacaktır; birisi takas diğeri de kök (/) bölümü. Bu şekilde yapılacak bölümleme ilk bakışta uygun gelebilir. Zira /var, /usr, /tmp vb. ayrı disk bölümleri olarak kullanabilirsiniz. Ancak bir bölümün kapasitesi dolmak üzere iken halen diskinizde gereğinden fazla boş alanınız olabilir ve bu alanı kullanamayabilirsiniz. Bunun önüne geçmek için bir tane (/) bölümü yeterli olur. Disk alanı açısından sorun olmaz. Kurulumu bu şekilde yapmanızı öneririz. Hatta bunu standart olarak sunan bir çok Linux dağıtımı bulurken, BSD ailesi bu konuda net tavır sergiler: "kendin ettin kendin buldun."

Bunu yapmanız olanaklı olsa da BSD kullanıcıları asla bunu yapmaz. FreeBSD ve diğer UNIX türevleri çoklu görevlilik için tasarlanmış güçlü sistemlerdir. Bir ağ üzerinde yer alan sunucuların büyük bir kısmı bu aileden gelmekle birlikte ağır işleri yerine getiren sunucu sistemler olarak işlerini yaparlar. Bu sistemlerde aynı anda binlerce dosya açılır, okunur ve yazılır. Sistem yöneticileri tarafından önlem alınmış olsa da ender olarak bir temizlik görevlisi cep telefonunu şarj etmek istemesi veya elektrikli süpürgeyi çalıştırmak için kazara sistemin fişini çekebilir. Bu tür elektrik kesintileri durumunda o sırada gerçekleşen yazma işlemlerinin kesintiye uğraması durumunda dosya sistemlerinde bozulmalar ortaya çıkabilir. tersine olarak disklere yazma işlemi yapılmadığı bir anda buna benzer durumun yaşanmasında dosya sisteminde bozulmaların ortaya çıkması olasılığı son derece düşüktür.

Bir tek bölüm yerine birden fazla bölüm kullanmak olası bir hata durumunda ortaya çıkabilecek olan zararı azaltırken aynı zamanda (/) dosya sisteminin de zarar görmesini önleyecektir Doğru şekilde yapılandırılmış bir FreeBSD sisteminde (/) üzerinde yazma işlemi yapılmaz.

Bir diğer neden ise bir sunucu üzerinde tek bir (/) bölümü kullanmanın tüm sistemi ağ üzerinden gelebilecek olan tehlikeler ile karşı karşıya bırakacak olmasıdır. Genellikle sistem üzerinde kullanıcıların kullanabilecekleri alan sınırlanmamış ise bir kullanıcının yanlışlıkla veya bilerek bazı dosyaları sitem üzerine kopyalaması sonucunda diskte boş alan kalmayabilir. Posta sunucusu gelen ve giden mesajları kuyruğa alamayacaktır zira diskte boş alan kalmayacaktır. Yazıcı sunucusu yazdırma işlerini kuyruğa alamayacağı için yazıcıdan çıktı alınamayacaktır. Web sunucusu log dosyasına yazamdığı için çalışmayacaktır. Syslogd log dosyasına yazamayacağı için kayıt işlemi gerçekleşemeyecektir. Kopyalanan dosyaların diski dolduması ile /tmp dizini aşırı şişecek ve diskte boş alan kalmayacaktır. Bu ve benzeri bir çok durum kısaca bir Denial of Service-Dos ile sonuçlanacaktır.

Disk bölümlerinin hepsinin birden dolması olasığı son derece düşüktür.

Ancak gene de birden fazla disk bölümü kullanmak bu düşük olasılığı sıfır yapacaktır. Genel olarak FreeBSD kurulumu için beş adet bölüm kullanılması yeterlidir: (/), takas-swap, (/var), (/usr) ve tüm kullanıcılar tarafından kullanılabilen (/tmp) Bazı kaynaklarda /var ile /usr aynı disk üzerinde tutmanızda sakınca olmayacağı belirtilebilir. Ancak kullanıcı dizinlerini /usr/home altında olarak /usr altında tutuyorsanız bu ileride sorun yaratabilir. Bu nedenle /usr ve /var bölümlerini aynı diskte tutacaksanız, mümkünse /usr/home ayrı bir diske taşımak yerinde olacaktır.

Ancak disk bölümlemesini yapacak olan kişi siz olduğunuza göre yol gösterici olması açısından hangi bölümlere hangi dizinleri atayacağınıza karar verirken kolaylık sağlaması amacıyla FreeBSD dizinleri hakkında burada biraz bilgi vermek yerinde olacaktır. Bu konu hakkında ilerleyen bölümlerde daha ayrıntılı bilgi verilecektir.

(/) Bildiğimiz root bölümü. Diğer tüm dizinler bu dizin altında yer alacaktır. Her ne kadar diğer bilgisayarlar ve diskler üzerinde olsalar bile. / bölümü için bir bölüm ayırmak yeterlidir. Bu bölümün büyüklüğü gereksinmelerinize göre değişir. Genel olarak 256MB-512MB arası bir büyüklük yeterlidir. FreeBSD için / ayrılabilecek olan en düşük alan 100 MB kadar bile olabilir ancak 118 MB altına inmeniz durumunda sistem sizi uyaracaktır.

(/boot) Çekirdek, modüller ve ilgili diğer dosyalar bu dizin altında yer alacaktır. 1024'cü silindiri aşan bir root bölümü oluşturmak için /boot'u / ile aynı bölüme alabilirsiniz.

(/usr) Kullandığınız tüm araçlar, programlar burada yer alır. /usr kesinlikle kendisine özel bir disk bölümüne kurulmalıdır. Genel olarak diğer bölümlerin gereksinim duymayacağı disk alanı /usr için ayrılır.

(/usr/local) İşletim sisteminin dışında kalan çeşitli üçüncü parti yazılımların -örneğin web suncusu, veritabanı yazılımları gibi- yer aldığı dizindir. Bazı kullanıcılar /usr'dan ayrı olan bir /usr/local bölümü kullanmayı tercih eder. Bu uygulamayı elinizde birden fazla disk varsa ve /usr bölmek gibi bir niyetiniz varsa yapmanız uygun olur ama tercih sizin.

(/var) Sistemin değişken-variable- dosyaları tuttuğu dizindir. Programlar tarafından oluşturulan/yazılan dosyalar, log dosyaları vb. bu dizin altında yer alır. Genel olarak /var ayrı bir disk bölümü üzerinde tutmayı tercih ederim. Sistem kayıtlatı, yazıcı kuyruğu eğer sistemde varsa posta vb. sunucuların kuyrukları için /var kullanacaktır. Eğer bir sunucu kurulumu gerçekleştirecekseniz /var bölümünü 1 GB veya üzeri olarak düşünmeniz yerinde olur. Hatta posta sunucusu kurmayı düşünüyorsanız bu durumda /var için birkaç GB'lık alan ayırmanız yerinde olur.

(/tmp) Programların ve kullanıcıların geçici dosyalarını yazdıkları dizindir. Sistemi yeniden başlattığınızda içeriği silinecektir. Eğer programlarınız veya kullanıcılarınız büyük boyutlu geçici dosyalar oluşturuyor ve kullanıyorlar ise bu durumda /tmp kullanmamalıdırlar. Onun yerine /var/tmp veya /usr/tmp kullanmaları daha uygun olur. Sysinstall /tmp'i kendine ait bir bölüme kursa da /usr/tmp'i sembolik bağ ile kullanabilirsiniz. Bu büyük boyutlu dosya probleminizi aşmanıza yardımcı olur.

(/home) Kullanıcıların ev dizinleri bu dizinde yer alır. Genellikle /usr altında bulunur. Sisteminizdeki kullanıcı sayısı fazla ise ve kendi ev dizinlerinde çok sayıda dosya bulundurmak gibi bir alışkanlıkları olabileceğini düşünerek /home halen /usr/home altında bağlayabileceğiniz ayrı bir diskteki bölüme almanız yerinde olur. Bu şekilde tüm diski /home olarak kullanabilirsiniz.

Root Bölümü

Eğer ayrı bir /boot bölümü kullanmayacak iseniz diskinizdeki ilk bölüm (/) olmalıdır. Disk Labeleditor ile aşağı yukarı hareket ederek FreeBSD olan bölümü seçip / bölümünü oluşturmak için C tuşuna basın. Size bölümün boyutunu soran bir diayalog kutusu açılacaktır. Buraya 256MB bölüm oluşturmak için 256M yazabilirsiniz. Tercihinize göre bunu büyütebilirsiniz. Alanı tanımladıktan sonra bölümün tipini -partition type- sorusuna root olacağı için FSA File System seçeneğini seçerek onaylayın. Sonraki aşamada bağlanacağı dizin olarak / belirtin. İşlem bittiğinde aşağıdakine benzer bir ekran göreceksiniz.

```
                    FreeBSD Disklabel Editor

Disk: ad0       Partition name: ad0s1   Free: 29761379 blocks
(113408MB)

Part    Mount    Size Newfs   Part      Mount       Size Newfs
```

```
----    -----    ---- -----   ----      -----       ---- -----
ad0s1a    /      256MB UFS2     Y



The following commands are valid here (upper or lower case):
C = Create         D = Delete    M = Mount pt.             W =
Write
N = Newfs Opts     Q = Finish    S = Toggle SoftUpdates    Z =
Custom Newfs
T = Toggle Newfs   U = Undo      A = Auto Defaults         R =
Delete+Merge

Use F1 or ? to get more help, arrow keys to select.
```

Yukarıdaki görüntün anlamı ad0s1a bir dosya sistemi oluşturdunuz ve root olarak etiketleyip / altına bağladınız. Boyutu 256MB ve dosya sistemi UFS2. Artık takas bölümünü oluşturabiliriz.

Swap - Takas Bölümü

FreeBSD sanal bellek kullanan bir işletim sistemidir. Sistemde var olan fiziksel bellekten -RAM- fazlasını kullanabilir. Bunu da programların kullandığı bellek sayfalarını -memory pages- geçici olarak diske yazarak ve gerektiğinde fiziksel belleğe aktararak yapar. Eğer ilgili sayfaya gerek duyulmuyorsa, bu sayfayı diske kopyalarken gereksinim duyulan bir diğer sayfayı fiziksel belleğe taşır. Ancak takas fiziksel belleğinizin yerine kullanabileceğiniz bir bileşen değildir. Sabit diskinizdeki dosyalara erişmek RAM ile karşılaştırıldığında çok yavaş olmaktadır. "Eğer yavaş oluyorsa neden takas kullanıyoruz?" Bu soru sıklıkla sorulur. Takas UNIX sistemler için olmazsa olmaz bir bileşendir. Zira yukarıda açıkladığım gibi fiziksel belleğin yetersiz kaldığı durumlarda takas ek bellek olarak kullanılır. Takasın yapılandırması sistemin genel olarak performansına etki eder. Bu nedenle takas bölümü oluştururken işinize kolaylaştıracak bazı öneriler:

* Takas bölümüzünü olabildiğince diskin başlangıç bölümüne yakın oluşturmaya çalışın. silindir numarası küçük olan bölümler kenara daha yakındır. Yüksek numaralı silindirlere göre daha hızlı erişilir ve okunur. Ortadakilerden daha hızlıdırlar. Açısal hızları aynı olsa da çizgisel hızları daha yüksektir.

* Eğer sisteminizde birden fazla disk kullanıyorsanız, takas bölümünüzü en hızlı diskinize atayın. Öte yandan en hızlı diskiniz aynı zamanda sunucu uygulamaları için de en uygun disk olacağı için takas bölümünüzü en az kullanılacak diske atayın. Bu şekilde takasa erişme zamanınız kısalacaktır. Zira bu disk üzerindeki okuyucu kafalar zaten takas bölümü üzerinde konumlanmış olacağı için okuma zamanınız kısalacaktır. Bu özellikle de takas işlemlerinin yoğun olarak gerçekleştiği sistemlerde önemli performans kazanımları sağlayacaktır.

Takas alanı ne kadar olmalı? Genel kural olarak takas alanı fiziksel belleğin iki buçuk katı olarak ayrılır. Bu kural fiziksel belleklerin pahalı olduğu ve kullanıcıların 4 - 16 MB RAM ile çalışmak durumunda kaldıkları zamandan kalmadır. Güncel sistemler ise 1 GB ile 4GB arasında değişen miktarlarda RAM ile sunulmaktadır. Bazı iş istasyonu ve sunucu sistemlerinde bu miktar daha da artmaktadır. Bununla birlikte depolama aygıtlarının da fiyatları ucuzladığı için kolaylıkla yüksek kapasitelere ulaşabiliyoruz. Sisteminize RAM ekleyerek fiziksel bellek miktarını arttırmak kolay olduğundan takas alanının dolması gibi bir durum ile karşı karşıya kalmanız söz konusu olmayacaktır. Ancak yine de adet yerini bulsun diyerek fiziksel belleğinizin iki katını takas olarak ayırıp kullanabilirsiniz. Takas alanını büyütmek durumunda kalmadan eski takası kullanabilirsiniz.

Kendi kullandığım sistemlerde takas halen 512MB. Bu alanın dolması söz konusu olmadı. Ancak gene de fiziksel belleğiniz kadar bir takas alanı oluşturmayı tercih edebilirsiniz. Eğer geliştirme amaçlı veya FreeBSD CURRENT kullanmayı düşünüyorsanız takası fiziksel bellek ile eşit miktarda tutun. Bunu gerçekten ihtiyacınız olacaktır.

Seyrek olmakla birlikte, kernel bazen beklenemdik bir durum ile karşılaşır. Bu durumlarda kernel panic hatası mesajını görebilirsiniz. Kernel panic hatasının ardından sistem yeniden başlatılır. Eğer kernel panic hatası alındığında RAM içeriğinin olduğu gibi takasa yazılacak şekilde sistemi yapılandırdıysanız takas bölümüne tüm RAM içeriği yazılır ve sistem bu işlemden sonra yeniden başlatılır. RAM içeriği geliştiriciler için önemli bir bilgidir. Bu bilgi ile hatanın kaynağını belirleyebilirler. Bu durumda takas RAM'dan daha küçük boyutlu ise bilgi edinmek olanaklı olmayacaktır.

Eğer kernel panic, çekirdekte dosya sistemleri ile ilgili bir sorundan kaynaklı ise takas dosyasına RAM içeriğini yazacaktır. Ancak burada takas alanını aşması söz konusu olabilir. Bu durumda takas alanından sonra gelen dosya sistemine de yazacağı için giderilmesi güç sorunlar ortaya çıkabilir.

Kernel panic hataları FreeBSD'de ender olarak görülür. 7 yıllık kullanım deneyimde sadece bir defa karşılaştım o da benim hatamdı. Ancak gene de takas alanını en az fiziksel bellek kadar tutmak yerinde olacaktır. Bu şekilde RAM içeriği kaybolmadan takasa yazılabilir ve dosya sisteminde sorun olmaz.

Takas alanının boyutuna karar verdiğinizde takas bölümünü yukarıda anlattığımız gibi oluşturabilirsiniz. Bu sefer dosya tipi olarak swap seçmek yeterli olacaktır. Bağlanma noktasına gerek yoktur zira takas bölümlerini bir dizin altına bağlamanıza gerek yoktur. FreeBSD gerekeni yapar.

Diğer Bölümleri Oluşturmak

/root (ve /boot da dahil) ile takas bölümlerini oluşturduktan sonra diğer bölümleri boyutlarına karar verip diğerlerini oluşturduğunuz gibi oluşturabilirsiniz. Her bölüm için bağlanacağı alanı tanımlamanız, örneğin /usr, /var gibi, başına mutlaka / koymanız gerekir. Eğer bölümlerden birisini değiştirmek isterseniz ilgili bölümüme ok tuşları ile gelip D tuşuna basarak silebilir ve sonra da yeniden C tuşuna basarak oluşturabilirsiniz. Birden fazla diskiniz varsa bu diskleri seçerek ilgili bölümleri oluşturabilir ve düzenleyebilirsiniz. Sadece seçtiğiniz diskteki bölümleri görebileceğinizi unutmayın. İşlemler sona erdiğinde aşağıdakina benzer bir ekran göreceksiniz:

```
                    FreeBSD Disklabel Editor

Disk: ad0       Partition name: ad0s1   Free: 0 blocks (0MB)
Disk: ad1       Partition name: ad1s1   Free: 0 blocks (0MB)

Part    Mount      Size Newfs   Part  Mount       Size Newfs
----    -----      ---- -----   ----  -----       ---- -----
ad0s1a  /         256MB UFS    Y
ad0s1b  swap      756MB SWAP
ad0s1e  /var      256MB UFS2+S Y
ad0s1f  /tmp      256MB UFS2+S Y
ad0s1g  /usr    48817MB UFS2+S Y
ad1s1e  /home  124664MB UFS2+S Y


The following commands are valid here (upper or lower case):
C = Create         D = Delete    M = Mount pt.             W =
Write
N = Newfs Opts     Q = Finish    S = Toggle SoftUpdates    Z =
Custom Newfs
T = Toggle Newfs   U = Undo      A = Auto Defaults         R =
Delete+Merge

Use F1 or ? to get more help, arrow keys to select.
```

Yukarıdaki örnekte iki disk görülmektedir. /home dizini ikinci diskte yer almakta ve diskin tamamını kullanmaktadır. root bölümü ve takas hariç diğer bölümlerde softupdates etkin bırakılmıştır. Önceki yazıda da değindiğimiz gibi FreeBSD 5.0'dan bu yana / hariç softupdates standart olarak kullanılmaktadır. Softupdates kendi başına bir inceleme konusu olduğu için bunu başka bir yazıda ele alacağız. Softupdates'i etkin kılmak sistem performansınıza önemli etkileri olacaktır. Bir dosya sisteminde softupdates etkin kılınmış ise dosya tipinin yanında UFS2+S olarak gösterilir.

Bölümleri oluşturduktan sonra Q tuşuna basarak işlemi sona erdirin. Burada bir defa daha anımsatmakta yarar var. W tuşuna basmayın. W sadece var olan dosya sistemleri üzerinde işlem yapmak için kullanılır. Bu aşamaya kadar diskte bir işlem yapmadığımız için W kullanamayız. Sysinstall ile çalıştığınız sürece bütün işlemleriniz bellekte saklanır ve kurulmasını istediğiniz yazılım bileşenleri seçilmeden dikinizde dosya sistemi oluşturulmaz.

[1]http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/using-sysinstall.html

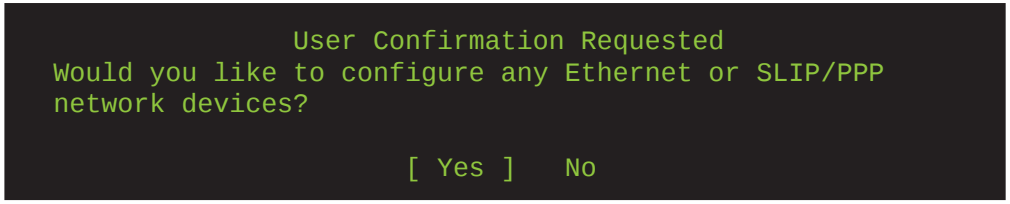
Gökşin Akdeniz
<goksin@enixma.org>

Kurulum Sonrası Yapılandırma

Dosyaların kopyalanması işlemi sona erdiğinde sistemin kurulum sonrası yapılandırma basamağına gelmiş oluyoruz. Bu aşamada geri dönerek Sysinstall ile önceden belirlemiş olduğunuz seçenekler üzerinde değişiklik yapabilirsiniz. Bir tercih değişikliğine gerek yoksa sistemi yeniden başlatmadan önce son ayarlarınızı yaparak kurulumu bitirebilirsiniz. Kurulum aşamasının son kısmında ağ, fare, zaman dilimi, masaüstü, yazılım, kullanıcı ekleme ile sistem yöneticisi-root- parolası ataması yapılarak kurulum tamamlanmaktadır.

Ağ Yapılandırması

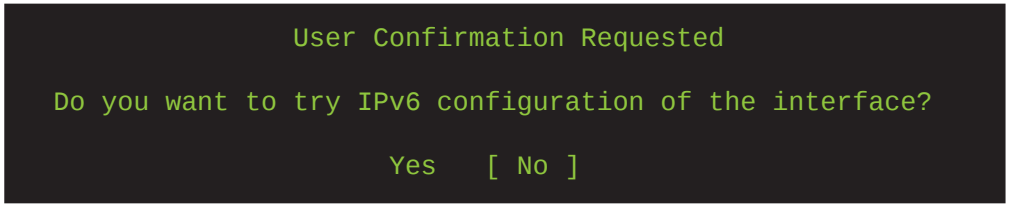
Sysinstall ekranına geri geldiğimizde Configure'u seçiyoruz ve 'enter'e basıyoruz. İlk olarak ağ yapılandırmasını yapmamız gerekecektir. Ağ kartınızın yapılandırması için aşağıdaki soruya 'Yes'i seçerek devam ediyoruz.



Eğer kurulum yaptığınız sistemin ağ yapılandırması ile ilgili bilgilere sahip değilseniz -örneğin iş yerindeki bilgisayarınıza kurulum yapıyorsanız ve ağ yapılandırması ile ilgili bilginiz yoksa bu durumda ilgili kişilerden bu bilgileri edinmeniz gerekecektir. Ağ kartı yapılandırmasını bu aşamada yapmanız zorunlu değildir. Sonrada Sysinstall ile bu işlemi yapabilirsiniz.

Linux kullanıcısı iseniz ağ donanımı isimlendirmesinin FreeBSD'de farklı olmasından dolayı sorun yaşayabilirsiniz. FreeBSD için ağ donanımları, kullandıkları yonga setleri ile tanımlıdır. Örneğin Nforce yonga setli anakarttaki ağ kartınız nf0 olarak tanımlanır. Sysinstall sisteminizde bulunan ağ donanımını listeleyerek sizden yapılandırmak istediğiniz donanımı seçemenizi isteyecektir.

Ağ kartınızı listeden seçip yapılandırmaya başladığınızda size IPv6 desteğini kullanmak isteyip istemediğiniz sorulacaktır. Eğer ağınız IPv6 yapılandırmasına sahipse kolaylıkla IPv6 yapılandırmasını kullanabilirsiniz. Değilse IPv6 desteğini kullanmayın.



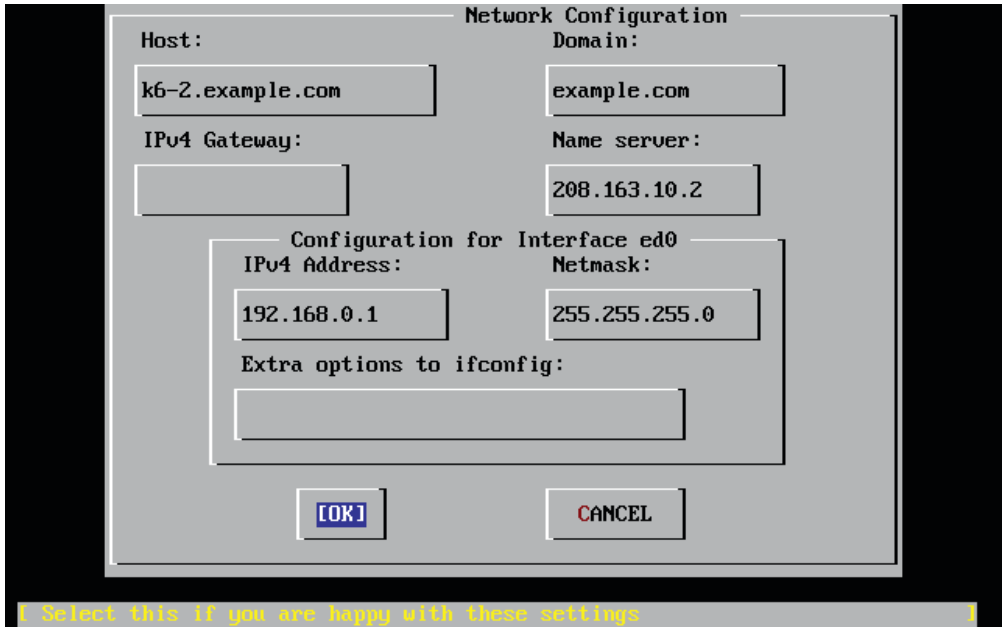
Sonraki aşama ise DHCP yapılandırması kullanıp kullanmadığınız olacaktır. Eğer ağınızda bir DHCP sunucus varsa bu seçeneği kullanabilirsiniz. Aksi halde statik ip yapılandırması gerekecektir.

```
                User Confirmation Requested

    Do you want to try DHCP configuration of the interface?

                          Yes    [ No ]
```

Statik yapılandırma aşamasında aşağıdaki ekran ile karşılaşacaksınız.



Buradaki alanların işlevleri sırası ile;

Sistem için genel seçenekler:

Host: Bilgisayarınızın ve bulunduğunuz ağın -varsa- alan adı.
Domain: Alan adı.
IPv4 Gateway: Eğer bir router üzerinden internete çıkıyorsanız ağ geçidi ip adresini buraya yazın.
Name server: DNS sunucusunun ip adresini yazın.

Ağ kartınıza özel bilgiler:

IPv4 Address: Sisteminizin ip adresi
Netmask: Ağ kartınıza atadığınız ağ maskesi
Extra Options to ifconfig: ifconfig ile kullanacağınız ek yapılandırma seçenekleri. Bu kısmı boş bırakabilirsiniz.

Bilgileri girmek için TAB tuşuna basarak bölümler arasında geçiş yapabilirsiniz. Gerekli yapılandırmayı tamamladığınızda OK tuşuna basarak ağ kartı yapılandırmanızı tamamlayabilirsiniz. Ağ kartınızın aktifleştirilmesi sorusuna ise

```
                User Confirmation Requested
    Would you like to Bring Up the ed0 interface right now?

                          [ Yes ]   No
```

No seçerek yanıt verebilirsiniz. Yes seçmeniz durumunda ağ kartı tanımladığınız yapılandırma seçenekleri ile aktif hale gelecektir. Ancak sistemi yeniden başlatmadan bu işlemin pratik bir yararı olmayacaktır.

Bu aşamadan sonraki adımda ise Sysinstall ağ yapılandırması ile ilgili bazı sorular soracaktır. İlk soru sistemi bir ağ geçidi olarak kullanmak isteyip istemediğiniz olacaktır.

```
                User Confirmation Requested
    Do you want this machine to function as a network gateway?

                          [ Yes ]    No
```

Eğer sisteminizi ağ geçidi olarak kullanmayı planlamadıysanız bu soruya No yanıtını verin. Ağ geçidi olarak yapılandırmayı gelecek bölümlerde okuyabilirsiniz.

Sonraki soru ise inetd ile çalıştırılan hizmetleri kullanıp kullanmayacağınızdır.

```
                User Confirmation Requested

Do you want to configure inetd and the network services that
it provides?

                          Yes    [ No ]
```

inetd, FTP, telnet, POP ve IMAP posta hizmetlerine gelen istekler ile ilgilidir. Bu hizmetlerin yapılandırılması konusunda yeterli bilginiz yoksa varsayılan yanıt olan 'hayır'ı seçerek devam edebilirsiniz.

Sisteme uzaktan erişmek için SSH'yi aktifleştirmek isterseniz bu soruya evet demeniz gerekir. Ancak sisteminize uzaktan erişmek istemiyorsanız bu durumda hayır seçeneğini seçin.

```
                User Confirmation Requested
            Would you like to enable SSH login?
                  Yes        [  No  ]
```

SSH ile FreeBSD uzaktan erişim konusunu ilerleyen bölümlerde yeniden ele alacağız.

Ağ yapılandırması ile ilgili sondan bir önceki soru ise sisteme anonim FTP erişimine izin verip vermeyeceğiniz olacaktır.

```
                User Confirmation Requested
Do you want to have anonymous FTP access to this machine?

                          Yes    [ No ]
```

Varsayılan yanıt hayırdır. Ancak ftp erişimini açmak isterseniz bunu sonradan yapabilirsiniz. Ağ yapılandırması ile ilgili son soru ise sistemi NFS sunucusu olarak kullanmak isteyip istemediğinizdir.

```
                User Confirmation Requested
Do you want to configure this machine as an NFS server?

                          Yes    [ No ]
```

NFS sunucusu, sistemdeki diskte belirlediğiniz dizinleri ağ üzerindeki diğer bilgisayarlar ile paylaşıma açmanıza olanak verir. Ağ üzerindeki diğer bilgisayarlar bu dizini oku/yaz izinleri ile bağlayıp kullanabilir. Bu soruya da hayır diyerek devam ediyoruz. Benzer biçimde NFS sistemcisi olarak bilgisayarınıza uzak sistemlerdeki NFS dizinlerine erişmek isteyip istemediğiniz sorusuna tercihiniz doğrultusunda yanıt verip ağ yapılandırmasını bitiriyoruz.

```
                User Confirmation Requested
Do you want to configure this machine as an NFS client?

                          Yes    [ No ]
```

Konsolun Yapılandırılması

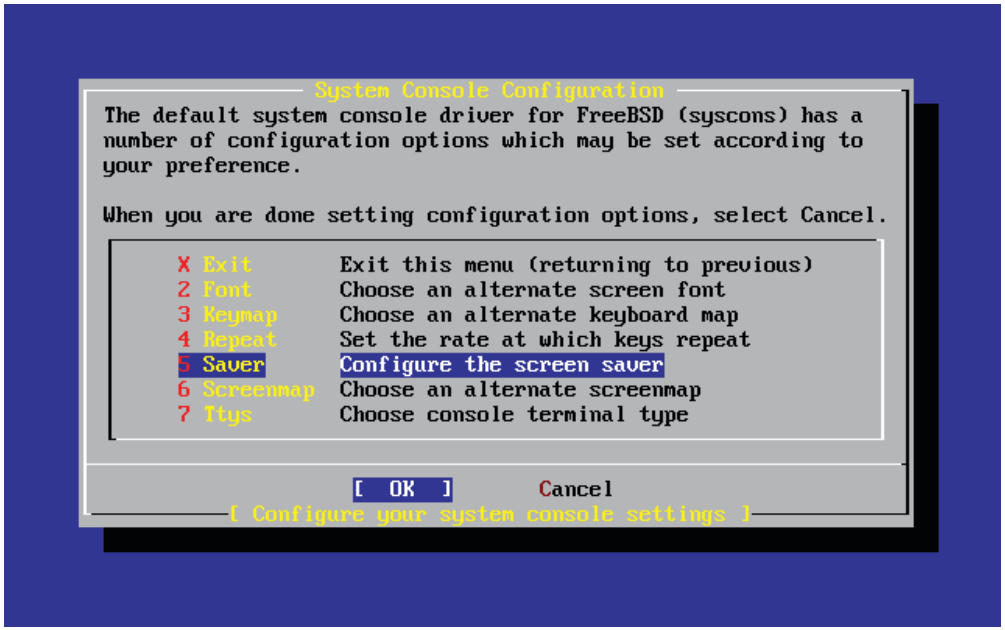
FreeBSD terminali -yani vty*- ayarlarını da düzenleyebilirsiniz. terminal üzerinde kullanılacak olan font, klavyeniz, ekran koruyucusu vb. yapılandırabilirsiniz. Yapılandırmak isterseniz aşağıdaki ekrana evet demeniz yeterlidir.

```
                User Confirmation Requested
Would you like to customize your system console settings?

                          [ Yes ]  No
```

Evet yanıtı verdiğinizde aşağıdaki ekranı görebilirsiniz. (Sonraki sayfada)

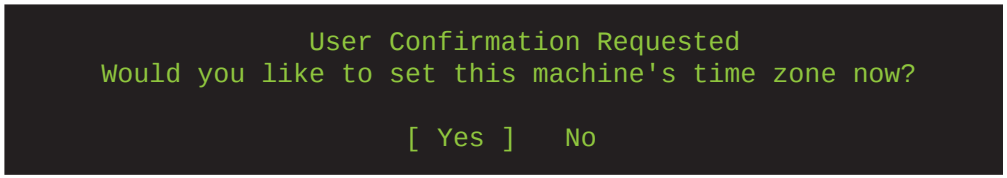
Bu akranda gördüğünüz yapılandırma seçeneklerinin büyük bölümüne ilişkin varsayılan ayarlar üzerinde değişiklik yapmanıza gerek kalmayacaktır. Ancak yine de bazı ayarları kendi tercihlerinize göre yapmak isteyebilirsiniz. keymap seçeneği, klavye düzenini yapılan-



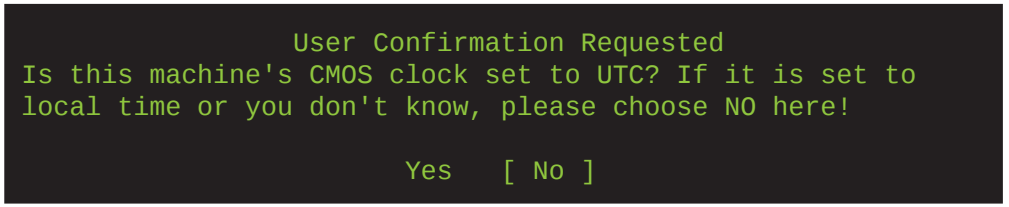
dırır. Varsayılan klavye seçeneği U.S. English-ABD İngilizce olarak tanımlıdır. Bunu isterseniz değiştirebilirsiniz. Screenmap seçeneği ise dil seçiminize uyan karakter setlerini ayarlamanızı sağlar. Eğer kullanmakta olduğunuz donanım eski ise be bu ayarları destek-lemiyorsa bu durumda Ttys-Terminal TYpes seçeneği ile konsolu VGA terminal emülasyonu yerine farklı bir emülasyonu kullanacak biçimde yapılandırabilirsiniz.

Zaman Dilimi Ayarı

Sysinstall zaman dilimi yapılandırması aşamasına geçecektir. Zaman dilimini yapılandırmak için evet diyerek devam ediyoruz.



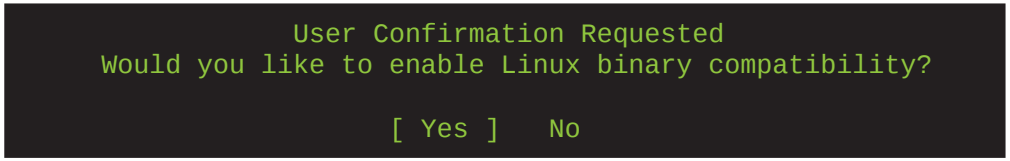
Sonraki adımda Sysinstall donanım saatinin UTC (Coordinated Universal Time, Greenwich Mean Time ya da Zulu Time olarak da anılır) ayarlı olup olmadığını soracaktır. Eğer bundan emin değilseniz Hayır seçerek devam edin.



Karşınıza gelecek olan menüden bulunduğunuz ülkeyi bulup seçin. Seçimizi yaptığınızda Sysinstall size üç harften oluşan bir isim sunacaktır. Bu isim bulunduğunuz zaman dilimini tanımlar. Evet diyerek zaman dilimi yapılandırmasını tamamlayın.

Linux Uyumluluğu

FreeBSD birçok farklı işletim sistemine ait olan uygulamaları çalıştırabilir. Standart kurulumun bir parçası olarak Linux uyumluluğu isteyip istemediğiniz sorulacaktır. Masaüstü sistem olarak kullanacak iseniz Flash'ın bazı MATLAB vb. gibi yazılımların BSD desteğinin olmadığını göz önüne alarak Linux uyumluluğunu aktif kılmanızda yarar vardır.

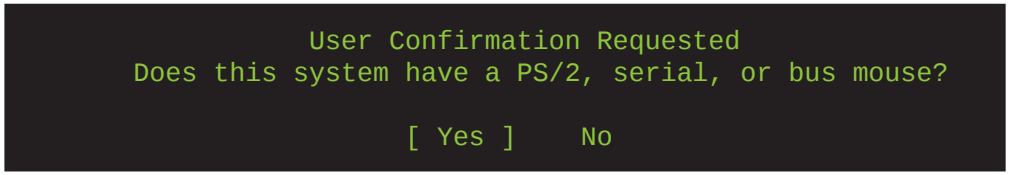


Uyumluluk aslında Linux emülasyonu ile gerçekleşmektedir. Emülatör denildiğinde diğer işletim sistemlerinden deneyiminiz varsa aslında asıl sisteme göre emülasyonların daha yavaş çalıştığını düşünebilirsiniz. freeBSD için ise durum böyle değildir. FreeBSD'deki Linux uyumluluğu bir emülatör yazılımı ile değil, FreeBSD çekirdeğindeki Linux binary uyumluluğu modülü ile sağlandığından bir emülatör

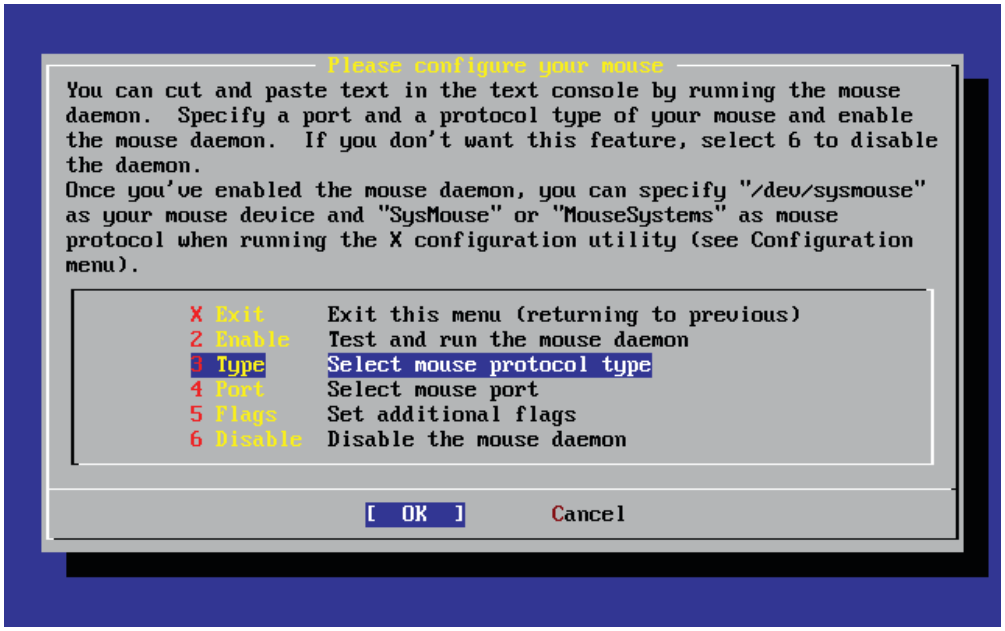
yazılımından daha performanlı olarak çalışmaktadır. Linux uyumluluğu isterseniz Sysinstall /usr dizinine Linux uygulamalarını çalıştırmak için gereken kütüphaneler ile diğer gerekli uygulamaları kuracaktır.

Fareyi Yapılandırmak

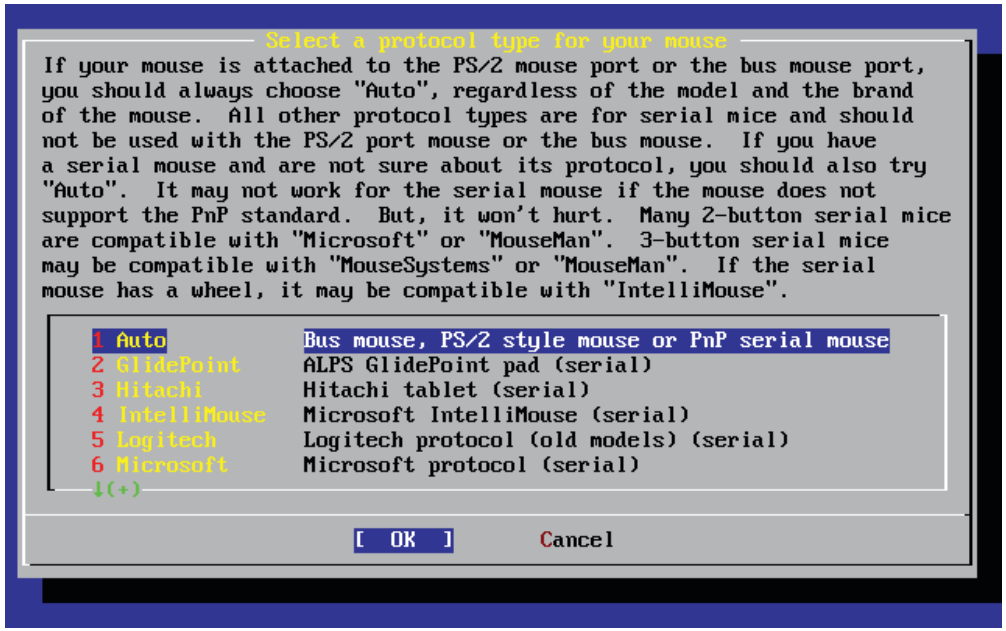
FreeBSD'de farenizi ister konsolda isterseniz grafik ortamda aynı biçimde kullanabilirsiniz. Eğer gerçekleştirdiğiniz kurulumda grafik arabirime gerek olmayacak ise bu durumda fare yapılandırmasını atlayabilirsiniz.



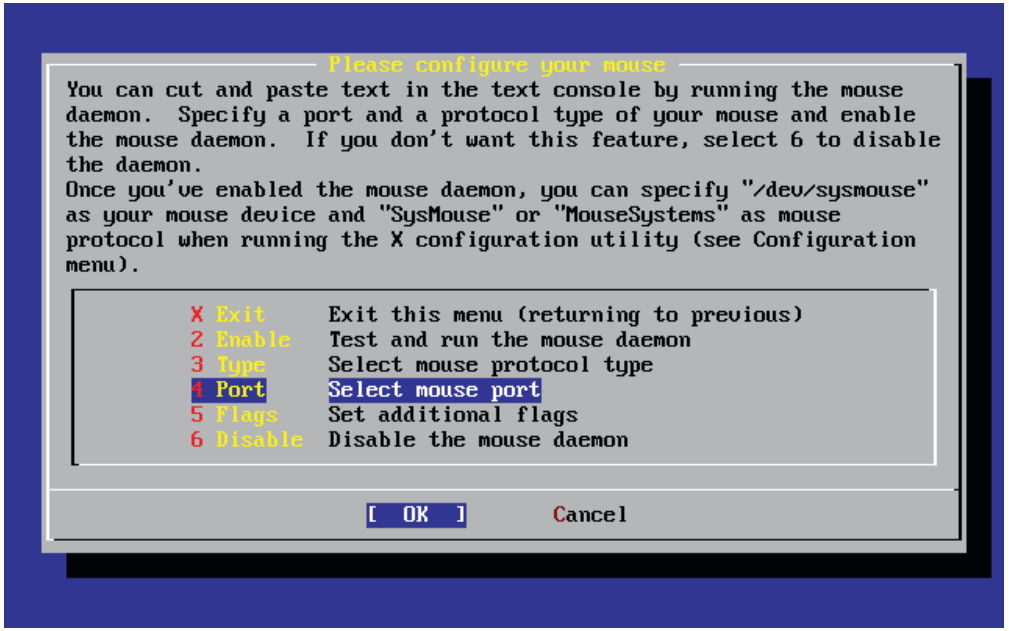
Öte yandan masaüstü sistem olarak kullanacak iseniz fare sürücüsünü yapılandırmanız gerekecektir. Size ilk sorulan soru PS/2, Seri bağlantıya sahip bir fare kullanıp kullanmadığınız olacaktır. Eğer USB bir fare kullanıyorsanız yukarda gördüğünüz soruya "No" diyerek devam edin. (Sol alttaki resim)



Karşınıza gelecek olan ekranda farenizi yapılandırmak için kullanabileceğiniz birçok seçenek bulunmaktadır. Eğer X kullanmak istemiyorsanız 'disable'ı seçip yapılandırmayı sona erdirebilirsiniz. farenizi yapılandıracak iseniz 'Type' seçeneğini kullanarak farenizin tipini seçerek yapılandırma işlemine devam edebilirsiniz.



Eğer bir PS/2 bağlantısı veya USB ile bilgisayarınıza taklı bir fare kullanıyorsanız "Auto" seçeneğini seçin. farenin takılı olduğu portu belirlemek için ise 'port' seçip devam edin. (Sonraki resim)



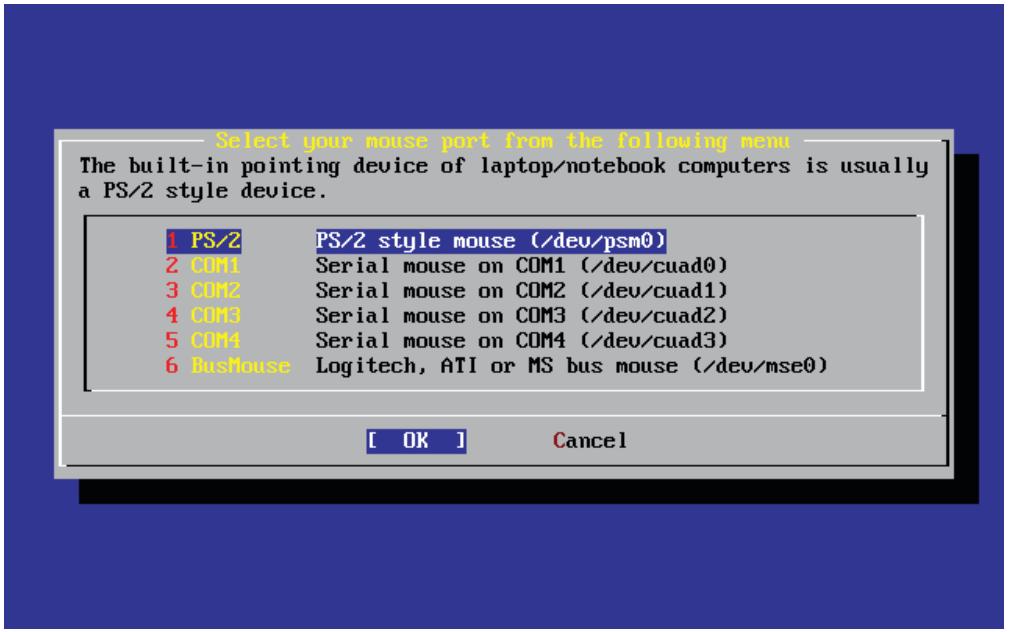
Farenin takılı olduğu port COM1/COM2 gibi bir port ise doğru portu tanımlayıp enter ile devam edin. (Yanda altta)

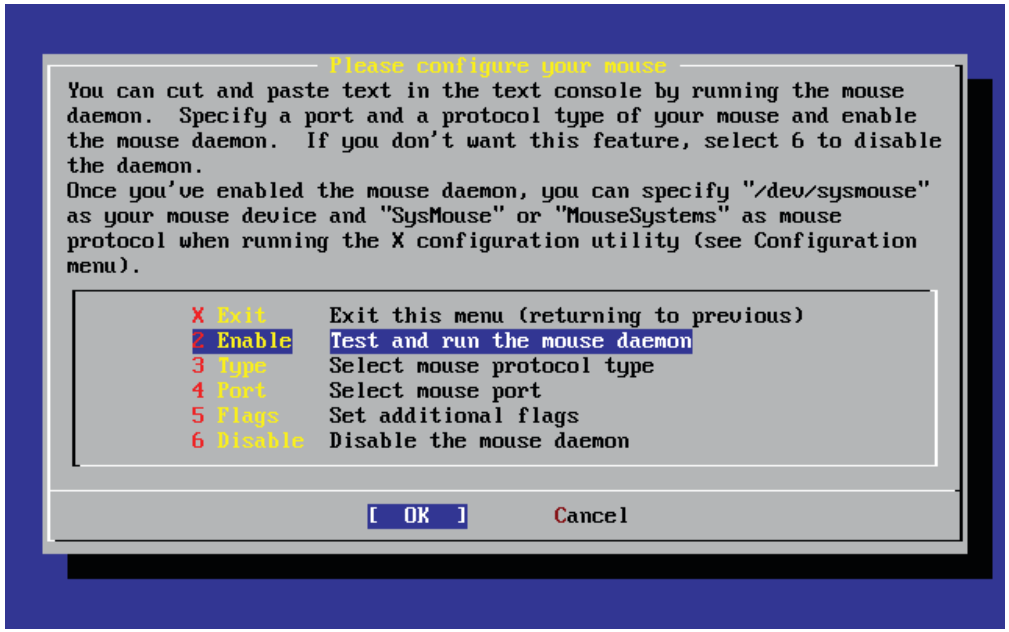
Port tanımlamasını bitirdiğinizde farenin ekrandaki hızı ile üç tuş emülasyonu yapılandırmasını yapamanız uygun olur.

Fare yapılandırmasında 3 tuş emülasyonuna gerek duyuyorsanız Flags seçeneğini kullanmanız gerekir. Flags, fare için üç tuş emülasyonu ve farenizin ekrandaki hızını ayarlayabilirsiniz. Flags ekranında karşınıza gelecek dialog kutusunda üç tuş emülasyonunu kullanmak için '3' değerini yazmanız yeterlidir. Farenin ekrandaki hızını da buradan ayarlayabilirsiniz. Ekranda hızlı hareket etmesi için

```
-r high
```

Aynı şekilde farenin hızını azaltmak için de

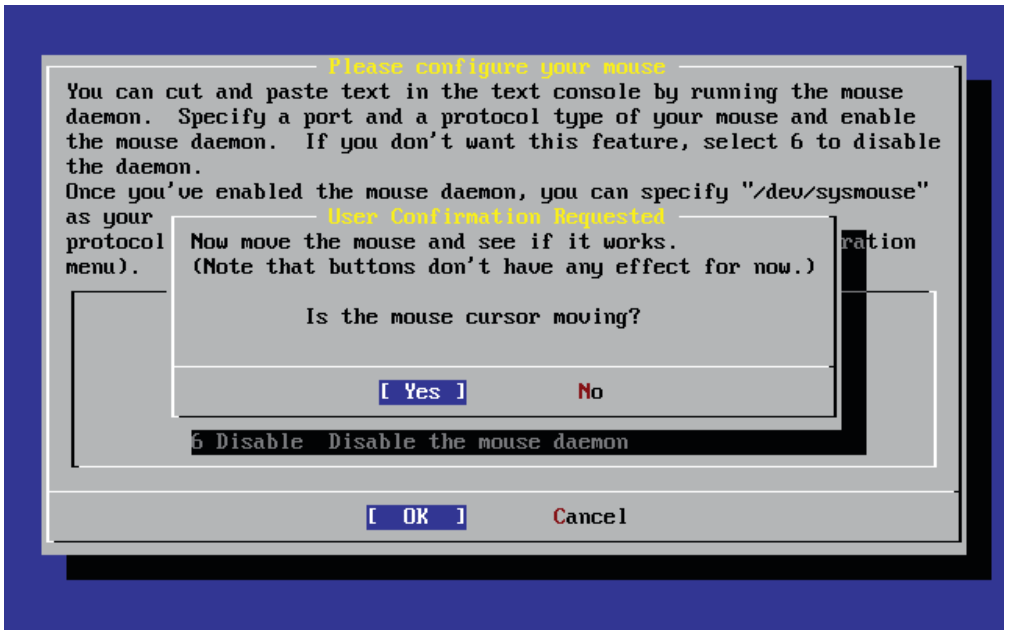




```
-r low
```

seçeneklerini girmeniz gerekir. Yukarıda adı geçen seçenekleri tamamladıktan sonra farenizin çalışmasını kontrol edin. Farenizi aktif hale getirmek için 'Enable' seçeneğini kulanın. (Önceki sayfa sol alt resim)

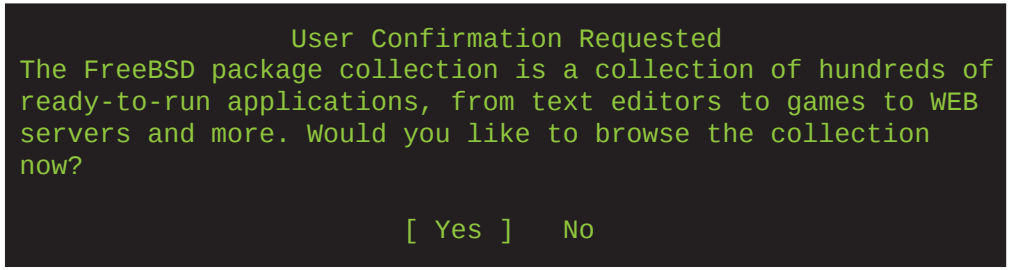
Farenizi hareket ettirmeye başladığınızda fare ekranda da hareket etmeye başlayacaktır. USB farelerin hareket etmesi PS/2, seri/COM portuna takılı farelere göre biraz gecikerek gerçekleşmektedir. Bu ise sadece yapılandırma aşamasında ortaya çıkan bir durumdur. Sistemi yeniden başlattığınızda bir fark görmeniz söz konusu değildir.



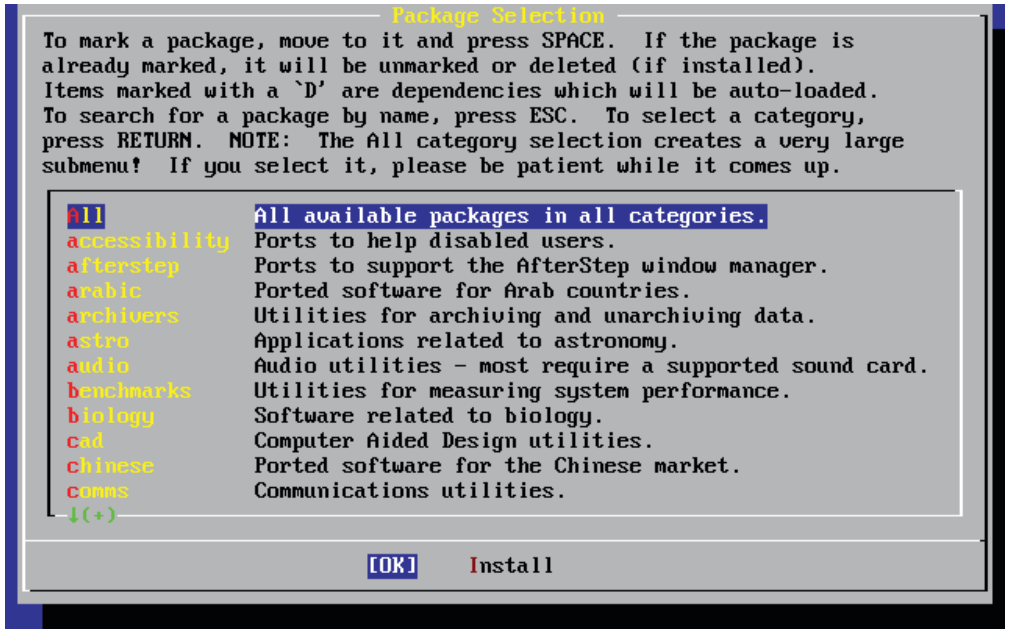
Fare yapılandırması bittiğinde 'Exit' seçeneği ile fare yapılandırma ekranından çıkıp yazılım kurma ekranına geçeriz.

Yazılım Kurulumu

Kurulacak yazılımların seçimini aslında daha önceden yapmış olmanıza rağmen gereksinim duyacağınız yazılımlar standart kurulumun dışında kalabilir. Bu nedenle kurulum işlemi sona erdikten sonra da bu yazılımları kurabileceğiniz gibi kurulum bitmeden önce elinizdeki kurulum kaynağından da bunları kurabilirsiniz. Ek yazılım kurulumu için aşağıdaki ekranda evet tuşuna basmanız yeterli olur.



Sysinstall, tanımlamış olduğunuz kurulum kaynağında yer alan pa-

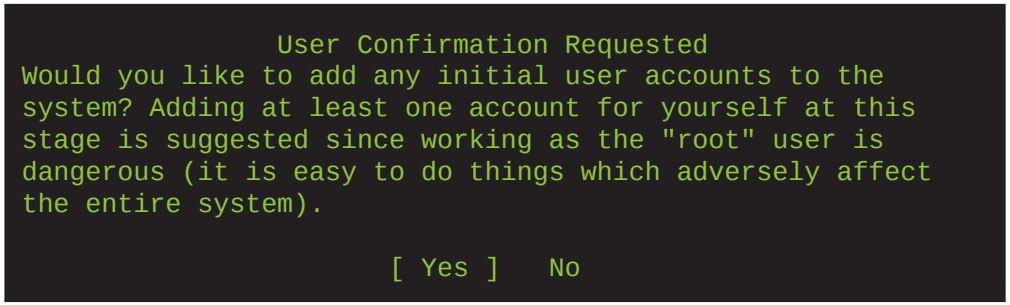


ketleri gruplar halinde listeler. İlgili grupta yer alan paketleri ayrıntılı olarak görüntülemek ve kurmak için bölümü seçip 'enter'e basmanız yeterlidir.

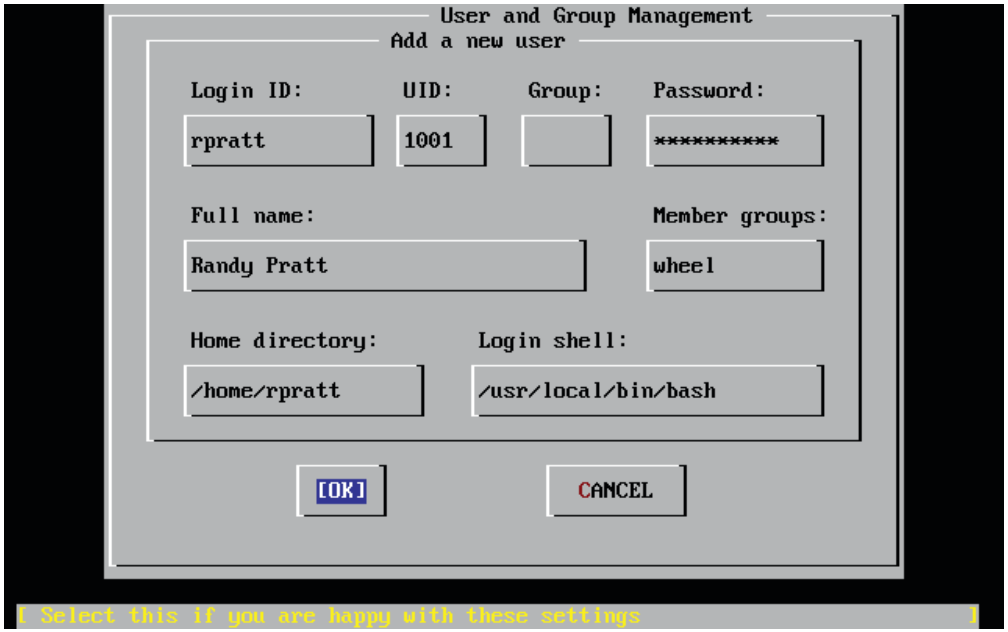
Sysinstall ile dağıtım seçiminde yazılım seçimi yaptığınız için kurulacak olan paketler X ile işaretlenmiş durumdadır. Bir yazılımı seçtiğinizde, kurulumu için gereken bağımlılıklar da işaretlenir. Bağımlılıklar seçim olarak D ile gösterilir ve isterseniz bağımlı paketlerin kurulumamasını seçebilirsiniz. Ama bu tavsiye edilmez. Ekranın alt kısmında o sırada seçtiğiniz paketlerin kısa bir tanımı görüntülenir. Paket seçimini onaylamak için OK'a bastığınızda yeniden paket grupları listesine döneceksiniz. İşleminiz bittiğinde 'Install'-kurulum'u seçip 'enter'e basarak paketlerin kurulumunu gerçekleşetirebilirsiniz.

Kullanıcı Eklemek

Paket kurulumu tamamlandığında, kullanıcıların sisteme eklenmesi aşamasına gelinir. FreeBSD sisteme en az bir normal kullanıcı eklenmesini gerektirir. Zira root bölümü son derece küçük olduğu için root hesabını kullanmanız durumunda disk alanı dolacaktır. Öte yandan sadece root hesabını kullanmak ciddi sorunlara da yol açabilir.



Yeni bir kullanıcı veya grup eklemek için aşağıdaki resimde gördüğünüz diyalog kutusunu kullanmanız gerekir. (Yanda üstte)



Grup eklemek veya çıkarmak için şimdilik bir işlem yapmayın. Sadece

kullanıcı eklemek yeterli olacaktır. Kullanıcı eklemek istediğinizde aşağıdaki diyalog kutusunu kullanmak gerekecektir. (Önceki sayfa sağ alt resim)

Ekran görüntüsünde gördüğünü tüm alanları doldurmanız gerekli değildir. Tersine, bazı alanları boş bırakmak ve bunların sistem tarafından atanmasını sağlamak için yeterlidir. Alanlar arasında geçiş yapmak için TAB kullanmak gerekir. Normal kullanıcı hesabının yapılandırılması sırasında doldurulması gereken alanlar şunlardır:

Login ID

Kullanıcının sisteme giriş yaparken kullanacağı kullanıcı adıdır. Kullanıcı adı -login id- sekiz karakterden uzun olmamalıdır. Bu sınırlama sadece bazı uygulamalar ile ortaya çıkabilecek olan uyum sorununu ortadan kaldırmak için önerilir. FreeBSD için kullanıcı adı 16 karakter uzunluğunda olabilir. Kullanıcı isimleri büyük/küçük harf duyarlıdır. Kullanıcı isiminde (-) ve (_) kullanabilirisiniz, (+),(/) gibi diğer özel karakterler ise kullanılamaz.

UID

UID sistemdeki kullanıcıların tanımlanmasını sağlayan bir sayıdır. Bu alanı sistem kendisi atayacaktır.

Group

Bu alan kullanıcının üyesi olduğu birincil grubu tanımlar. Bu alan da sistem tarafından atanacağı için boş bırakıyoruz.

Password

Bu kısma ise sisteme giriş yaparken kullanacağınız kullanıcı şifresini yazıyoruz. Şifreler de kullanıcı adları gibi büyük/küçük harf duyarlıdır.

Full Name

Kullanıcının tam adı ve soyadı alanıdır. Bu alanı boş bırakabilirsiniz. Standart olarak User & yazılı olarak gelmektedir. Bu alanla ilgili olarak dikkat etmeniz gereken nokta ise UNIX posta yazılımlarının bu alanı kullanmakta olduğudur. Eğer sistemin posta uygulamasını kullanacak iseniz bu alanı doldurmakta yarar vardır.

Member groups

Kullanıcının üyesi olduğu ikincil gruplar burada tanımlanır. FreeBSD için bu grup alanı "wheel"olarak atar. Böylelikle root yetkilerini kullanmak için sisteme root olarak giriş yapmak durumunda kalmazsınız.

Home Directory

Kullanıcının ev dizininin diskte bulunduğu yeri tanımlar. Eğer özel bir yapılandırma söz konusu değilse boş bırakın. Sistem gerekeni yapacaktır.

Login Shell

Sistemi yeniden başlatıp hesabınıza giriş yaptığınızda kullanacağınız kabuk ortamıdır. Eğer BASH seçtiyseniz kabuk ortamızın yolunu /usr/local/bin/bash olarak tanımlamanız gerekir. Başka bir kabuk ortamı da atayabilirsiniz. C kabuğu için /bin/csh TCSH için /bin/tcsh gibi.

Bu bilgileri girdikten sonra Ok'a basarak kullanıcı yapılandırmasını tamamlıyoruz. Çıkış için 'Exit'e basarak devam ediyoruz.

Root Şifresini Atamak

Normal kullanıcı hesabını yapılandırdıktan sonra root kullanıcısı şifresini atamanız gerekir. Karşınıza gelen ekranda root şifresi atamak için enter veya boşluk tuşuna basarak şifre atama işlemini gerçekleştirebilirsiniz.

```
                        Message
Now you must set the system manager's password.
This is the password you'll use to log in as "root".

                        [ OK ]

                  [ Press enter or space ]
```

Şifre atamak işlemi sırasında root şifresi ekranda görünmez. kullanıcı hesaplarındaki gibi her karakter için bir yıldız vb. karakter gösterilmez. Tersine şifre yazılırken ekran boş görünecektir.

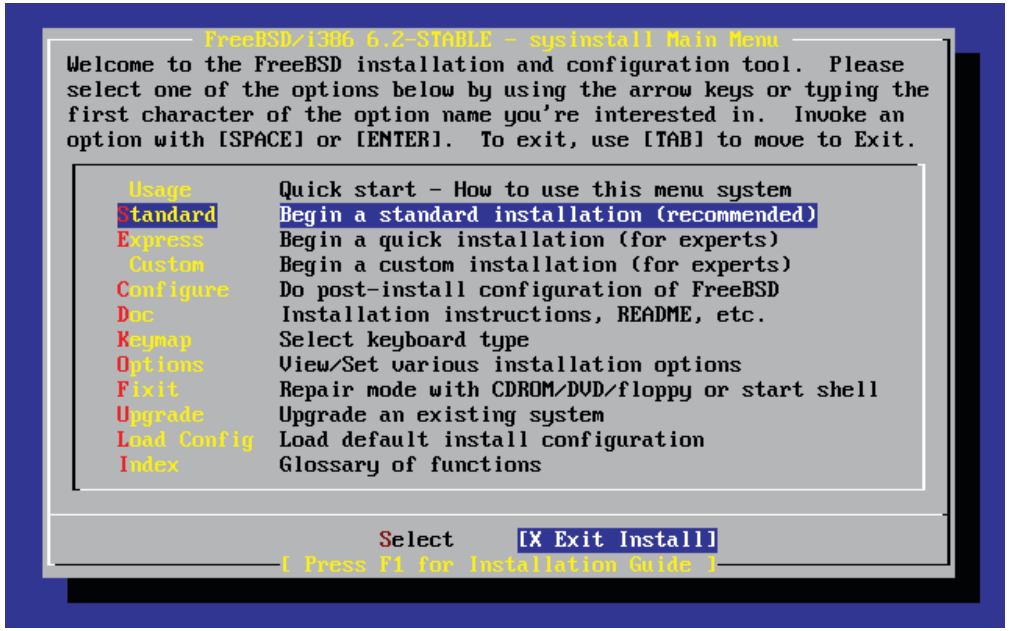
```
 New password:
 Retype new password :
```

Şifre girişleri eşleştiğinde atama yapılmış olur. Eşleşmeyen hatalı yazımlar nedeni ile aynı işlemi tekrarlamanız gerekir. Şifre ataması tamamlanınca sistemin genel yapılandırma seçeneklerini düzenlemek isteyip istemediğiniz sorulacaktır.

```
                User Confirmation Requested
Visit the general configuration menu for a chance to set any
last options?

                        Yes    [ No ]
```

Ağ ve sunucu servisleri ile ilgili yapılandırmayı yeniden düzenlemek isterseniz bunu yapabilirsiniz. Eğer yapılandırmanız bitti ise 'No' basarak çıkış yapın. Sysinstall'a geri döneceksiniz.



Exit Install'a bastığınızda onaylama mesajı görüntülenecektir. Onayladığınızda sistemi yeniden başatmadan önce sürücülerde bulunan disket, CD/DVD gibi şeyleri çıkartın.

```
                User Confirmation Requested
Are you sure you wish to exit? The system will reboot (be
sure to remove any floppies/CDs/DVDs from the drives).

                        [ Yes ]   No
```

FreeBSD Başlarken...

FreeBSD kurulumu tamamlandığında sisteminizi yeniden başlattığınızda eğer FreeBSD açılış yöneticisini kurduysanız bu durumda sizden başlatılacak olan sistemi seçmeniz istenecektir veya tek sistem olarak kurduysanız doğrudan FreeBSD başlatılacaktır. FreeBSD başladığında açılış seçeneklerini göreceksiniz.

Bir seçim yapmazsanız FreeBSD açılacaktır. Ekranda kernel tarafından bulunan donanımların çalıştırıldığını ve bu donanımlar ile ilgili birçok mesajın görüntülendiğini göreceksiniz. Kernel tarafından verilen mesajlar parlak beyaz renklidir. kernel kendi işini bitirdiğinde açılış sürecini init programına bırakır. 'init'e ait olan mesajlar ise gri renklidir.

Açılış süreci sona erdiğinde grafik arabirim karşınıza gelmeyecek. Onun yerine kullanıcı adı ve şifresi ile giriş yapabileceğiniz kabuk ortamı sizi karşılayacaktır. Grafik arabirim kurulu olduğuna göre X'in başlaması gerekecektir.

X11 Yapılandırması

X sunucusu kurulumu eksiksiz gerçekleşmiş ise bu durumda X'i başlatabilirsiniz.

```
startx
```

Yazdığınızda X sunucu çalışacak ve karşınıza TWM pencere yöneticisi gelecektir. Eğer gelmiyorsa, kurulum sırasında, X'in çalışmasını sağlayan bazı paketler eksik olabilir. Bu nedenle Xorg paketinin kurulması gerekecektir. Root olarak

```
# pkg_add -vr xorg
```

xorg ile gerekli olacak diğer sürücüler vd. kurulmasını sağlayacaktır. Bu işlemler zaman alıcı olacaktır. Zira tamamen internet bağlantınıza ve uzak sunucudaki yüke bağlı olarak değişecektir. Kurulum tamamlandıktan sonra X'i yeniden başlatmayı denediğinizde çalışacaktır.

TWM pencere yöneticisi ile çalışmak istemiyorsanız KDE, GNOME vs. bir diğer pencere yöneticisini kullanabilirsiniz. Bunu iki şekilde yapabilirsiniz. Birincisi sisteme kabuk ortamından girip grafik ortamı startx ile başlatmaktır.

Bunun için kendi ev dizininizde .xinitrc adlı dosyayı oluşturmanız ve kullanmak istediğiniz masaüstünü tanımlamanız gerekir.

GNOME kurduysanız ve GNOME başlatmak için

```
echo "/usr/local/bin/gnome-session" > ~/.xinitrc
```

KDE3 kurduysanız ve KDE3 başlatmak için

```
echo "exec startkde" > ~/.xinitrc
```

KDE4 kurduysanız ve KDE4 başlatmak için

```
echo "exec /usr/local/kde4/bin/startkde" > ~/.xinitrc
```

Xfce4 kurduysanız Xfce4 açılması için ise

```
echo "/usr/local/bin/startxfce4" > ~/.xinitrc
```

komutunu verdikten sonra startx komutunu vermeniz yeterli olur.

Eğer doğrudan X'i başlatıp KDM vb. kullanmak istiyorsanız KDM'nin aktif hale gelmesi gerekir. Bunun için ttys terminallerinden birinde KDM'nin çalıştırılması gereklidir. /etc/ttys dosyasını bir metin editörü ile açıp dosyada yer alan attyv8 satırını bulun.

```
#
# $FreeBSD: src/etc/etc.i386/ttys,v 1.10 2003/10/24 15:44:08
# simokawa Exp $
#       @(#)ttys        5.1 (Berkeley) 4/17/89
# This file specifies various information about terminals on
# the system. It is used by several different programs.
# Common entries for the various columns include:
#
# name  The name of the terminal device.
#
# getty The program to start running on the terminal.
# Typically a
# getty program, as the name implies.  Other common entries
# include none, when no getty is needed, and xdm, to start the
#       X Window System.
#
# type The initial terminal type for this port.  For hardwired
# terminal lines, this will contain the type of terminal used.
# For virtual consoles, the correct type is typically cons25,
# but vt220 will work better if you need interoperability
# with other systems like Solaris or GNU/Linux.
# Other common values include network for network connections
# on pseudo-terminals, dialup for incoming modem ports, and
# unknown
# when the terminal type cannot be predetermined.
#
# status Must be on or off.  If on, init will run the getty
# program on
#       the specified port.  If the word "secure" appears,
# this tty
#       allows root login.
#
# name  getty                 type    status          comments
#
# If console is marked "insecure", then init will ask for the
# root password
# when going to single-user mode.
console none                            unknown off secure
#
ttyv0   "/usr/libexec/getty Pc"         cons25  on  secure
# Virtual terminals
ttyv1   "/usr/libexec/getty Pc"         cons25  on  secure
ttyv2   "/usr/libexec/getty Pc"         cons25  on  secure
ttyv3   "/usr/libexec/getty Pc"         cons25  on  secure
ttyv4   "/usr/libexec/getty Pc"         cons25  on  secure
ttyv5   "/usr/libexec/getty Pc"         cons25  on  secure
ttyv6   "/usr/libexec/getty Pc"         cons25  on  secure
ttyv7   "/usr/libexec/getty Pc"         cons25  on  secure
ttyv8   "/usr/local/bin/kdm -nodaemon"  xterm   on  secure
# Serial terminals
# The 'dialup' keyword identifies dialin lines to login,
# fingerd etc.
```

KDE3 için

```
ttyv8 "/usr/local/bin/kdm -nodaemon" xterm on secure
```

KDE4 için

```
ttyv8 "/usr/local/kde4/bin/kdm -nodaemon" xterm on secure
```

Olacak şekilde değiştirin.

Sistemi yeniden başlattığınızda tercih ettiğiniz giriş yöneticisi ve masaüstünü kullanmaya başlayabilirsiniz.

BSD - IV

Diğer BSD Kurulum Seçenekleri

Gökşin Akdeniz
goksin@enixma.org

Önceki yazıda BSD kurulumu için DVD ve CD setlerini kullanarak BSD tek işletim sistemi olarak kurulumunu açıklamştık. Bu yazı ise önceki yazıda ele aldığımız konuları tamamlayacak şekilde diğer kurulum seçeneklerini ele alıyor. Bu kapsamda ağ üzerinden kurulum seçenekleri ve başka bir işletim sistemi ile birlikte BSD kullanımını ele alacağız.

Diğer Kurulum Seçenekleri

BSD kurulumunu ele aldığımız önceki yazıda BSD'nin sisteminize tek işletim sistemi olarak kurulacağını varsayarak kurulum basamaklarını anlatmıştık. Ancak kullanmakta olduğunuz sistemde BSD'yi tek sistem olarak kullanmanız olanaklı değilse, Windows veya Linux sisteminizin yanında BSD de kurmayı düşünüyorsanız bu yazıda ele alacağımız konular size yardımcı olacaktır.

Birden fazla işletim sistemini birlikte kullanmak tek bir sistemi kurup kullanmaya göre farklıdır ve bilgisyarınızı açtığınızda hangi sistemi kullanacağınızı seçmek için gereken yapılandırma ve elinizde kurulum için CD/DVD olmadığı durumlarda izlenecek olan kurulum yöntemleri bu yazının konularını oluşturuyor.

Sisteminizde kurulu olan Windows/linux sisteminizin yanına BSD kurmaya başlamadan önce kurulu olan işletim sisteminizde bulunan verilerinizin yedeklerini almanız gerekli. Bu veriler kişisel dosyalarınız, kurduğunuz yazılımlar vs. olabilir. Bunların yedeğini almadan okuyacaklarınızı uygulamaya geçmeyin. Var olan sisteminiz tüm disk alanını kullanıyorsa var olan sisteminizi ya kaldırıp yeniden diskinizi bölümleyek kurulum yapmanız gerekecektir veya sisteme zarar vermeden diskinizde yer açabilen yazılımları kullanarak diskinizde BSD kurmak için yeterli alan açmanız gerekecektir.

Diskinizde yer açmak için herhangi bir yazılım seçmeden önce kullandığınız dosya sistemine uygun olup olmadığını kontrol etmeniz gerekecektir. Partitionmagic Windows kurulumları için uygun bir seçenek olacaktır. Linux kurulumları için Gparted iş görebilir. Eğer partitionmagic kullanmak istemiyorsanız BSD ile gelen veya interneten indirebileceğiniz FIPS de kullanılabilir. Ancak FIPS'in kısıtı FAT ve FAT32 dosya sistemleri dışındakileri desteklememesidir.

Birden Fazla İşletim Sistemi Kullanırken Karşılaşabileceğiniz Sorunlar

Bugün için Windows ve Linux işletim sistemlerini hatta Mac OS X'i bir bilgisayar üzerinde aynı disk üzerindeki farklı bölümlere kurarak kullanabilirsiniz. Linux kurulumunda Windows ile sistemi başlatmak için gereken seçenek otomatik olarak yapılandırılabilir. BSD de sizin için bu işlemi gerçekleştirebilir. Ancak ve ancak diskinizde sistemin açılması için gereken bilgilerin bulunduğu bölümün doğru bilgiyi barındırması ile bu işlem gerçekleşebilir.

Diskinizin ilk 1024 silindiri, sistemin açılması için gerekli bilgilerin bulunması gereken bölümdür. Eğer bu bilgi, burada bulunmuyorsa veya erişilemiyorsa sistem açılmayacaktır. BSD sistemler kendi açılış yöneticileri ile geldiği için Linux sisteminlerdeki GRUB veya LILO'ya gerek kalmadan açılacak sistemi seçebilirsiniz. BSD açılış yönetisinin ise yukarıda adı geçen ilk 1024 silindir içinde bir yerlerden başlayarak diskinize yazılmış olması zorunludur. Diskinizde yer açabilen disk bölümlerinizi yeniden boyutlandıran programlar bu zorunluğu dikkate alarak gereken işlemleri yapar. BSD sisteminizi açabilmeniz için en azından root bölümünün tamamının bu ilk 1024 silindir üzerinde bulunması gerekir. Eğer root bölümünü ilk 1024 silindir üzerinde bir yere kuramıyorsanız bu durumda en azından /boot bölümünün tamamının bu alanda yer alması gereklidir. Bir diğer deyişle /boot bölümünün başlangıç ve bitiş silindirlerine ait değerler 1024'ten az olmalıdır. Başlangıcınız 1024'ten küçük iken bitişiniz 1024'ten büyük ise BSD gene açılmayacaktır. Diskinizde yer açarken bu duruma dikket etmeniz ya da en azından birincil disk bölümünün BSD için de gereken yere sahip olduğundan emin olmanız gerekir.

Windows/BSD Kurulumu

Birden fazla işletim sistemini kullanmayı tercih ederseniz dikkat etmeniz gereken nokta ilk olarak Windows/Linux kurmanız gerekeceğidir. Windows/BSD kurulumu yapacak iseniz öncelikle Windows kurmanız gereklidir. Aksi durumunda Windows kurulumunun ardından BSD açılış yönetisinizin üzerine yazılacağı için sadece Windows açılacaktır. Bu nedenle önce Windows kurmanız ve ardından BSD kurulumu yapmanız yerinde olacaktır.

BSD, kendi açılış yöneticisini kullanmak Windows/BSD kurulumları için yeterli bir çözümdür. Kurulum sırasında "Install the FreeBSD Boot Manager" seçeneğini seçmeniz gerekir. Kurulum bittikten sonra sisteminizi her başlattığınızda BSD açılış yükleyicisi sizi Windows veya BSD seçimi yapmanızı isteyen basit bir açılış yöneticisi ile karşılayacaktır. Yapmanız gereken uygun F tuşuna basarak işletim sistemini seçmektir.

Linux/BSD

Eğer sisteminizde Linux bulunuyorsa ve Linux/BSD kurulumu yapmayı düşünüyorsanız, iki seçenekten birisini seçmeniz gerekir.

BSD açılış yönetisini kullanarak BSD/Linux seçimi yapacak iseniz bu durumda BSD açılış yöneticisinin Linux sisteminizi açmak için kullanamazsınız. Onun yerine BSD açılış yöneticisini Linux açmak için GRUB veya LILO'yu başlatmak için kullanırsınız.

Yapılması gereken BSD açılış yönetisini LILO/GRUB master Boot Record/ana açılış kaydı üzerine değil, MBR'dan sonraki bölüme kurmanızdır. Bu şekilde GRUB/LILO ile Linux'u açabilirsiniz. BSD açılış yönetisinden Linux bölümünü seçmeniz durumunda GRUB /LILO karşınıza gelecektir.

Eğer LILO/GRUB üzerinden BSD'yi açmak isterseniz aşağıdaki düzenlemeleri yapmanız gerekecektir.

BSD'yi LILO üzerinden Başlatmak:

LILO yapılandırma dosyası olan /etc/lilo.conf dosyasını bir metin düzenleyici ile açıp aşağıdaki satırları ekleyip kayıt edin.

```
other=/dev/hdx
table=/dev/hda
label=FreeBSD
```

/dev/hdx olarak görülen alana diskinizde BSD kurulumu yaptığınız disk bölümünü yazmanız gereklidir. LILO yapılandırmasını tamamladıktan sonra root olarak lilo yazarak yapılandırmanızı güncelleyin. Bir sonraki açılışta BSD sistemini seçip başlatabilirsiniz.

BSD'yi GRUB üzerinden başlatmak:

GRUB yapılandırma dosyası olan /boot/grub/menu.lst dosyasını bir metin düzenleyici ile açıp aşağıdaki satırları ekleyip kayıt edin.

```
title FreeBSD
rootnoverify (hd0,0)
chainloader +1
```

Diskinizde BSD'nin kurulu olduğu bölüm birincil bölüm olarak hd0,0 şeklinde tanımlanmıştır. Gereken düzenlemeleri kendi sisteminizde yapılandırmaya göre düzenlemeniz gerekmektedir.

FreeBSD Boot Manager- FreeBSD Açılış Yöneticisi

BSD kurulumu sırasında sisteminizde birçok işletim sistemini kullanacak şekilde düzenleme yaptıysanız açılışta BSD açılış yöneticisini kullanarak BSD ve diğer işletim sistemleri arasında seçim yapabilirsiniz. Kurulum sırasında BSD açılış yöneticisi yapılandırılmış olacaktır. Sonradan açılış yönetcisini tercihleriniz doğrultusunda düzenleyebilirsiniz. Düzenleme yapmak için boot0cfg kullanabilirsiniz.

boot0cfg grafik bir arayüze sahip değildir. Kabuk ortamında root yetkileri ile kullanılır. Kullanabileceğiniz seçeneklerin tamamını akılda tutmaya gerek yoktur. Gerektiğinde tüm seçenekleri kılavuz sayfasından edinebilirsiniz. (man boot0cfg)

```
# boot0cfg -B
```

Bu komut MBR kaydına BSD açılış yönetisini kurar. Sisteminizde kurulu olan diğer işletim sistemlerinden birisinin herhangi bir nedenle yeniden kurulması durumunda MBR kaydı silinmiş olacağı için BSD açılış yöneticisini yeniden kurmak için kullanabilirsiniz. Bu işlemin ardından gereken düzenlemeleri yapmak için diğer seçenekleri kullanmanız gereklidir.

```
# boot0cfg -v
```

Verbose/Anti sessiz. boot0cfg'nin gerçekleştirdiği işlem basamaklarını izleyebilirsiniz.

```
# boot0cfg -b image
```

image: Açılış sırasında kullanılacak olan imaj dosyası. Varsayılan imaj dosyası /boot/boot0 dosyasıdır. Diğer bir imaj dosyasını kullanacak iseniz imaj dosyasının bulunduğu yeri /dizin/imaj olarak tanımlamanız gereklidir.

```
# boot0cfg -d disk
```

disk: Açılış için kullanılacak olan disk. Bilgisayarınızın BIOS'u ilk diski 0x80 ve ikinci diski de 0x81, vs. olarak tanımlar. Açılış için kullanacağınız diski tanımlamak için doğru değeri kullanmanız gerekir. Örneğin ilk diskten başlatmak için boot0cfg -d 0x80

```
# boot0cfg -f dosya_adı
```

dosya_adı: İşlem yaptığınız MBR'ın yedekleneceği dosyanın adı. MBR kaydını yedekler ve gerektiğinde bu kayıt geri yüklenebilir. -f dosya_adı ile yedeğin adı tanımlanır. Dosya bulunmuyorsa oluşturulacaktır. Eğer dosya var ise, üzerine yazılacaktır.

Ağ Kurulumu ile BSD Kurmak

BSD kurulumunu CD/DVD ile gerçekleştiremiyorsanız veya tercih etmiyorsanız, diğer kurulum seçeneklerini kullanabilirsiniz. Bu seçenekler arasında FTP veya NFS ile kurulum da bulunmaktadır. Eğer kurulum yapacağınız sistemde CD/DVD sürücüleri bulunmuyorsa veya sistemin CD/DVD açılış yapması kısıtlanmış ise ve de yerel ağ veya internet erişiminde bant genişliği sıkıntınız bulunmuyorsa bu seçenek en uygunu olacaktır. Kurumsal ağlarda bu tür kısıtlamalara gidildiği için yerel ağ üzerinde bulunan bir sunucudan yararlanılarak kurulum yapmak daha kolay olmaktadır.

Ağ üzerinden kurulum yapabilmeniz için minimal BSD kurulumu yapmmanız gereklidir. Bu minimal sistem kernel ile sysinstall programından oluşmaktadır. BSD kurulu olan sisteminizde sysinstall ile disketler oluşturabilirsiniz. Önceki yazılarda ele aldığımız gibi boot.flp dosyasını barındıran disket ile sistemi açıp diğer disketleri kullanarak kuruluma başlayabilirsiniz.

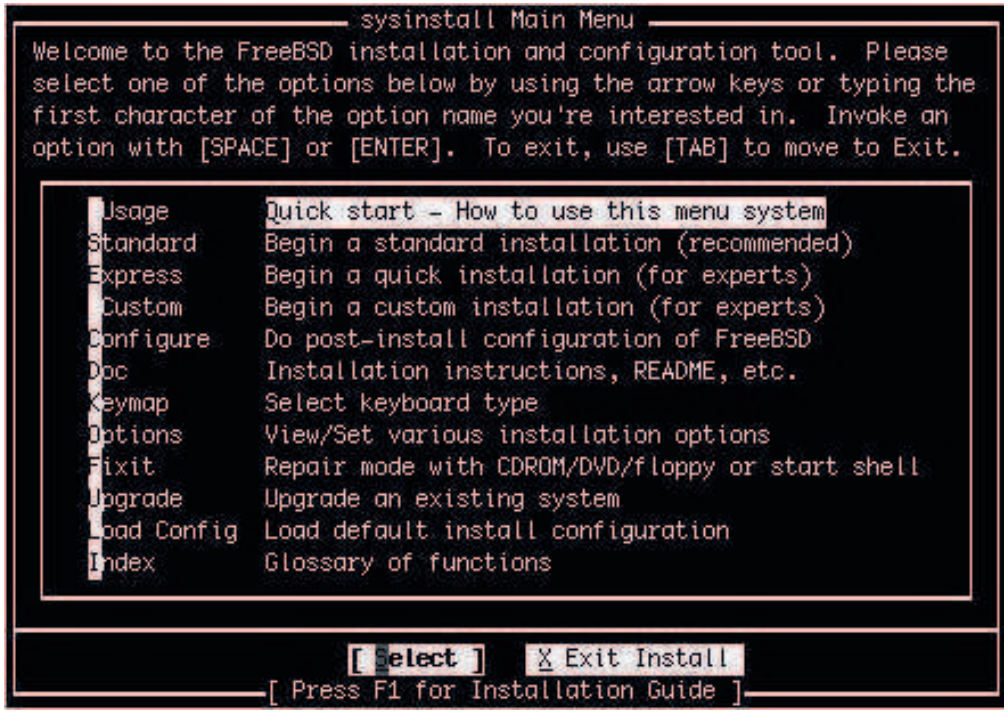
FTP Sunucularından Kurulum

FTP - File Transfer Protocol- bir ağ üzerinde bir bilgisayardan diğerine dosya aktarmanın en bildik yoludur. Sunucuya giriş yaparken güvenli bir bağlantı kullanmadığımız, kullanıcı adları ile şifrelerin metin olarak ağ üzerinden aktarıldığı bir protokol olmasına rağmen, bugün yaygın olarak kullanılmaya devam edilmektedir. Tabii ki BSD sisteminiz ile bir FTP sunucusu kurabilirsiniz. Bunu ilerleyen bölümlerde ele alacağız.

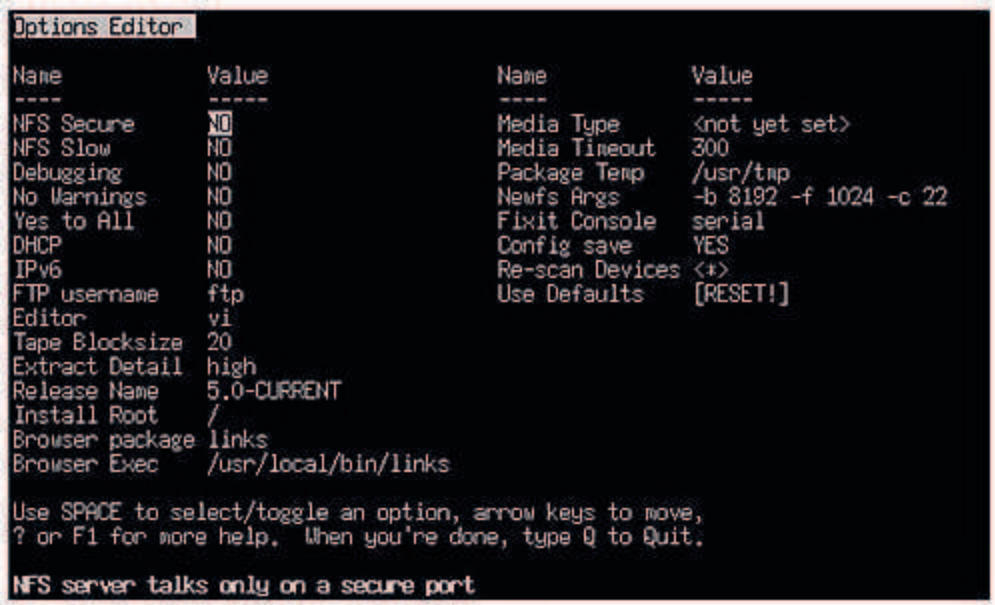
Bir FTP sunucusu üzerinde bulunan kurulum dosyalarını kullanarak BSD kurulumunu gerçekleştirebilirsiniz. Bunun için hızlı bir ağa gereksiniminiz olacaktır. Bir diğer deyişle bant genişliği kadar ağ trafiğinin de kısıtlamaya neden olmaması gereklidir. Eğer kurulum için BSD resmi ftp sunucuları veya yansılarından birisini kullanıyorsanız FTP sunucusuna anonim -anonymous- giriş yapabilirsiniz. Ancak yerel ağ üzerinde bulunan özel bir ftp sunucusunu kullanıyorsanız bu durumda ftp sunucu erişimini yapılandırmanız gerekecektir.

FTP Erişiminin yapılandırılması

Sysinstall FTP erişimleri için öntanımlı kullanıcı adı ve şifresini kullanmaktadır. Bunu değiştirmek için öncelikle sysinstall'u çalıştırmalı (RESİM 1) ve ok tuşlarını kullanarak OPTIONS (RESIM 2)



Resim 1



Resim 2

bölümüne gelmeniz gereklidir.

Ekranda FTP username seçeneği üzerine gelip boşluk tuşluna basın. Karşınıza gelen dialog kutusuna kullanıcı adını yazın ve entere basın. Ardından tanımladığınız kullanıcı adına atanmış olan şifreyi aynı biçimde tanımlayın. Kullanmanız gereken kullanıcı adı ve şifresini ilgili sistem yöneticisinden alabilirsiniz veya önceden tanımlanmış olan bir kullanıcı adı ve şifresi varsa bunu edinmeneiz yerinde olacaktır. Bu işlemin tamamlanmasının ardından Q tuşuna basarak sysinstall'a geri dönün.

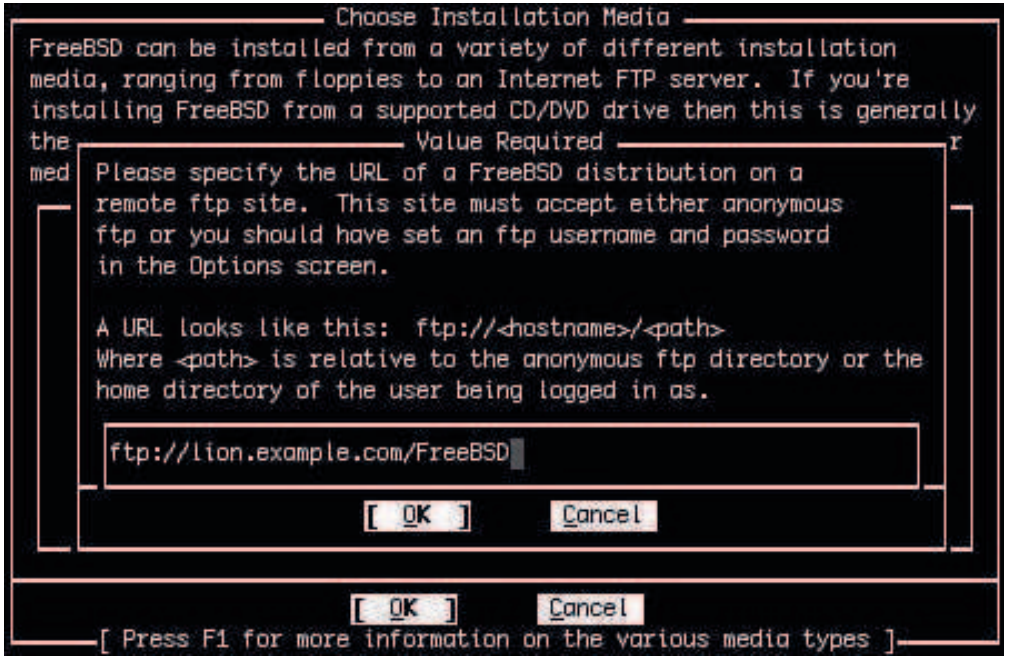
FTP Kurulumuna Başlama

FTP erişimi için kullanıcı adı ve şifresi yapılandırılması aşamasından sonra önceki yazılarda ele aldığımız kurulum aşamalarını gerçekleştirmeniz gereklidir. Diskinizi bölümleyip, etiketledikten sonra

Resim 3

paket seçimini yapmanız sonra da diğer kurulum basamaklarını tamamlamanız gerekecektir. Paket seçiminin ardından kurulum kaynağının tanımlandığı ekranda FTP veya FTP Passive'i seçmeniz gerekecektir. Eğer firewall ile yerel ağı korunuyorsa FTP passive seçimi gerekecektir. Seçim konusunda emin değilseniz sistem yöneticisinden bu konu ile ilgili bilgi alabilirsiniz. Bu işlemin ardından FTP sunucusu seçim aşamasına geleceksiniz. (RESİM -3)

Eğer BSD yansılarından birisini kurulum için kullanıyorsanız coğrafi olarak bulunduğunuz yere en yakın olan sunucuyu seçmeniz önerilecektir. Böylelikle kurulum süresi kısalmış olur. Ancak sunuculardaki ve ağ üzerindeki yüke bağlı olarak değişkenlik gösterebilir. Bir BSD yansısı kullanacak iseniz listeden yansı seçimi yapmanız yeterli olacaktır. (RESİM -4) Ancak yerel bir sunucuyu

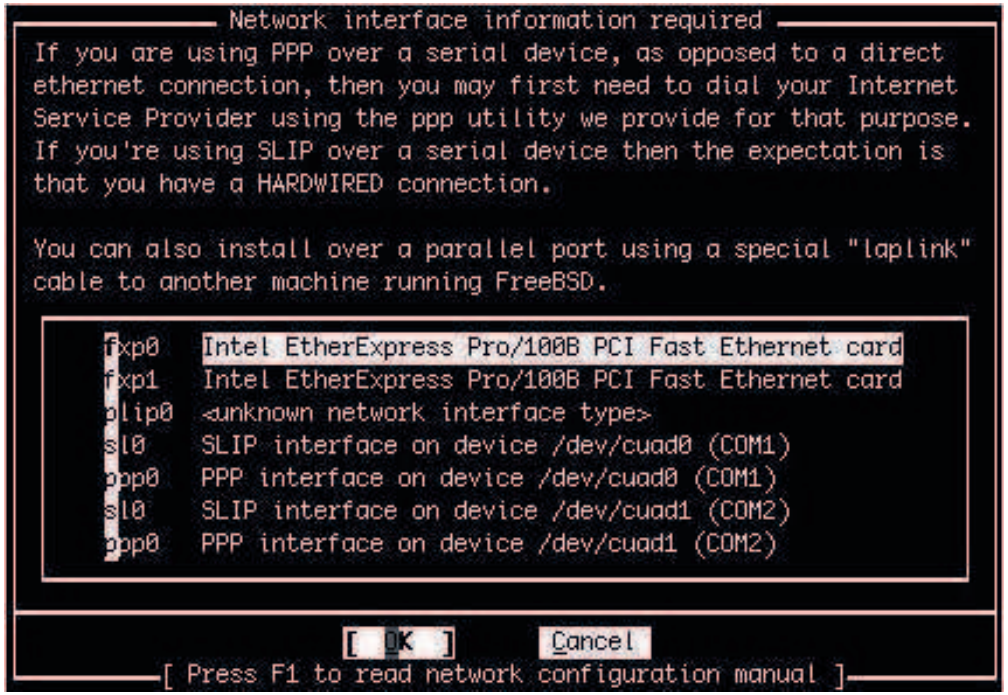


Resim 4

kullanacak iseniz sunucu adresini tanımlamanız gerekecektir. Sunucu adresi dışında sunucuda bulunan ve BSD kurulumu için gerekli olan dosyaların bulunduğu dizini de tanımlamanız zorunludur. Örneğin ftp.sunucu.net/BSD gibi

FTP sunucunun adresini tanımladıktan sonra ağ bağlantısını yapılandıracaksınız. (RESİM 5) Ağ yapılandırmasını geçmiş sayılarda kurulum konusunda yer verdiğimiz gibi gerçekleştirip sysinstall menüsünden ilgili diğer ayarları tamamlayabilirsiniz. Sysinstall'a geri döndüğünüzde ağ yapılandırması ile yeniden uğraşmanıza gerek kalmadan diğer ayarlarınızı yapabilirsiniz.

Kuruluma geçtiğinizde sunucudan dosyalar bilgisayarınıza kopyalanacak ve kurulum gerçekleştirilecektir. Bu işlem bir kaç saatinizi alabilir. Bu süre içinde başka işleriniz varsa onları tamamlayın.



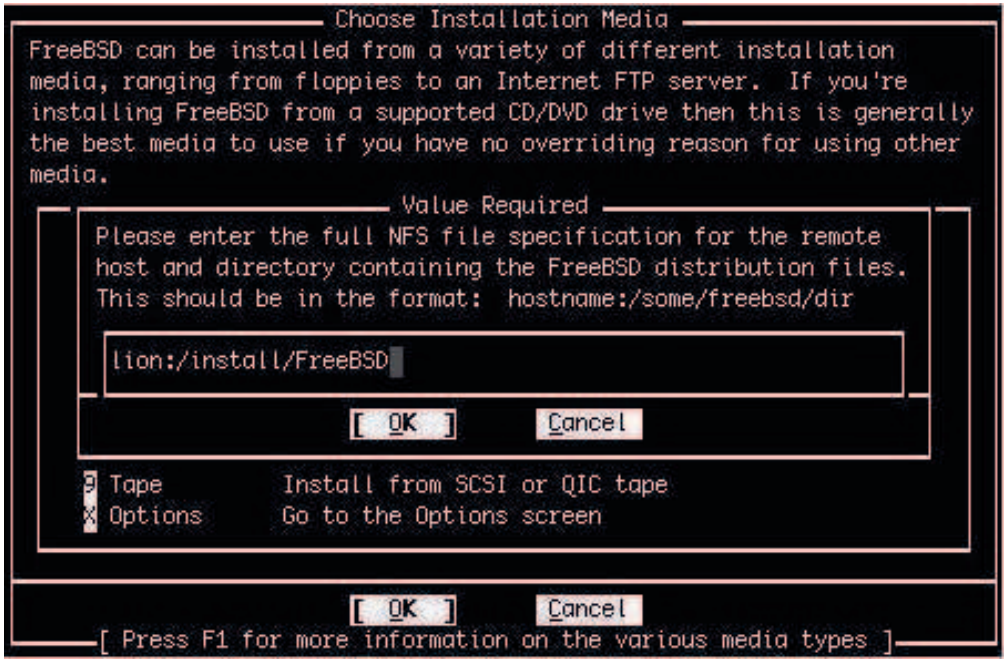
Resim 5

Kurulum işlemi bitince kurulum sonrası yapılandırma ve diğer ayarlarınızı yapabilirsiniz.

Eğer ağ üzerinden kurulumu BSD yansılarından birisini kullanarak gerçekleştiriyorsanız yerel ağ üzerindeki bir sunucuyu kullanmanın size sağlayacağı zaman kazancını tahmin edebilirsiniz. Yerel ağ üzerinde bulunan bir bilgisayara BSD kurup bunu ağ üzerinden BSD kurmak için kullanabilirsiniz. Gelecek yazılarda bu konuya BSD ağ servisleri konusunda yer vereceğiz.

NFS Kurulumu

NFS -Network File System- bir ağ üzerinde bulunan uzak bilgisayarlarda paylaşılan dizinleri kendi bilgisayarınızdaki bir



Resim 6

dizin gibi kullanmanızı sağlayan bir protokoldür. BSD'yi NFS ile de kurabilirsiniz. Yukarıda değindiğimiz gibi bir BSD sistemi NFS sunucusu olarak hazırlayıp ağ üzerindeki diğer sistemlere de BSD kurulumu yapabilirsiniz. NFS ile kurulum da FTP ile aynı biçimde gerçekleşmektedir. FTP sunucusunun yerini NFS sunucusu alır.

Yavaş Bağlantılar için Yapılandırma

Eğer NFS sunucusu güvenli bağlantı ile yapılandırılmışsa veya elinizdeki ağ kartınızın performansı nedeni ile bağlantınız yavaşlıyorsa, (RESİM - 6) gördüğünüz yapılandırma bölümünden NFS seçeneğini secure olarak seçin. İlgili seçeneğin üzerine ok tuşları ile gelip boşluk tuşu ile aktif hale getirin. Q tuşuna basarak yukarıda anlattığımız gibi kuruluma geçin.

Eğer ağ kartınız ile ilgili bir sorun yoksa ve NFS sunucusu güvenli bağlantı kullanmıyorsa yukarıda değindiğimiz güvenlik aşamasını atlayabilirsiniz. Kurulum kaynağı olarak NFS'i seçin ve NFS sunucusunun adresini ve sunucuda bulunan BSD kurulum dosyalarının bulunduğu dizini tanımlayıp ağ yapılandırmasını gerçekleştirin. Yukarıda değindiğimiz gibi ardından kurulum için seçtiğiniz paketler kopyalanıp kurulmaya başlanacaktır.

Ağ kurulumları CD/DVD kurulumlarına göre uzun sürmektedir. Öte yandan ise CD/DVD taşımaya gerek kalmadan sadece bir kaç disket ve ağ yapılandırması ile kurulum ağ üzerinde bulunan bir NFS/FTP sunucusu ile gerçekleştirilebilir.

BSD - V

X ve ayarları

Gökşin Akdeniz
goksin@enixma.org

BSD sistemlerde grafik arabirimler BSD kurulumu sırasında sistem yöneticisinin tercihine bırakılır. Eğer X kurmayı seçtiyseniz X zaten kurulmuştur ama ilk giriş yaptığınızda grafik arayüzler sizi karşılamayacaktır. Önceki bölümlerde yer verdiğimiz gibi X yapılandırmasını kendi tercihlerinize göre yapabilirsiniz. Bu bölümde ise X kurulumu yapmadığınızı veya yaptıysanız KDE dışındaki diğer pencere yöneticilerini nasıl kurup yapılandırabileceğinizi göreceksiniz.

BSD sistemlerde X Window System-X11 grafik arayüzü XFree86 ve sonrasında Xorg ile kullanılabilmektedir. FreeBSD 5.2.1-RELEASE ile XFree86, sonra ise FreeBSD 5.3-RELEASE ile Xorg kullanılabilmektedir. Bu yazı Xorg 7.3 temel alırken, 7.4 sürümünün port'lara girmesi ile yer yer de 7.4 sürümüne yer vereceğiz.

X Nedir?

X, UNIX için yazılmış olan ilk grafik arayüz olmamakla birlikte en yaygın olarak kullanılanıdır. X adının verilmesi ise geliştirme ekibinin X'ten önce İngilizce Window-pencere sözcüğünden dolayı W adını verdikleri grafik arabirim projesinin ardından başlamaları ile W'den sonra gelen harf olan X'i kullanmalarıdır.

Grafik arabirim ayrıca X, X Pencere Sistemi - X Windows System veya X11 olarak da adlandırılır. Bazen X Windows olarak da adlandırılır. X'i yapılandırmak için tüm bileşenleri ile nasıl çalıştığını kavramak gerekli değildir. Temel düzeyde bilgi edinmek kendi gereksinmelerinize göre gereken yapılandırmayı yapabilmenizi sağlar. Bu nedenle X çalışma biçimi ile temel yapılandırma ayarları konusunda bilgi edinmek yeterlidir.

İstemci-Sunucu Modeli

X başında beri ağ üzerinden çalışacak biçimde tasarlanmış ve geliştirilmiştir. Bu nedenle de istemci-sunucu mimarisi kullanılmıştır.

X'in üzerinde geliştirildiği modelde X sunucusu, klavye, monitör ve farenin bulunduğu bilgisayarda çalışır. Sunucunun görevi klavye, fare veya diğer bir girdi cihazından -örneğin bir tablet PC ekranı gibi- gelen girdileri almak, görüntüyü kontrol etmek veya bir istemciye aktarmaktır. Bu istemci bir projeksiyon cihazı da olabilir. X üzerinde çalışan herhangibi bir uygulama, -örneğin firefox ve xterm gibi- bir istemcidir. Bir istemci, sunucuya bir mesaj

gönderip "şu koordinatlarda bir dikdörtgen çiz" diyebilir. Sunucu da istemciye kullanıcı şu düğmeye tıkladı gibi bir mesaj gönderecek ve isteğin gereğini yerine getirecektir.

Bir ev veya ofis kullanıcısı iseniz X sunucusu ve istemcisi aynı sistem üzerinde çalışacaktır. Öte yandan X sunucusunu başka bir makinada çalıştırıp diğer uygulamaları da başka sistemlerde çalıştırabilirsiniz. Bu durumda X sunucusu ve istemcisi ağ üzerinden haberleşecektir. Bu senaryoda kullanacağınız X sunucusunun çalışacağı makinanın güçlü bir sistem olması gerekli değilken, istemcilerin ise tersine daha güçlü bir sistem olmasını gerektirir.

Bu son cümle bir çok kullanıcı için anlaşılmaz gelebilir. Zira bir çok kullanıcı X sunucusunun büyük ve güçlü bir sistemde çalışmasını, istemcilerin ise kendi makinalarında olması gerektiğini düşünür. Yukarıda açıkladığımız gibi sunucunun klavye, fare ve monitörün takılı olduğu sistemde yer alması gerekir. X istemciler ise pencerelerde uygulamaların grafik yüzlerini sunanlardır.

Kullanılan protokolün yapısı gereği istemci ve sunucunun aynı işletim sisteminde çalışmasını hatta aynı tip bilgisayar dahi olmasını gerektirmez. X sunucusu MacOS veya Windows üzerinde çalışabilirken istemciler farklı olabilir. Buna olanak veren bir çok ticari uygulamayı bulabilirsiniz.

X Pencere Yöneticisi

X tasarımındaki ana fikir, UNIX tasarımının ana fikrine çok benzerdir; "politika değil, araç". Daha basitçe söylersek, X, bir işin nasıl yapılacağını tanımlamaz, tersine kullanılacak olan araçları sunar. Araçları nasıl kullanacağınız ise size kalmıştır. Dolayısıyla X, pencelerin nasıl görüneceğini, düğmelerin ve araç çubuklarının nasıl olması gerektiğini, nasıl pencereleri hareket ettireceğinizi vb. belirtmez.

Tersine, bu işleri, "pencere yöneticisi -window manager" üstlenir. X ile kullanabileceğiniz bir çok pencere yöneticisi vardır. Bunlardan bazıları: AfterStep, Blackbox, ctwm, Enlightenment, fvwm, Sawfish, twm, Window Maker vb. Bu saydığımız pencere yöneticilerinin hepsi farklı araç çubukları, düğme konumları ve renk düzenlerine sahiptir. Farklı klavye tuşlarını kullanarak pencere yöneticisinin farklı biçimde kullanılmasını sağlarlar. Hatta bazıları sanal masaüstleri da sunarak bu masa üstlerinde farklı şekillerde davranabilirler. Bu adı geçen pencere yöneticileri ve ilgili uygulamaları /usr/ports/x11-wm altında bulabilirsiniz.

Bunlar dışında KDE ve GNOME masaüstü ortamları gibi kendi pencere yöneticileri olanlarını da kullanabilirsiniz.

Pencere yöneticilerinin kendilerine ait yapılandırma mekanizmaları bulunmaktadır. Bazıları yapılandırma dosyalarının elle yazılmasını gerektirirken, bazılarının bu yapılandırma için kullanılan garfik arayüzleri de bulunmaktadır. Örneğin Sawfish bir Lisp dialektiği ile yazılan bir ayar dosyası kullanmaktadır.

Widgets

Widget tanım olarak kullanıcı arayüzünde yer alan, düğmeler, listeler vb, şekilde kontrol edilebilen tüm öğeler için kullanılır. Windows ve MACOS sistemlerinde widget konusunda son derece sıkı bir kontrol politkası uygulanır. Tüm uygulamaların görüntüsü tek tip olması istenir. X için ise bu tür bir yaklaşım anlamsızdır. X tasarım anafikri widget konusunda da aynıdır. Politika değil, araç olarak yaklaşılır.

Dolayısıyla X tüm uygulamaların tek tip görünmesini zorunlu tutmaz. Bu konuda serbest bırakır. X için bir çok Widget bulabilirsiniz. Bunlar içinde en yaygın olarak bilinenlerinden MIT'de yazılmış olan Athena,

OpenLook ve Motif sayılabilir. Güncel X uygulamalarında KDE tarafından kullanılan QT, GNOME tarafından kullanılan GTK+ ile yeni kullanıcılar için daha rahat edecekleri bir arayüz elde edilir.

X11 kurulumu

Xorg, BSD sistemlerde de X11 kullanımına olanak sağlar. Xorg, X Window System kaynak kodu üzerinde, Xorg kodu XFree86 4.4 RC2 ve X11 6.6 kodu üzerinde geliştirilmiştir. FreeBSD portlarındaki güncel sürümü 7.4'tür.

Xorg'u ports üzerinden kurmak için 4 GB civarında boş disk alanına gereksinim olacaktır. Kaynak koddan kurmak için:

```
# cd /usr/ports/x11/xorg
# make install clean
```

Öte yandan X11, doğrudan derlenmiş, paketlenmiş olarak pkg_add aracı ile kurulabilir. pkg_add gerekli baketleri ve bağımlılıklarını kuracaktır. pkg_add ile kurulum yapmak istediğinizde sunucular, istemciler, fontlar vb. kurulacaktır.

```
# pkg_add -r xorg
```

Xorg portunun kurulması ile X'in yapılandırlmasına geçebilirsiniz.

X11'i yapılandırmak

X kurulumunu gerçekleştirdiyseniz yapılandırma aşamasına geçebilirsiniz. Ancak yapılandırma aşamasına geçmeden önce sisteminiz hakkında bazı temel bilgilere gereksinim olacaktır. Monitörünüz ve ayarları, grafik kartınızın yonga seti ve bellek miktarı.

Monitör bilgileri, yatay ve düşey frekans, tazeleme hızı, kullanabileceğiniz çözünürlük olacaktır. Bu bilgiler monitörünüz ile gelen kılavuzda yer alır. Eğer elinizde kılavuzunuz yok ise bu bilgileri internetten edinebilirsiniz.

Grafik kartınızın yonga seti X11 sürücü modüllerini tanımlamak için gerekli olacaktır. Bir çok yonga seti X11'i çalıştırdığınızda otomatik olarak tanımlanacağı için gerek duyulmayabilir. Ancak bu bilgiye sahip olmak olası bir otomatik yapılandırma hatasının aşılmasını sağlayabilir.

Grafik kartınızın bellek miktarı, ekran çözünürlüğü ve renk derinliğini kontrol eder. Bu nedenle bu bilgi de kritik öneme sahiptir.

Xorg 7.3 sürümü herhangi bir yapılandırmaya gerek duyulmadan kullanılabilir. X'i başlatmak için

```
$ startx
```

komutu yeterli olacaktır.

Xorg 7.4 ve sonrası için Xorg, klavye ve fareyi HAL aracılığı ile otomatik olarak tanımaktadır. sysutils/hal ve devel/dbus x11/xorg portunun kurulması sırasında ayrıca kurulur. Ancak HAL'in otomatik olarak başlaması için gereken yapılandırmanın sistem yöneticisi tarafından yapılması gerekir. Gereken satırları /etc/rc.conf dosyasına eklemeniz gerekli.

```
hald_enable="YES"
dbus_enable="YES"
```

Bu satırları ekledikten sonra HAL'i ya elle başlatın veya sisteminzi yeniden başlatmanız durumunda otomatik olarak başlayacaktır. HAL ile otomatik yapılandırma bazı donanımlarla gerektiği gibi çalışmamaktadır. Bu durumda elle yapılandırmak sorunu aşacaktır.

Elle yapılandırmaya geçmeden önce bir noktaya dikkat çekmekte yarar var: KDE, GNOME veya Xfce kullanıyorsanız, çözünürlük ve renk derinliği ayarlarını yapmanız için kullanılabilecek olan araçlar sunulmaktadır. Eğer otomatik olarak yapılan yapılandırma sizin için yeterli değilse, bu araçları kullanarak düzenleme yapabilirsiniz.

X11 yapılandırması bir kaç adımda yapılır. İlk olarak yapılaması gereken xorg.conf yapılandırma dosyasının oluşturulmasıdır. Bunu otomatik olarak oluşturmak için aşağıdaki komutu verin:

```
# Xorg -configure
```

Bu işlem ilk aşamada gerekli olacak yapılandırma dosyasını oluşturacaktır. Oluşturulan dosya /root dizininde xorg.conf.new adı ile yer alır. X11 grafik kartınızı ve diğer donanımınızı tanımaya çalışıp gereken yapılandırmayı bu dosyaya kayıt edecektir.

Bundan sonraki adım ise elimizdeki yapılandırma dosyası ile Xorg'un doğru biçimde çalışıp çalışmayacağını kontrol etmek olacak. Xorg'un 7.3 ve üzeri sürümleri için aşağıdaki komutu vermek gerekir:

```
# Xorg -config xorg.conf.new
```

Eğer yapılandırma dosyanız doğru ise karşınıza siyah, gri bir ekran ile X'in fare imleci gelecektir. Bu ekranı sona erdirmek için Ctrl+Alt+Backspace tuşlarına aynı anda basarak X'i sonlandırın. Bu tuş kombinasyonu 7.3 sürümünde standart olarak aktiftir. Ancak 7.4'te durum tersidir. Aşağıdaki satırları xorg.conf dosyanızadaki ServerLayout veya ServerFlags kısmına eklemeniz gerekir.

```
   Option "DontZap" "Off"
```

Xorg 7.4 ile önceki sürümlerdekinin aksine deneme işleminde karşınıza siyah bir ekran gelecektir. Bu siyah ekrana bakarak işlerin yolunda olup olmadığını anlamak kolay değildir. Bu nedenle denemek için twm pencere yöneticisini kullanmak tercih edilebilir.

Xorg 7.4 kullanacak iseniz twm ile yapılandırmanızı aşağıdaki adımları uygulayarak deneyebilirsiniz.

1. xorg.conf.new dosyanızı /etc/X11 altına kopyalayın.

```
# mv xorg.conf.new /etc/X11/xorg.conf
```

2. X11 başlangıç dosyalarınızı yedekleyin. Bu dosyalar ev dizninizdeki .xinitrc veya .Xsession dosyalarıdır.

3. startx komutunu verin. Geçerli herhangi bir ~/.xinitrc dosyası bulunmadığı için tüm sistemde geçerli olan /usr/local/lib/X11/xinit/xinitrc dosyası çalıştırılacaktır. Bu da twm pecere yöneticisini çalıştırır.

4. Eğer X11 doğru çalışıyorsa, CTRL+D tuşlarına basarak twm login ekranına dönebilirsiniz. Eğer sorun var ise sanal terminallerden birisine CTRL+ALT+Fn tuşlarına (n: 1..6) basarak geçin ve X sunucusunu CTRL+C ile sona erdirin.

Diğer ayarlarınızı değiştirmek veya ince ayar yapmak için /etc/X11/xorg.conf dosyasını düzenlemek gerekir. Burada xorg.conf.new dosyası üzerinde bir işlem yapmıyoruz. Düzenleme bittiğinde .xinitrc veya .Xsession dosyalarını kendi ev dizininize kopyalamanız gerekir.

Burada bir noktaya dikkat çekmek gerekiyor. Eğer fare çalışmıyorsa, Xorg 7.4'ten başlayarak xorg.conf dosyasında yer alan InputDevice kısmı donanımların otomatik olarak tanınması için göz ardı edilmektedir. Eski düzeni sevenler için ServerLayout veya ServerFlags kısmına aşağıdaki satırı eklemeleri gerekmektedir.

```
    Option "AutoAddDevices" "false"
```

Böylelikle Xorg'ta önceki sürümlerinde olduğu gibi klavye vb. donanımları eskisi gibi yapılandırabiliriz.

xorg.conf.new veya bizim verdiğimiz adı ile xorg.conf dosyası temel X yapılandırması testini başarı ile geçtiğine göre tercih ettiğiniz metin editörü ile düzenlemeye başlayabiliriz.

İlk adım olarak monitörünüz ile kullanacağınız yatay ve düşey frekans ayarları ile başlayın. Bu değerler xorg.conf.new dosyasındaki "Monitor" kısmında yer almaktadır

```
Section "Monitor"
        Identifier   "Monitor0"
        VendorName   "Monitor Vendor"
        ModelName    "Monitor Model"
        HorizSync    30-107
        VertRefresh  48-120
EndSection
```

Bu kısımdaki HorizSync ve VertRefresh kısımları ilk yapılandırmada yer almayabilir. Monitörün yatay ve dikey tazeleme hızı değerlerini yukarıdaki örnekte görüldüğü gibi uygun biçimde tanımlamak gerekir.

X aynı zamanda DPMS (Energy Star) güç koruma özelliklerini de desteklemektedir. xset(1) aracı, bekleme, askıya alma vb. desteklemektedir. DPMS desteğini kullanmak istiyorsanız monitor kısmına aşağıdaki satırı ekleyin:

```
        Option       "DPMS"
```

Bunlardan başka renk derinliği ve çözünürlüğünü de düzenlemek isteyebilirsiniz. İlgili olan bölüm "Screen".

```
Section "Screen"
        Identifier "Screen0"
        Device     "Card0"
        Monitor    "Monitor0"
        DefaultDepth 24
        SubSection "Display"
                Viewport  0 0
                Depth     24
                Modes     "1024x768"
        EndSubSection
EndSection
```

DefaultDepth ile renk derinliğini tanımlıyoruz. Bu değer sistem için varsayılan değerdir. Ancak bunu Xorg'un -depth seçeneğinde tanımlanan değerler ile değiştirip kullanabilirsiniz.

Bu değişiklikleri yaptıktan sonra dosyamızı kayıt edip yukarıda anlatıldığı gibi yeni ayarlarınızı deneyebilirsiniz.

Yapılandırmayı denerken hataları bulmak ve ayıklamak için X11 log dosyalarını kullanmak yerinde olacaktır. X11 log dosyaları olası hatalar ve modüllere ilişkin bilgileri barındırır. Log dosyası /var/log/Xorg.0.log olarak geçer. Dosyanın tam adı ise Xorg.0.log veya Xorg.8.log vb. şeklinde olabilir.

Yapılandırmanızda hata yoksa /etc/X11/xorg.conf veya /usr/local/etc/X11/xorg.conf olarak kayıt edin.

```
# cp xorg.conf.new /etc/X11/xorg.conf
```

Yapılandırmayı kayıt ettikten sonra startx ile veya xdm ile X başlatılabilir.

Bunların dışında X yapılandırması için X11 ile gelen xorgcfg aracı da kullanılabilir. xorgcfg kullanılacak sürücüleri, ayarları vb. etkileşimli olarak yapılandırmanıza olanak sağlar. xorgcfg -textmode komutu ile başlatılabilir. Bir diğer yapılandırma aracı ise 'xorgconfig'dir. Diğer araçlar gibi kullanıcı dostu olmamakla birlikte

diğerlerinin işe yaramadığı durumlarda kurtarıcı olabilir.

İleri Düzey Yapılandırma Seçenekleri

Intel® i810 Yapılandırması

Bazı anakartlarda Intel® i810 tümleşik grafik kartları kullanılmaktadır. X11 sürücüleri bu kartları kullanırken agpgart AGP programlama arayüzüne gereksinim duyarlar. Bu konu ile ilgili AGP kılavuz sayfasında bilgi bulunmaktadır.

Eğer kendi derlediğiniz çekirdeği kullanıyorsanız ve agp sürücüleri çekirdek yapılandırma dosyanızda bulunmuyorsa kldload ile ilgili çekirdek modüllerini yüklemniz olanaklı olmayacaktır. Sürücülerin derlenip açılış sırasında /boot/loader.conf dosyasında tanımlanarak yüklenmesi gereklidir.

Geniş Ekran LCD Monitörler

Eğer yukarıda anlatılan yapılandırma araçları işe yaramadıysa, X.org.log dosyasındaki bilgilere başvurmak ve tercih ettiğiniz metin editörü ile yeniden işe koyulmak gerekecektir.

Güncel geniş ekran monitörler (WSXGA, WSXGA+, WUXGA, WXGA, WXGA+, vs.) 16:10, 16:9, 10:9 oranlarını desteklemektedir. Bu monitörlerde kullanılacak olan çözünürlükler farklıdır. Örneğin 16:10 için 2560x1600, 1920x1200, 1680x1050, 1440x900, 1280x800 çözünürlükleri kullanılır.

Bazen gereken çözünürlüğün "Screen" bölümüne eklenmesi sorunu çözebilir.

```
Section "Screen"
Identifier "Screen0"
Device     "Card0"
Monitor    "Monitor0"
DefaultDepth 24
SubSection "Display"
    Viewport  0 0
    Depth     24
    Modes     "1680x1050"
EndSubSection
EndSection
```

Xorg, geniş ekran monitörün çözünürlüğünü ve frekansını I2C/DDC bilgisinden elde edebilir. Eğer bu bilgiler ModeLine satırında yer almıyorsa /var/log/Xorg.0.log dosyasından belirlenebilir. Böylece gerekli olan değerler sonradan tanımlanarak kullanılabilir. Dosyada aşağıdakine benzer bir çıktı aramanız gerekiyor:

```
(II) MGA(0): Supported additional Video Mode:
(II) MGA(0): clock: 146.2 MHz   Image Size:  433 x 271 mm
(II) MGA(0): h_active: 1680  h_sync: 1784  h_sync_end 1960
h_blank_end 2240 h_border: 0
(II) MGA(0): v_active: 1050  v_sync: 1053  v_sync_end 1059
v_blanking: 1089 v_border: 0
(II) MGA(0): Ranges: V min: 48  V max: 85 Hz, H min: 30  H
max: 94 kHz, PixClock max 170 MHz
```

Bu satırlar EDID bilgisi olarak tanımlanır. ModeLine satırını tanımlamak ise bu satırlarda yer alan bilgilerin doğru sırada yazılmasından ibarettir.

```
ModeLine <isim> <hız> <4 yatay_tazeleme> <4 düşey_tazeleme>
```

Dolayısıyla da xorg.conf dosyasındaki Monitor kısmını aşağıdakina benzer biçimde yazmanız gerekecektitr.

```
Section "Monitor"
Identifier      "Monitor1"
VendorName      "Vendor"
ModelName       "Model"
ModeLine        "1680x1050" 146.2 1680 1784 1960 2240 1050
1053 1059 1089
```

```
Option          "DPMS"
EndSection
```

Yapılandırmanız bittiğinde X'i yeniden başlatırsanız geniş ekran monitörünüzde X sorunsuz olarak çalışacaktır.

X11 ve Fontlar

5.5.1 Type1 Fonts

X11 ile gelen fontlar günlük kullanım için uygun olmakla birlikte sunumlarda arzulanan etkiyi yaratmaktan uzaktır. Küçük boyutlu fontlar ise Firefox'da görüntülenirken çok küçük görünmektedir. Portlarda birçok kaliteli, ücretsiz ve özgür Type1 (PostScript) fontları bulunmaktadır ve X11 ile kullanabilirsiniz. Örneğin URW fontları x11-fonts/urwfonts (Times Roman, Helvetica, Palatino vd.) kurulup kullanılabilir. Bunlardan başka grafik uygulamaları ile kullanılmak üzere tasarlanmış olan Freefonts x11-fonts/freefonts ekran kullanımı için pek uygun olmamaktadır. Birkaç ayar ile X11'de TrueType fontlarını kullanabiliriz. Kılavuz sayfasında TrueType fontları ile ilgili ayrıntılı bilgi verilmektedir.

Yukarıda adı geçen fontları ports üzerinden kurmak için:

```
# cd /usr/ports/x11-fonts/urwfonts
# make install clean
```

X sunucusunun bu fontları kullanabilmesi için yapılandırma dosyasına, '/etc/X11/xorg.conf'a eklenmeleri gerekmektedir:

```
FontPath "/usr/local/lib/X11/fonts/URW/"
```

Bundan başka doğrudan kabuk üzerinden:

```
% xset fp+ /usr/local/lib/X11/fonts/URW
% xset fp rehash
```

ile de kullanmaya başlayabilirsiniz. Bu yöntemin olumsuz yanı X oturumunu sonlandırdığınızda ayarlarınızın da kaybolmasıdır. Açlış dosyasına startx ile giriş yapıyorsanız ~/.xinitrc dosyasına, eğer grafik arabirim ile giriş yapıyorsanız ~/.xsession dosyanıza yukarıdaki komutları eklemeniz gerekir. Bir diğer çözüm ise

```
/usr/local/etc/fonts/local.conf
```

dosyasını kullanmaktadır. Bu yöntemi anti-aliasing font kurulumu bölümünde ayrıntılı olarak ele alacağız.

TrueType Fontların Kurulumu

Xorg, TrueType fontlar için render desteğine sahiptir. Bu desteği aktifleştirmek için iki seçeneğimiz var. Birincisi freetype modülünü kullanmaktır. Aşağıda vereceğimiz örnekte bu modül diğer font render uygulamaları ile uyumu dolayısıyla tercih edilebilir. Freetype modülünü etkilenştirmek için /etc/X11/xorg.conf dosyasındaki "Module" kısmına şağıdaki satırı ekleyin.

```
Load  "freetype"
```

Bir de TrueType fontlar için bir dizin oluşturulması gerekli. Bu dizini /usr/local/lib/X11/fonts/TrueType veya tercihiniz doğrultusunda uygun bir yerde oluşturun. Burada bir noktaya dikkat çekmek gerekiyor. TrueType fontların X11 tarafından kullanılabilmesi için uygun formatta olması gerekli. Dosyaları sözünü ettiğimiz dizine kopyaladıktan sonra ttmkfdir ile fonts.dir dosyasını oluşturmak durumundasınız. Böylece X font render motoru yeni font dosyalarını kullanabilir. ttmkfdir kurmadıysanız x11-fonts/ttmkfdir portundan kurabilirsiniz.

```
# cd /usr/local/lib/X11/fonts/TrueType
# ttmkfdir -o fonts.dir
```

TrueType fontların bulunduğu dizinin xorg.conf dosyasına eklenmesi gerekli. Bu işlemi Type1 fontlarda olduğu gibi yapabiliriz.

```
$ xset fp+ /usr/local/lib/X11/fonts/TrueType
$ xset fp rehash
```

Bu aşamadan sonra tüm uygulamalar TrueType fontları kullanabilir.

Anti-Aliased Fontlar

Anti-aliasing, X11 ile XFree86 4.0.2'den bu yana kullanılmaktadır. Ancak XFree86 4.3.0'ın çıkışına dek anti-aliasing kullanımı zahmetli bir süreç olmaktaydı. XFree86 4.3.0 ile başlayarak bugüne dek X11 ile kullandığımız tüm fontlar /usr/local/lib/X11/fonts/ ve ~/.fonts/ altındakiler, anti-aliasing destekleyen bir diğer deyişle Xft destekleyen uygulamalar tarafından kullanılmaya başlandı. Xft destekleyen uygulamalar arasında Qt 2.3 ve sonrası, GTK+ 2.0 ve sonrası sayılabilir.

Anti-aliasing ile kullanabileceğimiz fontlar ile anti-aliasing yapılandırması için /usr/local/etc/fonts/local.conf dosyası kullanılır. Bu dosya sisteminizde yoksa oluşturun. Varsa uygun şekilde düzenlemek gerekecektir. Bu dosyanın düzenlenmesi konusunda ayrıntılı bilgi fonts-conf kılavuz sayfasında yer almaktadır.

Dosyanın XML formatında olması gerekiyor. XML formatlı olmasının sakıncası alıştığımız temiz metin dosyalarının aksine imla hatalarından kaynaklı yapılandırma hatalarının başımızı ağrıtacak olmasıdır.

XML dosyaları HTML dosyalarının yapısına banzerdir. Yukarıda adı geçen dosyayı eğer ilk defa oluşturacak iseniz aşağıdakine benzer bir dosya oluşturmanız gerekecektir.

```
<?xml version="1.0"?>
<!DOCTYPE fontconfig SYSTEM "fonts.dtd">
<fontconfig>
  <match target="font">
      <test name="size" compare="less">
          <double>14</double>
      </test>
      <edit name="antialias" mode="assign">
          <bool>false</bool>
      </edit>
  </match>
  <match target="font">
      <test name="pixelsize" compare="less" qual="any">
          <double>14</double>
      </test>
      <edit mode="assign" name="antialias">
          <bool>false</bool>
      </edit>
  </match>
  <match target="pattern" name="family">
      <test qual="any" name="family">
          <string>fixed</string>
      </test>
      <edit name="family" mode="assign">
         <string>mono</string>
      </edit>
  </match>
  <match target="pattern" name="family">
      <test qual="any" name="family">
          <string>console</string>
      </test>
      <edit name="family" mode="assign">
          <string>mono</string>
      </edit>
  </match>
  <match target="pattern" name="family">
       <test qual="any" name="family">
           <string>mono</string>
       </test>
       <edit name="spacing" mode="assign">
           <int>100</int>
       </edit>
```

```
        </match>
        <match target="pattern" name="family">
            <test qual="any" name="family">
                <string>Helvetica</string>
            </test>
            <edit name="family" mode="assign">
                <string>sans-serif</string>
            </edit>
        </match>
        <match target="font">
            <test qual="all" name="rgba">
                <const>unknown</const>
            </test>
            <edit name="rgba" mode="assign">
                <const>rgb</const>
            </edit>
        </match>
    </fontconfig>
```

Daha önce de belirttiğimiz gibi /usr/local/lib/X11/fonts/ ve ~/.fonts/ altındaki tüm fontlar yapılandırma dosyası ile Xft destekli uygulamalarca kullanılabilir. Eğer bu dizinler dışında kalan bir dizindeki fontları kullanmak istiyorsanız, bu durumda /usr/local/etc/fonts/local.conf dosyasına aşağıdakine benzer bir satır eklemeniz gerekir.

```
<dir>/fontların/bulunduğu/dizin/</dir>
```

Yeni fontların bulunduğu dizini eklediğinizde font tampon belleğinin tazelenmesi gerekir.

```
# fc-cache -f
```

Yukarıdaki yapılandırma dosyasını incelediğinizde 14 pixelden daha küçük olan metinlerdeki anti-aliasing kullanımının söz konusu olmadığını göreceksiniz. Daha anlaşılır olması için ilgili bölümü aşağıda yeniden yazdık.

```
        <match target="font">
            <test name="size" compare="less">
                <double>14</double>
            </test>
            <edit name="antialias" mode="assign">
                <bool>false</bool>
            </edit>
        </match>
        <match target="font">
            <test name="pixelsize" compare="less" qual="any">
                <double>14</double>
            </test>
            <edit mode="assign" name="antialias">
                <bool>false</bool>
            </edit>
        </match>
```

Benzer biçimde bazı monospace fontlar da anti-aliasing ile düzgün görüntülenemeyebilir. Özellikle de KDE kulanıyorsanız bu durum ile karşı karşıya kalabilirsiniz. Bunun için uygulanabilecek olan bir çözüm monospace fontları 100 pixel olarak kullanmak olabilir.

Yukarıdaki yapılandırma dosyasındaki ilgili bölüm aşağıda verilmiştir.

```
        <match target="pattern" name="family">
            <test qual="any" name="family">
                <string>fixed</string>
            </test>
            <edit name="family" mode="assign">
                <string>mono</string>
            </edit>
        </match>
        <match target="pattern" name="family">
            <test qual="any" name="family">
                <string>console</string>
            </test>
            <edit name="family" mode="assign">
                <string>mono</string>
            </edit>
        </match>
```

Helvetica gibi bazı fontlar, anti-aliasing ile istenen sonuçları vermeyecektir. Fontlar yarıdan kesilmiş gibi görüntülenebilir. Bazen Firefox'un çakılmasına da neden olabilir. Bu sorunun önüne geçebilmek için local.conf dosyasına aşağıdaki bölümün eklenmesi gereklidir. Yukarıdaki dosya örneğinde bu bölüm bulunmaktadır.

```
<match target="pattern" name="family">
    <test qual="any" name="family">
        <string>Helvetica</string>
    </test>
    <edit name="family" mode="assign">
        <string>sans-serif</string>
    </edit>
</match>
```

X11 ile gelen fontlar anti-aliasing tercih edenler için tatminkar sonuçlar vermemektedir. Bu iş için en uygun fontlar x11-fonts/bitstream-vera portunda yer alanlardır. Port üzerinden kurulum yapıldığında /usr/local/etc/fonts/local.conf dosyası bulunmuyorsa otomatik olarak oluşturulur. Eğer dosya varsa /usr/local/etc/fonts/local.conf-vera dosyası oluşturulur. Bu iki dosyanın içeriğini birleştirip kullanabilirsiniz. Böylelikle Bitstream fontları otomatik olarak X11 Serif, Sans Serif ve Monospace fontların yerini alacaktır.

Son işlem olarak kullanıcılar kendi kişisel tercihlerini kullanmak istiyorlar ise yukarıdaki font.conf dosyasının içeriğini kendi tercihleri doğrultusunda düzenleyip ~/.fonts.conf dosyası olarak kayıt etmelidirler. Anımsatmakta yarar var; dosya mutlaka XML formatında olmalıdır.

LCD monitör kullanıcıları monitörlerinde fontların görünümünü iyileştirmek için sub-pixel sampling yapabilirler. Bu terimin Türkçe karşılığını bilemediğimiz için olduğu gibi kullandık. Sub-pixel sampling basit olarak kırmızı, yeşil ve mavi bileşenleri ayrıştırarak yatay çözünürlüğü değiştirmektedir. Bunu yapmanız durumunda görüntüde çarpıcı sonuçlar elde edebilirsiniz. Yukarıdaki örnek dosya LCD monitör için hazırlanıldığı için aşağıdaki bölümü barındırmaktadır.

```
<match target="font">
    <test qual="all" name="rgba">
        <const>unknown</const>
    </test>
    <edit name="rgba" mode="assign">
        <const>rgb</const>
    </edit>
</match>
```

Bu örnek bölümde yer alan rgb, bgr, vrgb veya vbgr olarak değiştirilebilir ve tatminkar sonuç veren değer kullanılmalıdır.

Anti-aliasing, X sunucusunun yeniden başlatılması ile etkinleşecektir. Ancak uygulamaların bu desteği vermesi durumunda kullanılabilecektir. QT toolkit ve KDE bu desteği sunmaktadır. GTK+ ve GNOME için de Font ayarları bölümünden gereken ayarlar yapılabilir. Mozilla uygulamaları için ise ayrı bir ayara gerek yoktur.

BSD - VI

Farklı kabuk sistemlerinin kurulumu ve ayarları

Gökşin Akdeniz
goksin@enixma.org

Önceki bölümlerde BSD sistemin kurulumu, farklı kurulum yöntemleri ile X yapılandırmasına yer vermiştik. BSD sisteminizde X olmadan da çalışabilirsiniz. BSD sisteminizde kabuk yani SHELL standart olarak yer alır. Kurulum aşamasında sizden kullanmak istediğiniz kabuğu seçmeniz istenir. Sistem yöneticisi hesabı için C kabuğu standarttır. Diğer kullanııclar için POSIX standartları gereği Bourne Shell - Bourne kabuğu - /bin/sh kullanılır. Ancak istediğiniz başka bir kabuğu da seçerek kullanabilirsiniz.

UNIX Sistemlerde kabuk kullanıcı ile işletim sisteminin kalbi olan çekirdek-kernel arasında yer alan bir arayüzdür. Çekirdeğin temel görevi sistem kaynaklarının paylaştırılması ve kontrolü ile süreçlerin çalıştırılmasıdır. UNIX ve türevi sistemleri tanımayan bir çok kullanıcı için kabuk ortamı "MSDOS" olarak algılanır. Kabuk ortamında çalışan birisi iseniz, omuzunuzun üzerinden baktıklarında "DOS mu" diye de soracaklardır. Hı deyip gülümseyip geçin.

Öte yandan Windows ve Mac kullanıcılarının çalıştıkları grafik arayüzlere aynı açıdan baktığımızda Windows ve Mac sistemlerin grafik arayüzleri de bir çeşit kabuk olarak düşünülebilir. Kullanıcıdan gelen komutları yorumlayıp bunları işletim sistemine ileten bu arayüzler BSD ve diğer UNIX benzeri sistemlerdeki gibi işletim sistemi ile kullanıcı arasında yer alırlar. Kabuk kullanıcıdan gelen komutları makine diline çevirip çekirdeğe iletirken, çekirdekten makina dili olarak gelen bildirimleri de kullanıcının anlayabileceği hale getirmektedir. BSD sisteminize kullanıcı adı ve şifreniz ile giriş yaptığınızda kurulum sırasında seçilmiş olan kabuk çalışır. Bu kabuk ortamı yukarıda kısaca değindiğimiz işlemleri gerçekleştirir.

İşletim sistemi tasarımı açısından bakıldığında çekirdek ve kabuk işletim sisteminin en temel işlevleri olan kaynak kullanımı ve süreçlerin gerçekleştirilmesini sağlayan kod ile kullanıcının doğrudan bu işlevlere erişimini önlenmiş ve kullanıcıların doğrudan kod çalıştırmasının neden olabileceği olumsuzlukların önüne geçilmiş olur. Bu yaklaşım UNIX sistemlerde daha kesinleşmiştir. Kullanıcılar için tanımlanmış olan erişim yetkileri sınırlandırılmıştır. Çekirdek tüm yetkilere sahip iken kullanıcılar karşılaştırıldığında yetkisi yok denecek kadar azdır. Kabuk yetki yelpazesinin iki uç noktası arasında kullanıcı ile sistem arasında yer alır. Kabuğun görevi kullanıcılara güvenli, kontrollü bir şekilde çekirdeğe ve kontrol ettiği donanımlara erişim sağlamaktır.

Kabuğun arada yer alması ile kullanıcıdan gelen komutları yorumlar, dosyalara erişir, çalıştırma vb. süreçlere çekirdek aracılığı ile dönüştürüp kullanıcıya sunar. Kabuk kullanıcı

atafından kontrolsuz olarak çalıştırılacak olan her türlü komut, kod ve benzerlerinin sisteme vereceği zararın önüne geçer. Bunun dışında sistem üzerinde çalıştırılabileceğiniz koda erişim sağlar.

UNIX sistemlerde kabuk/shell, komut satırı olarak yorumlanır. Komut satırının algılanması kabuğun bir program olarak düşünülmemesinden kaynaklanır.Kabuk ortamında çalışırken komutlar ve ilgili seçeneklerini kullanırız. Yazılan bir metindir ve yine sistem tarfından geri döndürülen yine bir metindir. Birden çok kabuk kullanmak olası olduğu için bir kabuğu başka bir kabuktan ayırmak son derece güçtür. Ancak bu farklar kabukta çalışırken gözlenebilir. Bazı durumlarda kritik öneme sahip olur.

Kabuk Seçimi

BSD sistemlerde tek bir kabuk ile sınırlı olmadığınızdan kabuk seçimini kullanıcıları tanımlarken yapmış olsanız da sonradan kendi tercihleriniz doğrultusunda kabuğunuzu değiştirebilirsiniz. Bu kabuklardan bazıları girdiğiniz komutu eksiksiz ve hatasız olarak girmenizi bekler, düzeltmenize olanak vermez ve baştan yazdırır. Bazıları ise hatalı girişinizi yeniden çağırıp üzerinde değişiklik yapmanıza olanak verir. Bazıları ise komutu TAB tuşu ile bir dizi olası seçenek arasından seçim yaparak tamamlamanıza olanak verir. Port üzerinde çok sayıda kabuk bulunduğu için bu yazıda hepsine yer vermemiz olanaklı değil. Burada adı geçen kabuklara /usr/ports/shells altından erişebilirsiniz.

Bourne Kabuğu (sh)

İlk kurulum aşamasında eğer kabuk seçimi yapmadıysanız varsayılan tanımladığınız kullanıcı için varsayılan kabuk Bourne kabuğudur. Bourne kabuğu en yaygın olarak kullanılan en eski kabuklardan birisidir. Bell Laboratuvarları'nda Steven Bourne tarafından AT&T'nin geliştirmekte olduğu UNIX için yazılmıştır. FreeBSD, ve diğer birçok UNIX türevi sistemler daha sonra Bourne kabuğunu, gelişmiş bir sürümü olan ve POSIX kabuğu olarak da bilinen sürümü ile değiştirmiştir. POSIX kabuğu, halefi olan Bourne'a göre yeni özelliklere ve işlevlere sahiptir.

Öte yandan sh bir çok kullanıcının günlük işlerini yapması için yetersiz kalmaktadır. Kullanıcıların daha fazla işleve sahip olan bir kabuk ortamını tercih etmeleri olağandır. sh kabuğunun halen kullanılmasındaki neden esnek yapısından kaynaklanmaktadır. sh kabuğu sadece komut yazmak için değil, aynı zamanda bir çok komutun birlikte kullanılmasına da olanak verir. Enixma'nın eski sayılarında yer alan kabuk programlama dizisinde görebileceğiniz gibi birçok komutun birlikte kullanıldığı programları yorumlayabilir. Böylelikle sh ile birçok karmaşık yönetimsel işleri kolaylıkla yapabilirsiniz. Kabuk programları ile birlikte değişken kullanımı, eğer sorguları, döngüler vb. birçok programlama dilinde kullanılan birçok özelliği kullanabilirsiniz.

C Kabuğu (csh)

C kabuğu, BSD'nin asıl kabuğudur. Berkeley'deki geliştiriciler UNIX'in asıl kabuk ortamı olan Bourne kabuğunun kısıtlarını aşmak için BSD'yi geliştirirken C kabuğunu de geliştirmiştir. C kabuğu olarak adlandırılmasının nedeni kabuğun söz diziminin C programlama diline benzemesindendir. C kabuğu komut tamamlama, etkileşimli kullanım ve süreç kontrolü ve benzerleri gibi özelliklere sahiptir. Bourne kabuğunda bu özellikler yoktur. Bugünkü BSD sürümlerinde C kabuğu root hesabı için standart olmakla birlikte tcsh kabuğuna verilmiş bir bağ olarak karşınıza çıkabilir.

Korn Kabuğu (ksh veya pdksh)

Berkeley geliştiricilerinin BSD ile sundukları C kabuğuna karşılık olarak AT&T Laboratuvarları'ndaki geliştiriciler 1986 yılında Korn kabuğunu duyurmuşlardır. David Korn tarafından geliştirilmiştir. Korn

kabuğu geriye doğru uyumlu olarak yani Bourne kabuğu ile uyumlu olacak şekilde tasarlanmıştır. C kabuğunda olduğu gibi süreç kontrolü, komut geçmişi, uzun komutlar için kısaltmalar ile C kabuğunda olduğu gibi çağrılan alt kabuklar için yapılandırma dosyası da sunar. Korn kabuğu aslında C kabuğunun sunmadığı ek özelliklere da sahiptir. Kabuk programları yazan programcılar için yeni komutlar ile diğer kabuklarda kullanılan söz dizimini de desteklemektedir. Korn kabuğu bu özellikleri nedeni kısa zamanda UNIX sistemlerde standart kabuk olarak kullanılmıştır, özellikle de ticari UNIX sistemlerde. pksdh ise kamu malı -public domain- olarak nitelendirebileceğimiz ve BSD sistemlerde kullanabileceğimiz kabuktur.

Bourne Again Shell (bash)

Bourne Again shell veya bash, Bpurne kabuğunun Free Software Foundation tarafından temelden yeniden yazılmış olan ve sh ile uyumlu olan kabuktur. Linux sistemlerde standart olarak gelir. Korn kabuğundaki gibi Bourne kabuğu ile uyumlu olduğu için Bourne kabuğu için yazılmış olan kabuk programlarını destekler. Korn kabuğuna benzer olmakla birlikte yeni birçok özelliğe sahiptir. Bunlar içerisinde yardım özelliği, komut satırında düzenleme yapabilmek, komut geçmişini kullanma ve çağırma ile diğer kabuklara göre daha çok sayıda ortam değişkeni kullanabilmesi sayılabilir.

BASH bu kadar becerikli olmasına karşın ticari UNIX Sistemlerde kendisine yer bulamamıştır. Eğer BASH için yazılmış bir kabuk programını bir başka UNIX sistemde çalıştırmak isterseniz sorunlar ile karşı karşıya kalacağınız açıktır.

tcsh Kabuğu

tcsh kabuğu, C kabuğunun geliştirilmiş halidir. t harfi ise DEC PDP-10 sistemlerin işletim sistemi olan TENEX'e atıfta bulunulduğu için kullanılır. tcsh komut satırının işleyişinin TENEX komut satırına benzetilmesinden dolayıdır. C kabuğunda olmayan birçok yeni özelliğe sahiptir. Dosya adı tamamlama, komut satırında düzenleme yapma, BASH'ta olan komut özellikleri kullanılabilir. Hatta BASH'ta olmayan bazı özelliklere da sahiptir. Yazdığınız komutun olası olumsuz sonuçları için sizi uyaracaktır. "Tüm bu dosyaları silmek istediğinizden emin misiniz?" gibi bir soru aslında tcsh kaynaklıdır. Microsoft ise sadece kopyalamış ve daha da ünlü yapmıştır.

Eğer etkileşimli kabukları kullanmayı tercih ediyorsanız, tcsh doğru seçim olacaktır. Ancak atası olan C kabuğunun bazı tatsız özelliklerini de barındırmaktadır. csh kabuk programlamadaki sıkıntıları bu kabuk için de söz konusudur. Kabuk programlamadaki kısıtlarının nedeni yaygın olarak kullanılmamaktadır.

zsh Kabuğu

Z kabuğu veya zsh olarak da bilinir. Geliştirme sürecinde türünün en iyisi olması amaçlanmıştır. Diğer kabuklarda beğenilen özellikler bünyesinde toplanmıştır. Korn benzeri bir tasarıma sahiptir. Bash ve tcsh özelliklerini barındırır. Bu nedenle fazla bellek kullanır. Emacs tuş kombinasyonlarına benzer tış kombinasyonlarını desteklemesi en ayırt edici özelliğidir. Eski "sıkı UNIX kullanıcıları" tarafından iyi bilinen ve kullanılan kabuktur.

Hangi Kabuğu Kullanmalıyım?

Aslında standart BSD komutları söz konusu olduğunda kabuk önemini yitirmektedir, zira hepsi aynı şekilde çalışacaktır. Ancak sistem yönetimi, ortam değişkenlerinin ayarlanması vb ileri düzey işlemlerde kabuklar arası farklar önem kazanmaktadır. Önceden yazdığınız komutları ok tuşları ile çağırmek ve dosya isimlerini TAB tuşu ile tamamlamak vb işlemler kolaylık sağladığı için günlük kullanım söz konusu olduğunda pdksh, bash veya zsh tercih edilebilir. Kabuk programlama ile ilgileniyorsanız bash veya pdksh daha uygun olacaktır.

Yeni Kabuk Kurulumu

Kurulum sırasında kullanıcı hesaplarına atanan kabukları değiştirmediyseniz normal kullanıcılar sh, root ise csh kullanacaktır. GNU/Linux kullanıcısı iseniz standart olarak bash kullanmışsınızdır. BSD ile Linux arasındaki farkın kökeninde BSD ile System V -GNU/Linux'un fazlasıyla benzeridir- tasarım prensipleri yatmaktadır. Bugün için bu farklar çok önemli değildir. Ticari UNIX sistemleri ilk sürümlerinden bugüne dek değişmemiş, bazısı GNU/Linux bazısı BSD olarak bugüne dek gelmiştir. Mimarideki farklılıkları bir kenara bırakırsak asıl farkın lisansta ortaya çıktığını görebiliriz. Linux dağıtımlarının bash ve BSD sistemlerin csh ve sh tercih etmesinin temel nedeni tercih edilen lisanslardır.

BASH GNU kökenlidir. Linux dağıtımları GNU araçlarını ve lisansını kullandıkları için BASH standart kabuk olarak seçilmiştir. csh,sh ve bash bazı açılardan benzer olmakla birlikte işleyişleri farklıdır. Yapılandırma ve işleyişteki farklar nedeni ile birbirlerinden kesin olarak ayrılırlar.

BASH, kurulum sırasında standart olarak X ve diğerlerini seçtiyseniz kurulmuştur. Ancak kabuk ayarınız sh olabilir. Yukarıda adı geçen kabukların bazıları standart olarak kurulmuş demektir.

Kabuk üzerinde çalışırken en önemli olan dosya /etc/shells dosyasıdır. Bu dosya kurulu olan kabukları ve yollarını tanımlar. Aşağıdaki örnek standart bir FreeBSD 7.2 kurulumundaki kabuk dosyalarını göstermektedir.

```
# cat /etc/shells
# $FreeBSD: src/etc/shells,v 1.5.34 2009/04/14 03:14:15
kensmith Exp $
#
# List of acceptable shells for chpass(1).
# Ftpd will not allow users to connect who are not using
# one of these shells.

/bin/sh
/bin/csh
/bin/tcsh
/usr/local/bin/bash
/usr/local/bin/rbash
```

Bu dosyanın amacı "geçerli" kabukları tanımlamaktır. chsh programı ile normal bir kullanıcı /etc/shells içinde tanımlı olmayan bir kabuğu kullanamaz. Bu sınırlamanın nedeni kullanıcının sisteme giriş yaptığında kendisi için tanımlanmış olan giriş kabuğu dışındaki bir programı çalıştırmasının önüne geçilmesidir. Özellikle de setuid ve kullanıcının kendi UID'sine sahip olan uygulamaların çalıştırılmasının önlenmesini sağlar. setuid ile root'a ait olan bir uygulama root yetkileri ile çalıştırılabilir. Normal bir kullanıcının sisteme giriş yaptığında root yetkileri ile uygulamaları çalıştırılması istenmeyen bir durumdur.

Tüm kullanıcılara ait kabuk bilgileri /etc/master.passwd kullanıcı veritabanında saklanır. Bu dosyada kullanıcıya ait olan bilgilerin yer aldığı satırın 10'ncu elemanı tanımlı kabuktur.

```
goksin:$1$p1lQG5SL$kmOqSYZOWn4ruaLjFuUk/0:1001:1001::0:0:Goksin Akdeniz:/home/goksin:/usr/local/bin/bash
```

kullanıcı için tanımlı olan kabuk BASH ve yolu da /usr/local/bin/bash olarak /etc/shells dosyasında tanımlıdır. Bir kabuk kurduğunuzda /etc/shells dosyasını düzenlemeniz gerekmez. Kurulan kabuk otomatik olarak bu dosyaya eklenir. Bu dosya ayrıca sisteme sonradan eklenecek olan kullanıcılara kabuk atanırken kullanılabilecek olan kabukların listesini de sunar.

Farklı Kabuklarda Çalışmak

Kullanıcılar sistemde tanımlı olan kabuklardan herhangi birisi ile çalışabilirler. Sisteme giriş yapıldığında tanımlı olan kabuk aktif iken

çalışmak istediğiniz kabuğu çağırıp çalışmaya devam edebilirsiniz. Örneğin standart olarak tanımlı kabuk sh iken BASH'a geçiş yapalım.

```
$ bash
bash-4.0.24 $
```

Bu durumda iken çıkış yapmak için iki defa exit komutunu vermeniz gereklidir. Birincisi BASH kabuğundan çıkış için, ikincisi de kullanıcı hesabınız ile sisteme giriş yaptığınız kabuktan çıkış yapmak içindir.

Kabuğu Kalıcı Olarak Değiştirmek

Sisteme giriş yağtığınızda farklı bir kabuğa geçiş yapmak için kullanacağınız kabuğu çalıştırmanıza gerek yoktur. Bunu ev dizininizde bulunan, giriş yaptığınızda çalıştırılan .login dosyasını düzenleyerek yapabilirsiniz. Bu durumda giriş yaptığınız kabuğun içinden bir diğer kabuk çalıştırıyorsunuz demektir. /etc/shells dosyasında tanımlı olan kabuklardan birisini seçerek kabuğu değiştirebilirsiniz. chsh (change shell) uygulaması bu amaçla yazılmıştır. chsh farklı sistemlerde de kullanılmaktadır. Ancak işleyişi farklı olabilmektedir. chsh ile chpass kullanıcı yapılandırması için kullanılır. root olarak kullanılması durumunda kullanıcı yapılandırmasını değiştirebileceğinizi unutmayın. Ortam değişkenlerinde EDITOR değişkeni ile tanımlı olan düzenleyici ile kullanıcı bilgilerini değiştirebilirsiniz. Standart olarak bu düzenleyici vi'dır.

```
#chsh goksin
#Changing user database information for goksin
Shell: /bin/sh
Full Name: Goksin Akdeniz
Office Location:
Office Phone:
Home Phone:
Other information:
~
~
:q!
```

Öte yandan normal kullanıcı hesabınızdan kullanırsanız sadece kabuk değişecektir. Kullanıcı şifrenizi yazın ve kabuğunuzu değiştirin.

```
$ chsh goksin
#Changing user database information for goksin
Shell: /usr/local/bin/bash
Full Name: Goksin Akdeniz
Office Location:
Office Phone:
Home Phone:
Other information:
~
~
~
~
:wq
```

Sistemde tanımlı bir kullanıcı giriş yaptığında bir kabuk çalıştırmayabilir. Örneğin /sbin/nologin bu amaçla hazırlanmış bir uygulamadır. Çalıştığında bir mesaj görüntüleyip giriş eylemini sonlandırır. Bunun gibi bazı uygulamaları kullanarak sisteme giriş yapan bir kullanıcının doğrudan tanımlı olan uygulama ile kaşılanması söz konusu olabilir. Bir başka örnekte /usr/bin/mail çalıştırılması ile kullanıcının sadece kabuk ortamında postayı okuması sağlanabilir. Bu uygulamaların dışında kendi hazırladığınız bir uygulama da kullanılabilir.

chsh uygulaması ile kabuk olmayan bir uygulamayı tanımladığınızda düzenlediğiniz kullanıcıya ait bilgiler veritabanına kayıt edilecektir. Ama kullanıcı sisteme giriş yapamayacaktır.

Burada giriş kabuklarının -login shell- çalışması üzerinde durmakta yarar var. Sisteme giriş işlemi yapıldığında giriş kabuğunuz çalıştırılır ve bir pty (pseudo-terminal) atanır. Bu terminal giriş kabuğu sona erilinceye dek giriş yapan kullanıcıya atanmış olur. Eğer giriş kabuğu çalıştırılamıyor ise pty ataması gerçekleşmez. Kullanıcıya ise bir hata mesajı döndürülür. kullanılan terminala bağlı olarak bu hata mesajı

değişkenlik gösterecektir.

BSD'de kullanıcı adı ve şifresi alındıktan sonra kullanıcıya sistem yöneticisi tarafından yapılandırılmış olan /etc/motd döndürülür. (motd mesaage of the day - günün mesajı) Bazen kurallara aykırı davranan kullanıcıları nazikçe uyarmak için bunu kullanabilirsiniz. Bu aynı zamanda /sbin/nologin gibi kabuk olmayan uygulamalar için de geçerlidir. /sbin/nologin kullandığınızda normal giriş süreci işleyecektir. Kullanıcıya ilgili mesaj döndürülecek ve giriş yapılan pty serbest kalacaktır. /sbin/nologin BSD sistemde kullanabileceğiniz tüm terminallerde çalışabilir. SSH'tan seri terminallere kadar hepsinde...

Kabuk Yapılandırma Dosyaları

Sisteme giriş yaptığınızda çalışan kabuğu kendi tercihlerinize göre şekillendirebilirsiniz. Giriş yaptığınızda bir uygulamayı çalıştırabilirsiniz, ortam değişkenlerini kendi tercihlerinize göre değiştirebilir, bu düzenlemeyi sistem veya kullanıcı genelinde yapabilirsiniz. BSD'de kullanılan kabuklar farklı yapılandırmalara sahip oldukları için sistem genelinde ve kullanıcı özelinde olmak üzere BSD sistemlerde iki farklı düzeyde yapılandırılabilir. Bu yapılandırmayı sisteme yeni bir kullanıcı eklendiğinde nasıl gerçekleştiğine bakarak daha kolay kavrayabiliriz.

Sisteme eklenen her kullanıcı için temel kabuk yapılandırması /usr/share/skel dizinindeki dosyalarda yer alır. Eğer isterseniz bu dosyaların bir kopyasını alıp /usr/local/share/skel altına kopyalayabilirsiniz. Üzerinde kendi tercihinize göre düzenleme yaptıktan sonra /etc/adduser.conf dosyasını kendi yapılandırmanızı kullanacak biçimde düzenleyerek bu dosyaların esas alınmasını sağlayabilirsiniz. Bu şekilde sisteme ekleyeceğiniz her kullanıcıya kendi belirlediğiniz yapılandırmayı kullanmasını sağlayabilirsiniz.

```
[goksin@droideka /usr/share/skel]$ ls
dot.cshrc   dot.login_conf    dot.mailrc  dot.rhosts
dot.login   dot.profile dot.shrc
```

```
[goksin@droideka /usr/share/skel]$
```

csh Dosyaları: .cshrc, .login, ve .logout

BSD sistemdeki csh için genel yapılandırma dosyaları /etc/csh.cshrc dosyasıdır. Giriş işlemi ardından yapılacak olan işlemler ise /etc/csh.login altında yer alır. Bu dosyalara göz atacak olursanız bunların # ile yorum satırına dönüşütürüldüğünü görebilirsiniz. Eğer bu dosyalardaki # işaretlerini kaldırıp kayıt ederseniz sistem genelindeki csh yapılandırmasını değiştirmiş olursunuz. csh kullanan tüm kullanıcılar bundan etkilenecektir. Ancak kullanıcılar kendi ev dizinlerindeki dosyalarda düzenleme yapmışlar ise bu durumda kendi yapılandırmaları geçerli olacaktır. Bu dosyalar kullanıcı sisteme başarılı bir giriş yaptığında temel yapılandırma dosyalarından sonra okunarak gereken işlemler yapılır.

cshrc dosyası temel yapılandırma seçeneklerini sunar.

```
# $FreeBSD: src/etc/csh.cshrc,v 1.3.54.1 2009/04/15 03:14:26
kensmith Exp $
#
# System-wide .cshrc file for csh(1).
```

cshrc.login dosyası ise csh giriş yapılandırmasını belirler.

```
# $FreeBSD: src/etc/csh.login,v 1.21.28.1 2009/04/15 03:14:26
kensmith Exp $
#
# System-wide .login file for csh(1).
# Uncomment this to give you the default 4.2 behavior, where
disk
# information is shown in K-Blocks
# setenv BLOCKSIZE      K
#
# For the setting of languages and character sets please see
# login.conf(5) and in particular the charset and lang
options.
# For full locales list check /usr/share/locale/*
```

```
#
# Read system messages
# msgs -f
# Allow terminal messages
# mesg y
setenv LANG tr_TR.ISO8859-9
setenv MM_CHARSET ISO-8859-9

# Denenecek olan
# setenv LANG tr_TR.UTF-8
# setenv MM_CHARSET UTF-8
```

Çıkış yapılandırması ise csh.logout dosyası tarafından kontrol edilir.

```
# $FreeBSD: src/etc/csh.logout,v 1.3.54.1 2009/04/15 03:14:26
kensmith Exp $
#
# System-wide .logout file for csh(1).
```

cshrc dosyası ile csh temel yapılandırması ve ilgili tüm değişkenler tanımlanır. Örneğin aşağıdaki satır uygulamaların bulunduğu dizinlerin yolunu tanımlamaktadır.

```
set path = (/sbin /bin /usr/sbin /usr/bin /usr/games
/usr/local/sbin /usr/local/bin /usr
/X11R6/bin $HOME/bin)
```

Ortam değişkenlerini kendi tercihlerinize göre düzenleyebilirsiniz. Aşağıda kendi ev dizininizde bulabileceğiniz .cshrc dosyasının bir örneği görülmektedir.

```
# $FreeBSD: src/share/skel/dot.cshrc,v 1.14.8.1 2009/04/15
03:14:26 kensmith Exp $
#
# .cshrc - csh resource script, read at beginning of
execution by each shell
#
# see also csh(1), environ(7).
#
```

```
alias h         history 25
alias j         jobs -l
alias la    ls -a
alias lf    ls -FA
alias ll    ls -lA

# A righteous umask
umask 22

set path = (/sbin /bin /usr/sbin /usr/bin /usr/games
/usr/local/sbin /usr/local/bin $HOME/bin)

setenv      EDITOR      vi
setenv      PAGER more
setenv      BLOCKSIZE   K

if ($?prompt) then
      # An interactive shell -- set some stuff up
      set filec
      set history = 100
      set savehist = 100
      set mail = (/var/mail/$USER)
      if ( $?tcsh ) then
            bindkey "^W" backward-delete-word
            bindkey -k up history-search-backward
            bindkey -k down history-search-forward
      endif
endif
```

Ev dizininizde yer alan .login dosyası giriş işleminin ardından okunur. Bu dosyaya sisteme giriş yaptığınızda çalıştırılmasını istediğiniz uygulamaları ekleyebilirsiniz. Standart olarak fortune bu dosyada yer alır. /usr/share/skel/dot.login dosyasından alınan yapılandırma gereği fortune uygulamasının bulunup bulunmadığını denetlenip ardından freebsd-tips görüntülenmektedir.

```
# $FreeBSD: src/share/skel/dot.login,v 1.16.34.1 2009/04/15
03:14:26 kensmith Exp $
#
```

```
# .login - csh login script, read by login shell, after
`.cshrc' at login.
#
# see also csh(1), environ(7).
#

[ -x /usr/games/fortune ] && /usr/games/fortune freebsd-tips
```

Bunların dışında kullanıcıların sistemden çıktıklarında gerçekleşecek olan işlemleri de tanımlayabilirsiniz. Bu dosya /etc/csh.logout dosyasıdır. Kullanıcı çıkış yaptıktan sonra geride bıraktığı geçici dosyalar silinebilir vs vs. Varsayılan yapılandırma herhangi bir işlem barındırmamaktadır.

```
# $FreeBSD: src/etc/csh.logout,v 1.3.54.1 2009/04/15 03:14:26
kensmith Exp $
#
# System-wide .logout file for csh(1).
```

Ayrıca çıkış işlemi sırasında eğer varsa kullanıcı ev dizininde yer alan .logout dosyası okunur ve tanımlanmış olan eylemler gerçekleştirilir. Standart olarak bu dosya oluşturulmaz ve kullanıcı ev dizinlerinde yer almaz.

BASH Yapılandırması: .profile, .shrc, and .bash_logout

Normal kullanıcı hesabınız için BASH kullanacak iseniz giriş, yapılandırma ve çıkış işleyişi yukarıdakine benzer şekilde gerçekleşecektir. İlk olarak sistem genelindeki yapılandırma dosyaları işlenecek ardından da kullanıcı ve dizinlerindeki dosyalar işlenecektir. İlk olarak sistem genelinde BASH yapılandırmasını sağlayan /etc/profile dosyası okunur ve işlenir.

Aşağıda /etc/profile dosyasının standart içeriği görülmektedir. En altta yer alan satırlar yerelleştirme ayarlarıdır ve sonradan eklenmiştir.

```
# $FreeBSD: src/etc/profile,v 1.14.28.1 2009/04/15 03:14:26
kensmith Exp $
#
# System-wide .profile file for sh(1).
#
# Uncomment this to give you the default 4.2 behavior, where
disk
# information is shown in K-Blocks
# BLOCKSIZE=K; export BLOCKSIZE
#
# For the setting of languages and character sets please see
# login.conf(5) and in particular the charset and lang
options.
# For full locales list check /usr/share/locale/*
# You should also read the setlocale(3) man page for
information
# on how to achieve more precise control of locale settings.
#
# Read system messages
# msgs -f
# Allow terminal messages
# mesg y

# Eklenen yerel ayarı

export LANG=tr_TR.UTF-8
export MM_CHARSET=tr_TR.UTF-8
```

/etc/profile dosyası csh.profile ve csh.login dosyalarının işlevlerini tek bir dosyada toplamaktadır. Sistem genelinde BASH ile ilgili tek yapılandırma dosyası budur. BASH kabuğundan çıkış için tanımlanmış olan bir logout dosyası bulunmamaktadır.

Ev dizinindeki .profile dosyası kişisel hesaplar için yapılandırmaları barındırır. En sonda yer alan gpg satırları sonradan eklenmiştir.

```
# $FreeBSD: src/share/skel/dot.profile,v 1.22.8.1 2009/04/15
03:14:26 kensmith Exp $
#
# .profile - Bourne Shell startup script for login shells
#
```

```
# see also sh(1), environ(7).
#

# remove /usr/games if you want
PATH=/sbin:/bin:/usr/sbin:/usr/bin:/usr/games:/usr/local/sbin:
/usr/local/bin:$HOME/bin; export PATH

# Setting TERM is normally done through /etc/ttys.  Do only
override
# if you're sure that you'll never log in via telnet or xterm
or a
# serial line.
# Use cons25l1 for iso-* fonts
# TERM=cons25;     export TERM

BLOCKSIZE=K;       export BLOCKSIZE
EDITOR=vi;         export EDITOR
PAGER=more;        export PAGER

# set ENV to a file invoked each time sh is started for
interactive use.
ENV=$HOME/.shrc; export ENV

[ -x /usr/games/fortune ] && /usr/games/fortune freebsd-tips

#gpg-agent startup

eval $(gpg-agent --daemon)
```

Sondan ikinci satır .shrc düsyası için ENV değişkenini atar ve bunu kullanıcının ev dizininde de yapar. Bu daha sonraki sırada .profile ile okunur. Bunun amacı kullanıcı bekleme işaretinin özelleştirilmesi gibi diğer birkaç seçeneğin yanısıra benzerleri gibi komut 'alias'larını da belirlemektir.

```
# some useful aliases
alias h='fc -l'
alias j=jobs
alias m=$PAGER
alias ll='ls -laFo'
alias l='ls -l'
alias g='egrep -i'

# # be paranoid
# alias cp='cp -ip'
# alias mv='mv -i'
# alias rm='rm -i'

# # set prompt: ``username@hostname$ ''
# PS1="``whoami`@`hostname | sed 's/\..*//'`"
# case `id -u` in
#       0) PS1="${PS1}# ";;
#       *) PS1="${PS1}$ ";;
# esac

# search path for cd(1)
# CDPATH=.:$HOME
```

Ortam Değişkenlerini Tanımlamak

BSD'de csh, sh, BASH ve diğer kabuklar kullanılabilir. Bu kabuklar arasındaki fark yapılandırmaları kadar, söz dizimi, işlevler ve kullandıkları ortam değişkenleri arasında bulunmaktadır. Hangi kabuğu kullanırsanız kullanın hepsi ortam değişkenlerini - environment variables- kullanır ve bu değişkenlere atanmış olan değerlere göre çalışırlar. Bu değerler çalıştırdığınız uygulamaların davranışını belirler. Örneğin EDITOR değişkeni standart metin düzenleyicisini; TERM, ekranda konsolda metnin nasıl görüntüleneceğini; BLOCKSIZE ise du ve df gibi uygulamalarda boyutların KB cinsinden gözterilmesini sağlar. Bu değişkenler sadece giriş kabukları ile sınırlı değildir, aynı zamanda bir kabuktan çalıştırılan bir programın kendi yarattığı kopyaları ve diğer uygulamalarınca da kullanılır.

BASH kabuğunda normal kullanıcı hesabında iken printenv komutu aşağıdaki çıktıyı döndürmektedir:

```
MM_CHARSET=tr_TR.UTF-8
DM_CONTROL=/var/run/xdmctl
```

```
GPG_AGENT_INFO=/tmp/gpg-siEuo2/S.gpg-agent:888:1
TERM=xterm
SHELL=/usr/local/bin/bash
DESKTOP_STARTUP_ID=
XDG_SESSION_COOKIE=8fe105fb073cc29c21aa47664a0d569b-
1243401168.549865-848987023
XDM_MANAGED=method=classic
KONSOLE_DBUS_SERVICE=:1.36
GS_LIB=/home/goksin/.fonts
WINDOWID=52430082
KDE_FULL_SESSION=true
USER=goksin
ENV=/home/goksin/.shrc
SESSION_MANAGER=local/freebsdpmyo.anadolu.edu.tr:/tmp/.ICE-
unix/944
PAGER=more
FTP_PASSIVE_MODE=YES
XDG_CONFIG_DIRS=/usr/local/kde4/etc/xdg::/etc/xdg:/usr/local/e
tc/xdg:/usr/local/etc/xdg/xfce4
PATH=/usr/local/kde4/bin:/sbin:/bin:/usr/sbin:/usr/bin:/usr/ga
mes:/usr/local/sbin:/usr/local/bin:/home/goksin/bin
DESKTOP_SESSION=kde
MAIL=/var/mail/goksin
BLOCKSIZE=K
PWD=/usr/home/goksin
EDITOR=vi
KDE_SESSION_UID=
LANG=tr_TR.UTF-8
XCURSOR_PATH=/usr/local/kde4/share/icons::~/.icons:/usr/share/
icons:/usr/share/pixmaps:/usr/X11R6/lib/X11/icons
KONSOLE_DBUS_SESSION=/Sessions/3
COLORFGBG=15;0
HOME=/home/goksin
SHLVL=1
KDE_SESSION_VERSION=4
LANGUAGE=
XCURSOR_THEME=Oxygen_Black
LOGNAME=goksin
XDG_DATA_DIRS=/usr/local/kde4/share::/usr/share:/usr/local/sha
re:/usr/local/share/gnome
DBUS_SESSION_BUS_ADDRESS=unix:path=/var/tmp/dbus-
```

```
BDIV3hbJ7u,guid=4f68bda59da96bc9e5eafddc4a1ccbd4
PROFILEHOME=
DISPLAY=:0
QT_PLUGIN_PATH=/home/goksin/.kde4/lib/kde4/plugins/:/usr/local
/kde4/lib/kde4/plugins/
_=/usr/bin/printenv
```

Ortam değişkenlerini tanımlamak için katı kurallar olmamakla birlikte USER ve PATH gibi değişkenler standarttır. Bunun dışında kendi tanımladığınız değişkenleri de kullanabilirsiniz. Öncelikle değişkeni ve ardından ilgili değeri tanımlamalısınız. Bu işleyiş kabuklar arasında farklılık gösterebilir.

csh için aşağıdaki tanımlama kullanılabilirken

```
# setenv METINEDITOR emacs
```

BASH için ise aşağıdaki gibi bir kullanım mevcuttur.

```
# METINEDITOR=emavs
# export METINEDITOR
```

Değişkenlerin çalışılan kabuğa göre tanımlaması değişkenlik göstere-cektir. Bu nedenle kullandığınız kabuk ortamına ilişkin değişken tanımlaması ve ataması özelliklerini iyi bilmek gerekmektedir. BSD ile kullanabileceğiniz tüm kabukları ve değişkenlerinin kullanılmasını ele almak bu yazının kapsamı dışında kaldığı için burada sona erdiriyoruz. Farklı kabuklar ile çalışmak ve kabuk programları yazmak isteyenler için geçtiğimiz sayılarda yer verdiğimiz BASH kabuğunu temel alan kabuk programlama yazı dizisi yol gösterici olacaktır. Diğer kabuklara ve programlamasına gelecek sayılarda yer vereceğiz.

BSD - VII

Kullanıcılar, gruplar ve izinler

Gökşin Akdeniz
goksin@enixma.org

Kullanıcılar, Gruplar ve İzinler

UNIX işletim sisteminin temel taşlarından birisi kullanıcılar, gruplar ve dosya izinleridir. Birden çok kulanıcı için tasarlanmış olan bir sistem ile tek bir kullanıcı için tasarlanmış olan sistemi karşılaştırdığınızda ilk göze çarpacak olan sistemde tanımlı kullanıcılar, gruplar ve tanımlı olan izinler olacaktır. Çok kullanıcılı sistemler için bu özellik hem gizlilik hem de güvenlik açısından zorunludur. Kullanıcı kendisine ait olan kişisel dosyaları böylece diğer kullanıcılardan saklayabilirken aynı zamanda tanımlı olan yazma, çalıştırma ve okuma izinleri kapsamında sistem kaynaklarına erişebilir. Böyle kullanıcıların sistem kaynaklarına erişimi kadar yapılandırmasına erişimi de kontrol altına alınabilir. Özelliklede kullanıcıların sınırsız yetkilere sahip olarak isteyerek veya bilmeden yapacakları değişiklikler istenmeyen sorunlara neden olabilecektir. Bu konu BSD sistemlerde böyle iken, Linux dağıtımlarında da aynı olmakla birlikte Pardus dağıtımının eski sürümlerinde tüm kullanıcıların sınırsız yetki ile sistemde paket kurma-kaldırma işlemleri yapabilmeleri gibi bir "açık" ile sunulması bu konunun ne kadar önemli olduğunu göstermektedir. Aynı bilgisayarı paylaştığınız diğer kişilerin dün kurmuş olduğunuz istatistik yazılımlarının oyun oynamak için yer açmak amacı ile kaldırılmış olduğunu düşünün... Windows için olağan bir durum ama Linux dağıtımları için kötü bir örnek!

Çok sayıda kullanıcıya sahip olan bir işletim sisteminde her süreç hangi kullanıcıya ait olduğunu tanımlayan bir kullanıcı adı ile eşleştirilmiştir. Bu şekilde kullanıcı ve çalıştırdığı süreçlerin kullanabileceği sistem kaynakları ile yapabilecekleri sınırlandırılmış olur.

Windows sistemler ile çalıştıysanız bu sistemlerin bu özelliklere sahip olduğunu düşünebilir ve hatta birden fazla kullanıcı tanımlayarak sisteme giriş yapmış olabilirsiniz. Bu, özellikle Windows sunucu ailesi için BSD sistemlerdeki kullanıcı, grup ve izin kavramları ile aynı olmayan bir özelliktir. Bu durum Apple Bilgisayarların MAC işletim sistemi için de böyledir.

Söz konusu sistemler multiuser-çok kullanıcılı sistemler oldukları iddasında bulunsalar da durum tersidir. Bu sistemler temelde bir kullanıcıya sahiptir. Tanımlanan kullanıcılar ise aslında birer profildir. Bu profiller ise kulanıcının tercihlerine göre dosya sistemi ve sair verinin kullanıcıya sunulmasını yani kullanıcı tercihlerine göre gösterilmesinden başka bir şey değildir. Sistem yapılandırmasının değiştirilmesi -dosyaya okuma-yazma ve çalıştırma izinleri verilmiş- yani programların kurulması ve kaldırılması yanında sistemin kapatılması da dahil birçok "olağan" işlemin yapılabilmesi aslında bir tane kullanıcı olmasından kaynaklanmaktadır. Bu kullanıcının sistem üzerinde mutlak egemenliği söz konusudur. Windows NT ve sonrasındaki

sunucu sistemler ise BSD sistemlerde olduğu gibi veri ve sistem güvenliği konusundaki gereksinmelerden kaynaklı olarak çok kullanıcılı bir işletim sisteminin tasarımı esas alınarak geliştirilmiştir ancak yine de BSD sistemleri ile karşılaştırılamayacak kadar konuya uzak sistemlerdir.

Sistemde tanımlı olan tüm kullanıcılar kendilerine ait olan bir kullanıcı hesabına sahiptir. Kullanıcılar bunun yanında sistem kaynaklarına erişimlerini düzenleyen izinlere sahiptir. Windows ve BSD sistemlerinde sistemin tamamı üzerinde kontrolü olan bir sistem yöneticisi hesabı bulunur. Her iki sistemde de gruplar ve farklı erişim düzeyleri bulunur. Gerçi Windows içinde bu erişim yetkilendirmesi farklı boyutlarda ve farklı özelliklerde gerçekleşmetedir. Bir 'domain controller' ile sisteme erişim yetkilieri ile sistemde tanımlı kullanıcının erişim yetkileri farklıdır, yönetimsel yetkilere sahip bir çok hesap bulunmaktadır. Bu karmaşıklık içinde deneyimli bir kullanıcı bile kolaylıkla hata yapabilir. Windows sunucu ailesi sürekli geliştirilmesine rağmen halen UNIX'in çoklu kullanıcı yapısına ve özelliklerine denk duruma gelememiştir. Bu karşılaştırdığımız sistemlerin farklı tasarım ve geliştirme süreçlerine sahip olmasına da bağlıdır. Bu nedenle söz konusu sistemlerde yapısal olarak büyük farklılıklar ortaya çıkmaktadır.

Windows ailesinde sunucu sistemler makinanın başında oturup yönetilmez. Bunun yerine Windows sistemlerin uzaktan yönetilmesini sağlayan uzakten erişim ve yönetim uygulamaları kullanılır. Bu uygulamalar her ne kadar grafik bir arayüz sunarak yönetimi kolaylaştırmakta olduğu düşünülse de yoğun ağ trafiğinin olduğu durumlarda gecikmelere neden olarak istenmeyen durumlara neden olabilmektedir. Bu grafik arayüzleri bir web tarayıcısı olarak düşünebilirsiniz. Web tarayıcısında bulunan özellikler sizin kullanabileceklerinizdir ve ağ trafiğinde ortaya çıkan dar boğazlar nedeni ile zaman aşımı olacak ve hata döndürülecektir. Bu nedenle istemci-sunucu mimarisi üzerinde geliştirilmiş olan bu uygulamalar yapıları gereği kısıtlamaları da barındırır.

Bu maimarinin olumlu yanları da sözkonusudur. Çalıştırılan uygulama sadece kendisie tanımlanan kaynakları kullanabilir. Örneğin bir e-posta sunucusuna bağlandığınızda sadece e-posta mesajlarınızı okuyabilir, yazabilir, yollayabilir ve bilgisayarınıza indirebilir ve silebilirsiniz. Başka bir eylem gerçekleştiremezsiniz. Sınırlama gereği kullanıcılara ait olan veri uygulama tarafından kontrol edilir ve işletim sistemi bunun için bir işlem yapmaz. Windows sistemlerin çalışma şekli bu şekildedir.

Öte yandan BSD, UNIX ve türevleri sistemlerde işleyiş farklıdır. Her na kadar sunucu-istemci mimarisi ile geliştirilmiş olan uygulamalar bulunsa da UNIX ve türevi sistemlerde uzaktan erişim sıklıkla kullanılır. Bir BSD sistemi uzaktan aynı anda birden fazla kullanıcı tarafından erişilebilerek kullanılabilir. SSH -secure shell- bağlantısı ile kullanıcılar konsol üzerinden sisteme bağlanarak sistemi kullanabilirler. Bu nedenle de BSD sistemlerde kullanıcıların sisteme doğrudan erişebilmelerine olanak verildiği için kulanıcı hesapları, gruplar ve dosya izinleri kullanılarak kullanıcının yetkileri tanımlanır ve sınırılandırılır.

MacOS X işletim sistemi kullanıcıları işletim sistemi olarak Darwin ve Apple geliştiricilerinin hazırladıkları kodları kullandıkları için BSD sistemdeki yapılandırma ile benzerlikler bulunmaktadır. Sürekli olarak root hesabını kullanmak yerine sudo kullanmaları yerinde olacaktır.

Kullanıcılar ve Gruplar

UNIX ve BSD sistemlerde kullanıcılar ve izinler yapı olarak çok basittir. Sisteme göz attığınızda sadece iki tip kullanıcı hesabı görürsünüz. Kullanıcılar ve sistem yöneticisi ya da birebir Türkçe'ye çevirirsek süper kullanıcı/root-super user/root. Kullanıcılar veya bazı kaynaklarda rastlayacağınız gibi normal kullanıcı hesapları sınırlı

yetkilere sahiptir. Sınırsız yetkiye sahip olan kullanıcı ise sistem yöneticisi yani root kullanıcısıdır. Diğer izin sistematikleri örneğin Windows'daki gibi daha karmaşık yapılara sahiptir. (NT soyundan gelenlerde olduğu gibi) Bunun nedeni bazı işlevlerin/süreçlerin sistem genelinde farklı şekilde çalışmasını sağlamak için bunun yapılmasının zorunluluğudur. Bu işlevlere örnek olarak yetkilendirme/authentication ve işletim sistemin tarafından çalıştırılan süreçler sayılabilir.

BSD sistemlerde ise örneğin web sunucusu kurulumu ve yaplandırması daha az yetkilendirme/izin düzenlemesi ile gerçekleştirilebilir. Windows'ta olduğu gibi çok sayıda izin vs. kullanarak da bunu yapabilirsiniz. Ancak karmaşıklık arttıkça sorunlar da büyüyecek, hataların giderilmesi de o derece zor olacaktır. Bu güvenli bir sistem yaratmaktan ziyade güvensiz bir sistem yaratmaya daha elveişli bir durum ortaya çıkaracaktır.

BSD sisteme ister uzaktan bağlanın ister kendi bilgisayarınızdan kullanın, oluşturduğunuz kullanıcı hesabınız ile giriş yaptığınızda her zaman kendi kullanıcı hesabınızın ev dizinine giriş yaparsınız. Bu ev dizinindeki dosyaları istediğiniz gibi düzenleyebilirsiniz. Bu dosyaların sahibi siz olduğunuz için bunu yapabilirsiniz. Ancak sistemdeki diğer dosyalara size verilen izin çerçevesinde erişşebilirsiniz. Ancak ve ancak root olarak sistem genelindeki dosyalar üzerinde sınırsız yetkiye sahip olduğunuzu unutmayın. Bu şekilde tek kullanıcılı işletim sistemlerindeki gibi sorumsuzca haraket edebilirsiniz.

Root kullanıcısı olarak veya root yetkilerini alabilmeniz için "su root" veya "su -" komutunu kullanmanız gerekir. Ancak bunun için öncelikle "wheel" grubuna üye olmanız gereklidir. Aksi halde yetkinizin olmadığını belirten bir mesaj döndürülecektir. BSD sistemlerde normal kullanıcılar ve süper kullanıcı dışında başka kullanıcı olmadığı için, wheel grubu "su root" veya "su-" kullanarak root yetkilerini alabilecek olan özel kullanıcılar grubunu tanımlar.

Burada bir noktaya değinmek gerekiyor. BSD sistemlerde uzaktan erişerek kullanıyor olsanız da sistemin standart yapılandırması gereği root girişine izin verilmez. Tersine normal kullanıcı olarak giriş yaptıktan sonra "su -" veya "su root" ile root yetkilerini kullanabilirsiniz. Tersine bir yapılandırma tehlikelere davetiye çıkaracaktır.

Normal kullanıcı olarak tanımladığımız kullanıcılar arasında da ayrım söz konusudur. Bu ayrım sisteme giriş yapabilen gerçek kulanıcılar ile giriş yapamayan otomat kullanıcılar arasındaki farkı gösterir. Bu otomat kullanıcılar terimi derslerde öğrencilere konuyu daha iyi anlatabilmek için kendi bulduğum bir terim. Otomat kullanıcılar dediğim, bin, operatr, daemon, nobody gibi tanımlı olan kullanıcılardır. Bu kullanıcılar sistemde yer alırlar ama asla sisteme giriş yani login yapamazlar. Bu kullanıcı hesaplarının bulunmasının nedeni bazı sistem süreçlerine ve sunucu uygulamalarına bir kullanıcı atanması gerektiği içindir. Bu zorunlukluk sistemdeki tüm süreçlerin mutlaka bir kullanıcıya ait olması gerektiğindendir. Böylece sözkonusu sürecin diğer süreçler ve dosyalar ile etkileşimi bağlı olduğu kullanıcı hesabının izinlerine bağlı olarak belirlenen sınırlar içinde gerçekleşecektir. Root kullanıcısına ait olarak tüm süreçleri çalıştırmak istenmeyen bir durumdur.

Bu izin yapısından dolayı kulanıcılar kendi dosyalarına dahi doğrudan erişmezler. Kullanıcılar kendi dosyaları üzerinde yaptıkları işlemleri komutlar vererek geçekleştirirler. Bu komutlar ise bir süreç başlatır. Başlatılan süreç ise kullanıcının sahip olduğu izinler çerçevesinde dosya üzerinde işlem yapacaktır. Bir süreç ancak sahibine ait olan süreçler ve dosyalar üzerinde işlem yapabilir. Diğer süreçler ve dosyalar üzerinde işlem yapamaz.

Bu nedenle UNIX ve BSD sistemlerde süreçler kullanıcılaraca kontrol edilir. Normal kullanıcı dışında yukarıda adı geçen otomat kullanıcılar güvenlik için de zorunludur. Eğer bir süreci root kullanıcısına atamış iseniz, bu süreç sistemde sınırsız yetkiler ile çalışacaktır. Söz konusu

sürecin bir yapılandırma dosyasını değiştirdiği ve sistemin standart yapılandırması dışında bir yapılandırmaya giderek farklı kullanıcılara yeni yetkiler tanımlaması durumda bir güvenlik açığı ortaya çıkabilir. Daha da önemlisi kullanıcıya ait olan dosyaların diğer kullanıcılar tarafından okunması, yazılması ve çalıştırılması da söz konusu olabilir. Veya normal bir kullanıcı hesabının sınırsız yetkiler ile kötü amaçlı olarak uygulamaları çalıştırması da söz konusu olabilir. Bu nedenle UNIX ve BSD sistemlerde bu otomat kullanıcılar bulunurlar. Bu kulanıcılara ait olan süreçlerde tanımlanan izinler kapsamında işlerini yaparlar.

Sistemde tanımlı olan tüm kullanıcılar bir grubun üyesidir. Kullanıcı birden çok gurubun üyesi olabilir. Bu gruplardan birisi ise birincil grup -primary group- olarak tanımlanır. Genelde birincil grup kullanıcı adı ile aynıdır. Bu daha esnek ve güvenli bir yapılandırma sağlar. Öte yandan sisteme eklediğiniz kullanıcılarınızı users grubuna da üye yapabilir ve bunu birincil grup olarak tanımlayabilirsiniz.

Kullanıcıların grup üyeliklerini ise root kullanıcısı belirler. Yukarıda sözünü etrtiğimiz wheel grubu kullanıcıların root yetkilerini alıp alamayacağını belirler. Bundan başka, sistemde tanımlı bir çok grup bulunur. Bu grupların kullanılmasının nedeni ise bazı kulanıcılara bazı yetkilerin verilmesini sağlamaktır. Böylelikle bir gruba üye olan kullanıcı o grup için tanımlanmış olan izinleri otomatik olarak kullanabilir. Böylece her bir kullanıcı için ayrı ayrı izin tanımlamasına gerek kalmaz. Bir diğer kolaylığı ise bir kullanıcıya ait olan dosyaya erişmek isteyen kulanıcıya diğer kullanıcının kullanıcı adı ve şifresini vermek durumunda kalmayacağınızdır. Aynı grupta iseler, grup için tanımlı olan izinler kapsamında farklı kullanıcılar dosyaya erişebilirler.

Dosya Sahipliği

İzinler ve gruplar hakkında bilgi verirken dosyaların, süreçlerin sahiplerinden söz etmiştik. tüm UNIX sistemlerde tüm dosyaların bir sahibi bulunur. Tüm dosyalar ve dizinlerin sahibi olan kullanıcılar ve gruplar tanımlıdır. Bir dizin ve dosya üzerinde kulanıcıların erişim yetkileri yani izinleri tanımlıdır. Böylece hangi kullanıcının hangi yetkilere sahip olduğu belirlenir.

Aşağıdaki çıktıda kendi ev dizinimde yer alan bazı dosyalara ait izinler görülüyor.

```
[goksin@droideka:~]$ ls -l 01.06.2009.txt 01.01.2009.txt
sb_ra0409_pdf.zip
-rw-r--r-- 1 goksin goksin   306742 Oca 31 23:58
01.01.2009.txt
-rw-r--r-- 1 goksin goksin    81758 Haz  3 12:10
01.06.2009.txt
-rwxr-xr-x 1 goksin goksin 16409165 Nis 23 22:56
sb_ra0409_pdf.zip
[goksin@droideka:~]$
```

Benzer şekilde test dizini için

```
[goksin@droideka:~]$ ls -l test/
drwxr-xr-x 2 goksin goksin   512 23 Haz 18:16 .
drwxr-xr-x 3 goksin goksin   512 23 Haz 18:16 ..
[goksin@droideka:~]$
```

Bir dosya/dizin için üç yetkilendirme kipi söz konusudur; kullanıcı (sahibi), grup ve diğerleri. Ayrıca üç adet erişim kipi de söz konusudur; oku, yaz ve çalıştır. Bu altı adet erişim yetkisi "bit" olarak tanımlanır ve bir dosya/dizin için tanımlanmış olan yetkileri belirtir. Dosyanın/dizinin sahibi, grubu ve okuma-yazma-çalıştırma yetkileri tanımlanır. Dosyalara ait olan çıktıda dosyanın sahibinin dosyalara yazma,okuma ve çalıştırma hakkı bulunurken grup ve diğerlerinin sadece okuma hakkı bulunmaktadır. Okuma, yazma ve çalıştırma yetkileri birbirine bağlı değil, bağımsızdır.

Bir Dosyanın/Dizinin Sahibini Değiştirmek

Dosyalar ve dizinler üzerinde sınırsız yetkiye sahip olan kişi root olduğu için dosyanın sahibini ve grubunu da değiştirebilir. Diğer kullanıcılar kendilerine ait olan bir dosyayı bir başka kullanıcıya "veremez" veya bir başka kulanıcının dosyasını "alamaz". Eğer bu olanaklı olsaydı, kullanıcılar ve izinler anlamsız ve işlevsiz hale gelirdi. Bir dosya ve dizinin sahibini değiştirmek için -CHange OWNer-chown kullanılır.

```
droideka# chown akdeniz 01.01.2009.txt
droideka#
```

Böylelikle bir dosyanın sahibi değiştirmiş olduk. Grup ise değişmedi Dosyanın yeni sahibi akdeniz kullanıcısı olurken goksin kullanıcısı artık 01.01.2009.txt dosyasına yazamaz veya dosyayı okuyamaz. Benzer biçimde dizinlerin de sahiplerini değiştirebilirsiniz.

```
droideka# chown akdeniz /home/goksin/test
droideka#
```

Bir dizin üzerinde chown kullandığınızda "." üzerinde işlem yapar. Üst dizin -parent directory- üzerinde işlem yapmaz. Yukarıdaki komutu verdiğimde akdeniz kullanıcısı artık test dizini üzerinde dosya oluşturabilen ve silebilen kullanıcı durumundadır. Akdeniz kullanıcısı yazma, okuma yetkisi olan tüm dizinlerde dosya oluşturabilir. Ancak kendisi tarafından oluşturulmamış olan dosyalar üzerinde işlem yapamaz. Bu kısım karışık gelebilir. Ancak aşağıdaki tablo kavramınızda yararlı olacaktır.

```
Sıra  Değer       Anlamı
-----------------------------------------------------
1     d, -        Dizin ise (d), dosya ise (-) belirtir
2     r, -        Sahibinin okuma izni var.
3     w, -        Sahibinin yazma izni var.
4     x, -, s     Sahibinin (x) çalıştırma izni,(s) varsa
                  setuid kipi ile çalıştırılır.
5     r, -        Grubun okuma yetkisi vardır.
6     w, -        Grubun yazma yetkisi vardır.
7     x, -, s     Grubun (x) çalıştırma yetkisini, (s) varsa
                  setgid kipi ile çalıştırılır.
8     r, -        Diğerlerinin okuma izni vardır.
9     w, -        Diğerlerinin yazma izni vardır..
10    x, -, t     Diğerleri tarafında çalıştılabilir. (t)
                  varsa "sticky" dizindir. Yani kullanıcı
                  kendisine ait olmayan dosyaları silemez.
```

chown -R <dizin< şeklindeki kullanım chown komutunun alt dizinlere ve dosyalara da uygulanmasını sağlar. Özellikle sistemin yönetiminde bir dizindeki dosyalar ve alt dizinleri olduğu gibi bir başka kullanıcıya atayacak iseniz bu komutu kullanmanız işinizi kolaylaştıracaktır.

Bir Dosyanın/Dizinin Grubunu Değiştirmek

Bir dosyanın/dizinin sahibi değiştirebildiğimiz gibi grubunu da değiştirebiliriz. Bunun için -CHange GRouP- chgrp kullanılır. chown ile aynı şekilde çalışır.

```
droideka# chgrp wheel bsd_temmuz_2009_yazısı.txt
droideka# ls -l bsd_temmuz_2009_yazısı.txt
-rw-rw-r-- 1 goksin whhel 19218 Haz 23 22:32 bsd-
Temmuz_2009_yazısı.txt
droideka# pwd
/usr/home/goksin
droideka#
```

Yukarıdaki işlemi aynı zamanda aşağıdaki gibi de gerçekleştirebilirsiniz.

```
droideka# chown goksin:wheel bsd_temmuz_2009_yazısı.txt
```

Benzer olarak da

```
droideka# chgrp :wheel bsd_temmuz_2009_yazısı.txt
```

aynı işlevi yerine getirecektir.

Dosyanın sahibi ve grubu okuma ve yazma iznine sahip olduğu için üçüncü ve altıncı değerler w olarak görülmektedir. Bu durumda goksin kullanıcısı ile aynı grupta olan diğer kullanıcılar dosyaya yazabilir.

BSD sistemlerde standart olarak sisteme eklediğimiz her kullanıcı için kullanıcı adı ile aynı ada sahip olan bir grup oluşturulur. Bu grup o kullanıcı için birincil gruptur. Bu gruba üye olan tüm kullanıcılar dosyaya yazabilir. Bu durumda basit bir dosya paylaşımı yaratmış olursunuz.

Dosya ve Dizin İzinleri

Dosyalara ve dizinlere ait izinleri tam olarak kavrayabilmek için bit değerlerini okumak ve yorumlamak gerekir. Bu bit değerlerinin ilki bir dizin ise "d", dosya ise "-" olarak gösterilir. Bu değer izinler konusunda bir bilgi vermez ancak dosyalar ile dizinleri ayırmanızı kolaylaştırır.

Diğer bitlerdeki değerler ise daha kolaylıkla anlaşılabilir. Üç izin üç ayrı grup için tanımlanır. -rw-rw-r-- bitlerinin kullanıcı ve grubu için okuma yazma izinleri verilmiş iken diğerleri sadece dosyayı okuyabilir. Bir dizinin çalıştırılma iznine sahip olması gereklidir. Aksi halde içeriği görüntülenemez.

Dosya ve Dizin İzinleri Arasındaki İlişkiler

Aşağıdaki tabloda kullanıcı ve üyesi olduğu grup dikkate alınarak bir dizin üzerinde tanımlanan izinlere bağlı olarak gerçekleştirilebilecek olan işlemler gösterilmektedir.

İşlem	lanıcı	Grup	Diğerleri
Dosya oluştur	Evet	Evet	Hayır
(Kullanıcı dosyaya sahip/yazabilir) Dosyayı sil	Evet	Evet	Hayır
(Grup dosyaya sahip/yazabilir) Dosyayı sil	Evet	Evet	Hayır
(Diğerleri dosyaya sahip/yazabilir) Dosyayı sil	Evet	Evet	Hayır
(Kullanıcı dosyaya sahip/yazabilir) Dosyayı yeniden adlandır	Evet	Evet	Hayır
(Grup dosyaya sahip/yazabilir) Dosyayı yeniden adlandır	Evet	Evet	Hayır
(Diğerleri dosyaya sahip/yazabilir) Dosyayı yeniden adlandır	Evet	Evet	Hayır
(Kullanıcı dosyaya sahip/yazabilir) Dosyayı düzenleyebilir	Evet	Evet	Evet
(Grup dosyaya sahip/yazabilir) Dosyayı düzenleyebilir	Evet	Evet	Evet
(Diğerleri dosyaya sahip/yazabilir) Dosyayı düzenleyebilir	Hayır	Hayır	Hayır

Dosya/Dizin İzinlerini chmod ile Düzenlemek

Bir dosya ve dizindeki izinleri yorumlamak kadar aynı zamanda düzenlemek de gereklidir. Bazı kaynaklarda izin yerine kip veya İngilizce karşılığı olan mode terimi de kullanılmaktadır. Bir dosya ve dizinin izinlerini sayısal olarak veya harfler/karakterler ile düzenleyebilirsiniz.

Dosya/Dizin İzinlerini Sayısal Olarak Düzenlemek

Bir dodya ve dizinin izinlerini düzenlemenin sayısal yolu 8 tabanında üç basamaklı sayısal değeri tanımlayarak yapabilirsiniz. Bu üç basamaklı sayıda her basamak sahip, grup ve diğerleri için tanımlanan izinleri belirtir. Bazen dördüncü basamağın yer aldığı komutlara rastlayabilirsiniz veya kullanabilirsiniz. Bu dördüncü basamağa ilerleyen bölümlerde yer vereceğiz.

Üç basamaklı sayıda kullandığımız rakamlar farklı izinlerin sayısal karşılıklarının aritmetik toplamıdır. Bu toplam okuma+yazma+ çalıştırma izinlerinin toplanmasıdır. Aşağıdaki tabloda bu değerleri görebilirsiniz.

Bit	Karşılığı
0	İzin yok
1	Çalıştır (dizinler için içeriği gösterir)
2	Yaz
4	Oku

Bu sayıları kullanarak bir dizin/dosyanın okuma, yazma ve çalıştırma izinlerini sayısal olarak tanımlayabilirsiniz. Okuma+yazma+çalıştırma izni 7 iken oku+yaz izni 6 olacaktır. Bu sistematik ile bir dizinin/dosyanın izinlerini aşağıdaki örneklerde görüldüğü gibi tanımlayabilirsiniz.

Değer	Dizge	İşlemler
755	-rwxr-xr-x	kullanıcı okua+yaz+çalıştır, grup ve diğerleri oku+çalıştır
644	-rw-r--r--	Kullanıcı oku+yaz, grup ve diğerleri okuyabilir.
600	-rw-------	Kullanıcı oku+yaz, grup ve diğerlerinin erişimi yok

Bu tablolara baktığımızda sekizlik tabanda yazılan değerlerin aslında onluk sistemde verildiğini görüp bir hata olduğunu düşünebilirsiniz. Sekizlik tabanda 755 karşılığı ikilik sistemde 111101101 'dır. Tüm izinlerin tanımlandığı rwxrwxrwx dizgesi ile 755 değerini "and" işlemine tabi tutarsanız sonuçta rwxr-xr-x dizgesini elde edersiniz.İzin değerlerinin "bit" olarak tanımlanmasındaki neden gerçekten bir "bit" olmalarından kaynaklanmaktadır.

BSD sistemlerde kullanabileceğimizi izin sisteminin sadece üç basamağı dışında yukarıda adı geçen dördüncü basamak dizinlerin ve dosyaların özel durumlarda davranışını kontrol etmektedir. Bu dördüncü "bit" ile kullanılan değerler ve işlevleri şu şekildedir:

0 Normal izinler.

1 "Sticky bit" Bu değer dizinlere atanır. Atanan dizin "append-only" yani dizine dosya eklenebilir ama silinemez. Normal kullanıcılar dosyanın sahibi ve yazma yetkisine sahiip ise dosyayı yeniden adlandırabilir ve silebilir. Çalıştırılabilen dosyalar için geçerli değildir.

2 Group ID (setgid) atar. Çalıştırılabilen bir dosyaya atanırsa dosyanın sahibinin üye olduğu birincil grup izinleri ile çalıştırılır. Bu değer ancak root kullanıcısı tarafından atanabilir.

4 User ID (setuid) atar. çalıştırılabilen bir dosyaya atanırsa, bu durumda dosyanın sahibinin izinleri ile çalıştırılır. Bu değer ancak root

kullanıcısı tarafından atanabilir.

En sol başta yer alan dördüncü basamak özel durumlar için kullanılır. 755 olarak tanımladığınız aslında 0755'tir. Diğer basamaklarda kullandığımız gibi dördüncü basamak için değerleri aritmetik toplam olarak tanımlayabiliriz. 3755 ile oluşturacağınız dizin "sticky bit" ve setgid ile birlikte 755 izin atamasına sahiptir.

Dosya/Dizin İzinlerini Harf/Karakterle Düzenlemek

Sayısal değerler dosya sisteminde saklamak için tasarlanmıştır. Dosya sisteminin etkili bir şekilde kullanılmasını sağlar ve bilgisayar tarafından kolaylıkla okunur ve işlenir. Bu nedenle sistemi kullanan bizler için sayısal değerler yerine kolay anlaşılabilir harfler kullanılır. Bu harfler farklı düzenlemelerle kullanılabilir. Aşağıdaki örneklerde yaygın olarak kullanılan bazı uygulamalara yer verilmiştir. Ayrıntılı bilgi kılavuz sayfasından edinilebilir.

İlk harf sahip izinlerini tanımlar. ardından gelen karakter ise yapmakta olduğunuz düzenlemeyi tanımlar. (+, -, or =), Üçüncü karakter ise atama yaptığınız "bit"leri tanımlar.

```
Dizge             İşlem
-------------------------------------------------------
go+w        Yazma iznini grup ve sahibe atanır

+x          Herkese çalıştırma izni atanır.

o-r         Diğerlerinin okuma izni iptal edilir.

ugo=rw      Herkes için okuma ve yazma izni atanır

a=rw        Herkes için okuma ve yazma izni atanır.

+t          "Sticky bit"  atanır.

+s          "setuid" ve "setgid" bitleri atanır.
```

Kullanıcı ve Grupların Yönetimi

Kullanıcı ve gruplar UNIX ve BSD sistemler ile türevi olan Linux dağıtımlarında ve diğerlerinde hazır olarak sunulsalar da bazen sistem yöneticisi tarafından yeni kullanıcıların ve grupların eklenmesi, bazen de var olan kulanıcıların ve grupların da kaldırılması gerekebilir. Bu ekleme ve kaldırma işlemlerini yaparken iki seçeneğimiz bulunur: Birincisi sysinstall kullanmak ikincisi kullanıcı ve grup eklemek için adduser, rmuser kullanmaktır.

BSD'de kullanılan adduser aracı, Linux dağıtımlarında kullanılan useadd/adduser araçlarından farklı çalışsa da aynı işlevi yerine getirmektedir. Sisteme yeni bir kullanıcı veya grup eklemek root kullanıcısının yetkisindedir. Yukarıda sözünü ettiğimiz gibi sudo kullanarak bunu yapabilirsiniz.

Kullanıcı Hesaplarını Yapılandırmak

Sistemi ilk kurduğunuzda bir veya daha çok sayıda kullanıcıyı sysinstall ile eklediniz. Sysinstall ile eklediğiniz kullanıcılar varsayılan yapılandırma ile eklenmektedir. Bu yapılandırmayı sistemin işletilme amacına göre düzenleyebilirsiniz. Kullanıcı hesapları için varsayılan yapılandırmayı görüntülemek ve düzenlemek için aşağıdaki komut kullanılır:

```
# adduser -C
```

Bu komut, adduser aracının kullanacağı yapılandırma dosyasını oluşturmanızı sağlar. Bu dosya sisteme sonradan eklenecek olan tüm kullanıcıların yapılandırılmasında kullanılacaktır. Kullanıcılar için kabuk, grup üyelikleri, kullanıcı şifreleri vb. bu yapılandırmaya göze düzenlenecektir. Bu yapılandırmayı kendi gereksinmelerinize göre düzenlemek sonradan eklenecek her kullanıcı için yapılandırmayı baştan düzenleme zahmetinden sizi kurtaracaktır. Aşağıda örnek bir

yapılandırma görülmektedir.

```
# adduser -C
Login group []:
Enter additional groups []:
Login class [default]:Turkce
Shell (sh csh tcsh bash nologin) [sh]: csh
Home directory [/home/]:
Use password-based authentication? [yes]:
Use an empty password? (yes/no) [no]:
Use a random password? (yes/no) [no]: yes
Lock out the account after creation? [no]:
Pass Type  : random
Class      :
Groups     :
Home       : /home/
Shell      : /bin/csh
Locked     : no
OK? (yes/no): yes
Re-edit the default configuration? (yes/no): no
Goodbye!
```

Yapılandırmada kullanılan ön tanımlı değerler köşeli parantez içinde gösterilmektedir. Eğer farklı bir değer girmez iseniz ve enter tuşuna basarsanız parantez içindeki değerler kullanılacaktır. Yukarıdaki örnekte standart yapılandırma dışında fazlaca bir değişiklik yapılmamıştır. Yapılandırma tamamlandığında "yes" değeri girilip yapılandırma /etc/adduser.conf dosyasına kayıt edilmektedir. Eğer sonradan değişiklik yapmak isterseniz ya adduser.conf dosyasını bir metin editörü ile yeniden tercihlerinize göre düzenlemeli veya "adduser -C" ile yeniden yapılandırmalısınız.

Yapılandırmada bazı seçenekler örneğin "login group" gibi kullanıcı eklenirken tanımlanacağı için boş bırakılmıştır. Benzer olarak kullanıcıların ev dizinleri /home/<kullanıcı_adı> olarak yapılandırılacağı için bu değer de olduğu gibi bırakılmıştır. "login class" değişkenine Turkce değeri atanarak sistem yapılandırmasında kullanıcıların kulanacağı dil Türkçe olarak tanımlanmıştır. Bu değer sisteme eklenecek olan kullanıcıların dil ayarlarının otomatik olarak Türkçe atanmasını sağlar. Bu yapılanma ise /etc/login.conf dosyasının sonuna eklenen bir kaç satır ile tamamlanır. Yapılandırılmanın tamamlanmasından sonra sisteme yeni kullanıcılar ekleyebilirsiniz.

Aşağıdaki adduser ile sisteme yeni bir kullanıcı eklenmetedir.

```
droideka# adduser
Username: ben10
Full name: Ben Ten
Uid (Leave empty for default):
Login group [ben10]:
Login group is ben10. Invite joe into other groups? []:
Login class [Turkce]:
Shell (sh csh tcsh bash nologin) [sh]:
Home directory [/home/ben10]:
Use password-based authentication? [yes]:
Use an empty password? (yes/no) [no]:
Use a random password? (yes/no) [yes]:
Lock out the account after creation? [no]:
Username   : ben10
Password   : <random>
Full Name  :
Uid        : 1002
Class      :
Groups     : joe
Home       : /home/ben10
Shell      : /bin/csh
Locked     : no
OK? (yes/no): yes
adduser: INFO: Successfully added (ben10) to the user
database.
adduser: INFO: Password for (ben10) is: wquAd0GYccFmLr
Add another user? (yes/no): no
Goodbye!
```

Kullanıcıya ait bilgiler /etc/adduser.conf dosyasında tanımlandığı için özel bir değişikliğe gerek duyulmuyorsa ön tanımlı değerler kullanılarak kullanıcı sisteme eklenecektir. Kullancılara kullanmaları için

kullanıcı şifrelerinin yaratılması seçildiği için kullanıcıya verilecek olan kullanıcı şifresi de oluşturulmaktadır. Bu şifreyi kullanıcıya verdiğinizde sisteme tanımlanmış olan kullanıcı adı ve şifresi ile giriş yapabilecektir. Eğer bir şifre oluşturulmasını seçmediyseniz bunu kullanıcıyı sisteme eklerken kendiniz şifre atamak ve bu şifreyi de iki defa arka arkaya girerek yapmanız gerekecektir. Şifre seçilirken özel karakterleri pek kullanmamanız gerekir. Zira kullanıcılar bu karakterleri yanlış girmekte veya şifereye akılda kalmadığı gerekçesi ile itiraz edebilmektedirler. Bu nedenle büyük-küçük harf ve sayıdan oluşan bir şifre kullanılması en uygun çözüm olacaktır.

Kullanıcı ekleme işleminin ardından bir sorun olup olmadığını kontrol etmekte yarar vardır. Bunun için kullanıcıyı finger ile kontrol ederek kullanıcı hesabının aktif olup olmadığını kontrol edebilirsiniz.

```
droideka# finger ben10
Login: ben10                            Name: Ben Ten
Directory: /home/ben10                  Shell: /bin/sh
Never logged in.
Mail last read 25 Haz Perş 19:41 2009 (EEST)
No Plan.
```

Kullanıcı hesabı eklenmiş ve aktif durumda. Kullanıcı ve dizinindeki dosyaları da kontrol etmek gerekir. Bu dosyalar /usr/share/skel dizininden kopyalanır. Kullanıcı ev dizininde ls -l ile bu dosyaları kontrol edin.

```
droideka# pwd
/usr/home/ben10
droideka# ls -l
.cshrc          .login_conf     .mailrc         .rhosts
.login          .mail_aliases   .profile        .shrc
```

Bu dosyalar kopyalandığına göre kullanıcı için gereken temel yapılandırma dosyaları da hazır demektir.

Kullanıcıyı Sistemden Çıkarmak: rmuser

Bir kullanıcıyı sistemden çıkarmak son derece kolaydır. Bu iş içim rmuser kullanılır. Silmek istediğiniz kullanıcıya ait kullanıcı adını vererek kullanıcı sistemden kaldırılabilir.

```
droideka# rmuser ben10
Matching password entry:

ben10:*:1002:1002::0:0:Ben Ten:/home/ben10:/bin/csh

Is this the entry you wish to remove? y
Remove user's home directory (/home/ben10)? y
Removing user (ben10): mailspool home passwd.
```

"matching password entry" satırı kullanıcıya ait olan bilgilerin saklandığı veritabanını belirtmektedir. Daha önceki yazılarda kısaca değindiğimiz /etc/passwd dosyası burada yeniden karşımıza çıkıyor. Kullanıcılarai ilişkin bilginin saklandığı dosya /etc/passwd dosyasıdır. UNIX ve türevi işletim sistemlerinde mutlaka bir /etc/passwd dosyası bulunur. Bu dosyanın işlevi ise sistemler arasında farklılıklar gösterir. Bazılarında kullanıcılara ilişkin tüm bilgilerin tutulduğu dosyadır. Bu durumda bu dosyaya bir metin editörü ile eklenecek olan bir satır ile sisteme kullanıcı ekleyebilirsiniz. Bugün ise kullanıcı şifreleri "shadow passwords structure" - maskelenmiş şifreleme yaklaşımı ile saklanır. (Okuyucular için maskelenmiş şifreleme yapılandırması terimi yabancı gelebilir. Ancak "gölge şifreler" terimi yerine kullanılanıldığında daha açıklayıcı olduğunu düşündüğüm için kendi bulduğum bir terim. Yazıların bu ve bundan sonraki bölümlerinde yeri geldikçe bu terimi kullanacağım.) Maskelenmiş şifreler yapılandırılması ile kullanıcı şifreleri /etc/passwd dosyasında saklanmaz. Şifreler sadece root tarafından okunabilecek olan bir dosyada şifrelenerek saklanır. Bazı sistemlerde kullanıcı şifreleri bu şekilde /etc/shadow veya etc/security/master.passwd dosyasında saklanır. BSD sistemlerde ise bu bilgiler /etc/master.passwd dosyasında saklanır.

Bir BSD sistemi kurup yönetecekseniz /etc/passwd ve /etc/master.passwd dosyaları kritik öneme sahiptir. Her iki dosya da basit birer metin dosyasıdır. Kullanıcılar ve bilgileri tek satırlık girdiler olarak yer alır. Kullanıcı adı, UID, GID ve diğer bilgiler (:) ile ayrılmış olarak saklanır.

/etc/passwd dosyasının izinleri 644, /etc/master.passwd dosyasının izinleri ise 600'dür. Her iki dosyanın sahibi root kullanıcısıdır. Her iki dosya normal kullanıcılar tarafından erişilebilir. /etc/masterpasswd dosyasının içeriği ise sadecde root tarafında görülebilir. /etc/master.paswd dosyası /etc/passwd dosyasından farklı olarak maskelenmiş şifreleri ikinci girdi olarak saklar.

Az sayıda kullanıcının bulunduğu bir sistemde kullanıcı bilgileri maskelenerek ve metin dosyaları olarak saklanması yeterli olabilir. Ancak çok sayıda kullanıcının bulunduğu bir sistemde metin dosyaları içinde kullanıcı bilgilerinin tutulması uygun bir çözüm değildir. Bir kullanıcının sisteme giriş yapması için bu dosyanın okunup işlem yapılması gereklidir. Onlarca, yüzlerce veya daha çok sayıda kullanıcının kullandığı bir sistemde bu yaklaşım ciddi darboğazlara neden olacaktır.

BSD sistemler bu nedenle /etc/pwd.db ve /etc/spwd.db dosyalarını kullanırlar. Bu dosyalarda veritabanı formatında maskelenmiş olarak kullanıcı şifreleri saklanır. Bu iki dosya yukarıda adı geçen dosyalara karşılık gelir. İzinleri de yine yukarıda olduğu gibidir. Veritabanı yöntemi çok sayıda kullanıcının bulunduğu sistemlerde hız, güvenlik ve kolaylık sağlar. Bu dosyalar pwd_mkdb uygulaması tarafından oluşturulurlar. Kullanıcı bilgilerinin passwd, shfn veya adduser/rmuser ile düzenlenmiş olsa da bu veri tabanına kayıt edilirler. BSD sistemlerde sistemin temel parçası olanlar ve diğer komutlar/araçlar kulanıcı bilgilerini /etc/pwd.db dosyasından okurlar.

Kullanıcı bilgilerinin metin dosyaları içinde saklanması nedeni ile kullanıcı yapılandırma araçları genelde bir metin dosyasını düzenleyip bu dosyayı kayıt ederler. chfn adlı araç da kullanıcı bilgilerini güncellemek için kullanılan bir uygulamadır. EDITOR değişkeninde tanımlı olan düzenleyiciyi çalıştırıp kullanıcı bilgilerini düzenler ve kayıt eder. BSD sistemlerde EDITOR değişkeni vi olarak tanımlıdır. chfn ile vi kullanarak kullanıcı bilgilerini düzenleyebilirsiniz. Kullanıcı bilgileri düzenlenip kayıt edildiğinde /etc/master.passwd dosyası da yeniden yazılır. Bunun ardından pwd_mkdb -p otomatik olarak çalıştırıp diğer ilgili dosyalar da yeniden düzenlenir. Burada /etc/master.passwd dosyasının asıl dosya olduğu ve diğerlerinin bu dosyada yapılan değişikliklerin kopyalandığı dosyalar olduğunu vurgulamakta yarar var. Bu dosyanın bir yedeğini almak ve güvenli bir yerde tutmakta yarar var.

Bu dosyayı bazı durumlarda örneğin donanım terfisi vb nedenler ile kullanıcıları sorunsuz bir şekilde bir sistemden diğerine taşırken kullanabilirsiniz. Eski sistemdeki /etc/master.passwd. dosyasını ve diğer verilerinizi yeni donanıma taşıdıktan sonra aşağıdaki komutu kullanarak eyni sisteme tüm kullanıcılarınız taşımış olursunuz. Aşağıdaki örnekte eski sisteme ait olan /etc/master.passwd dosyasını yeni sisteme /etc/master.passd.aktarma olarak taşındıktan sorna yeni sisteme kullanıcıların aktarılmasını görüyorsunuz. Kullanıcılar yeni sistemdeki /etc/passwd dosyasına ve ilgili diğer dosyalara aktarılmış olur. /etc/master.pwd dosyasındaki verilerin /etc/passwd dosyasına aktırılmasını sağladıktan sonra /etc/spwd.db ve /etc/pwd.db dosyaları da oluşturulacaktır. Eğer -p seçeneğini kullanmazsanız yeni sistemdeki /etc/passwd dosyası olduğu gibi kalacak ve eski sistemdeki kullanıcılar yeni sisteme aktarılmamış olacaktır.

```
# pwd_mkdb -p /etc/master.passwd.aktarma
```

Gruplar ve Grup Yönetimi

Sisteme eklenen yeni kullanıcıların birincil grubu kullanıcı adı ile aynıdır. adduser aracı sisteme yeni bir kullanıcı eklediğinde kullanıcıyı

birincil grup olarak kullanıcı adı ile aynı olan gruba dahil eder. Grup ID ile kullanıcı ID aynıdır. Bu eşleşme durumu zorunlu değildir ve normal kulanıcıların eklenmesinde önemli bir durum değildir. Tersine olarak sistem genelinde eklenecek olan gruplarda örneğin sunucu uygulamalarında olduğu gibi önemlidir. Sistem genelinde kullanılacak olan grupların ID aralığı 100 ile 1000 arasında değişir. 1000 ve sonrasındaki değerler sisteme eklenecek olan normal kullanıcı hesaplarına atanır. Örneğin goksin kullanıcısı 1001 ve benten kullanıcısı 1002 kullanıcı ve grup ID değerlerine sahiptir. 100'ün altındaki değerler ise sistem gruplarına ayrılır ve atanır. Bu değerleri kendi yarattığınız kullanıcılar için atamayın. Sonradan portlar üzerinden kuracağınız bazı uygulamalar yapılandırılması gereğ bu aralıktaki değerleri grup ID olarak alabilir. Bu da problem demektir.

Grup bilgileri /etc/groups dosyasında tutulur. Bu dosya da basit bir metin dosyasıdır ve başka bir veritabanı karşılığı bulunmaz. Gruplara kullanıcı şifresi atanmayacağı için /etc/groups dosyasında güvenlik önlemleri almaya gerek yoktur ama sahibi ve yazma yetkisi olan kullanıcı ise root kullanıcısıdır. /etc/groups dosyasını bir metin düzenleyicisi ile açarak içeriğine göz atacak olursanız aşağıdakine benzer girdiler görebilirsiniz. Aşağıdaki örnekte wheel grubuna ait olan satır gösterilmektedir.

```
wheel:*:10:root,goksin
```

Dosyada dört adet alan görülmektedir. İkinci alan şifreler için bırakılmıştır. Yukarıda şifrelere gerek olmadığını belirttik ama burada şifre alanı ile karşı karşıyayız. Bu alanın bulunmasının nedeni geliştiricilerin şifreye gerek olmadığına karar verip boş geçmelerinden kaynaklanmaktadır. Ancak birisinin veya birilerinin olur da şifre kullanılmasını zorunlu olduğuna ikna edebilecekleri düşünülerek bu alan "*" ile ayrılmıştır. Ancak bu gün için gruplara şifre atamayı gerektirecek bir durum söz konusu değildir.

Bir kullanıcıyı bir gruba eklemek isterseniz en son alanda yer alan kullanıcı isimleri bölümüne kullanıcının adını ekleyin. Çıkarmak isterseniz de kullanıcının adını silin. Peki ya yeni bir grup eklemek veya kaldırmak istersek ne yapacağız? Sysinstall ile yeni bir grup ekleyebilirsiniz. Sysinstall dialogları ile grup ekleyebilir bir ID atayabilirsiniz. Diğer yöntem ise doğrudan metin editörü ile /etc/groups dosyasını açıp grup bilgilerinin elle girilmesi ile yapılabilir. Grup ID değerinin ise, sistemde kullanılmayan ve en önemlisi yukarıda değindiğimiz sakıncaları ortaya çıkarmayacak bir değer olarak seçilmesine dikkat edin. Basitçe var olan bir grup satırını alıp kopyaladıktan sonra gereken düzenlemleri yaparak yeni bir grup oluşturabilirsiniz. Bir grubu silmek için de o grubun bilgilerinin bulunduğu satırı silin.

BSD - VIII

BSD açılış sürecinin işleyişi ve yapılandırması

Gökşin Akdeniz
goksin@enixma.org

BSD Açılış Sürecinin İşleyişi ve Yapılandırması

BSD kurulum aşamasının tamamlanmasının ardından sisteminizi yeniden başlatmanız gerekir.Başlattığınızda diğer işletim sistemlerinin aksine BSD açılışta size gösterişli açılış menüleri ve ekranları sunmaz. Bu tür şeylerden hoşlanıyorsanız kendiniz hazırlayıp ekleyerek kullanabilirsiniz.

BSD açılış sürecini ayrıntıları ile ele almadan önce bilgisayarın açılış sürecine göz atmak yerinde olacaktır. Aşağıdaki bölüm eldeki pc donanımına uygun olarak x86 ve AMD64 esas alınarak hazırlanmıştır. Diğer platformlardaki açılış süreçlerini FreeBSD Handbook ile diğer kılavuzlarda ayrıntılı olarak bulabilirsiniz.

POST ve BIOS

Bilgisayarın açılması sırasında ilk yapılan işlem POST yani Power-On-Self-Test'dir. POST aşamasında BIOS yani Basic Input/Output System çalıştırılır. BIOS basitçe bilgisayarınızın tüm donanımını kontrol eder. Bu kontrol işlemi donanımın belirlenip çalışma durumunun kontrol edilmesinin ardından açılacak olan işletim sisteminin belirlenmesinden ibarettir. Bu işlemlerin tamamlanmasının ardından POST aşaması tamamlanmış olur.

Bootstrap

POST aşamasının tamamlanmasının ardından BIOS sistemin başlatılması için tanımlanmış olan donanımı seçerek bilgisayarı başlatır. Genel olarak açılış için takılı olan ilk sabit disk kullanılır. Ancak açılış sırasında kullanılacak olan donanım tercihi yapılandırmaya göre disket, optik sürücü veya bir başka donanım örneğin ağ kartınız ile uzak bir sunucuya erişerek başlatılabilir. BIOS işletim sisteminin başlatılması için bootstrap kullanır. Bootstrap işletim sisteminin parça parça belleğe yüklenmesini sağlar.

Açılış Dilimi - Bootable Slice (boot0)

BIOS'un sistemdeki donanımı kontrol etmesinin ardından Ana Açılış Kaydı-MBR yani boot0 bloğu ile ardından gelen boot1 ve boot2 bloklarını okuyarak yükleyici çalıştırır ve yükleyici de çekirdeğin belleğe yüklenmesini sağlar. boot0 ve boot1 blokları 512 Byte boyutundadır.

boot0'da yer alana açılış yöneticisi ile seçim yaptığınızda boot1 ve boot2 bloklarında yer alan yükleme programı çalıştırılarak çekirdek yani kernel yüklenecektir. Çekirdeğin yüklenebilmesi için öncelikle açılış için tanımlanmış olan disk vb. donanımdaki açılış bölümünda yer alan programın belleğe yüklenmesi gereklidir. Bu konuyu yazının ilerleyen bölümlerinde yeniden ele alacağız. Bazen açılış sırasında ortaya çıkan problemler bu sürecin kontrol edilmesi ile aşılabilir.

boot1, bu blokta çok basit bir program yer alır. Disklabel uygulamasının bir benzeridir. Disk üzerindeki dilimi, BSD tarzı dilimlere ayırıp boot2'yi bulmaktır. boot1 üzerinde herhangi bir kullanıcı arayüzü yer almaz.

boot2, sistemin başlatılacağı dosya sistemindeki dosyaları yükleyecek olan 'loader'ı çalıştıran açılış bloğudur.

boot0, yani bildiğimiz adı ile MBR, FreeBSD açılış yönetisinin kurulduğu yerdir. Program bilgisayarda bulunan dilimler ile bu dilimlerdeki işletim sistemlerinin bir listesini sunar. F1, F2 vs. basarak bu işletim sistemlerinden birisi seçilir.

```
F1 FreeBSD
F2 Linux
F3 DOS

Default: F1
```

Ekrandaki listeden seçtiğiniz sistemi başlatmak için ilgili tuşa basmanız yeterlidir. Burada boot0 bloğu ile ardından boot1 boot2 bloklarında yer alan programlar çalıştırılarak açılış süreci devam eder.

Loader

Loader, /boot dizininde yer alır. /boot/defaults/loader.conf ve /boot/loader.conf yapılandırma dosyalarını okuyarak çekirdeği ve modülleri yükler.

Loader 10 sn süreyle sizden bir giriş yapmanızı bekler. Eğer bir seçim yapmaz iseniz, bu durumda varsayalın seçeneği yani çekirdeği, tanımlı servisleri vs.çalıştıracaktır. Diğer seçenekler ise çekirdeğin başlangıç sürecini kontrol etmenize olanak verir. Güvenli kip-safe mod- sistemi başlatabilirsiniz. Bu kipte bir çok çekirdek modülü yüklenmez. Tek kullanıcı kipi-single-user mode- sistem yöneticisinin giriş yaparak bakım moduna geçmesini sağlar. Escape to Loader Prompt-Komut satırı kipinde ise loader komut satırı kipine geçiş yapar. Böylece farklı bir çekirdek, CD/DVD üzeriden sistemi açabilir, farklı çekirdek modüllerini yükleyebilir veya kaldırabilirsiniz. Yapılandırma dosyalarını düzenleyebilirsiniz. Daha ayrıntılı bilgi loader kılavuz sayfasında yer alıyor. (man loader).

Bootstrap sürecinin sonuna geldiğinizde çekirdeğin yüklenmesi başlar. Çekirdek belleğe yüklenir, bulunan donanımı kontrol eder, yapılandırma programlarını çalıştırır. Bu aşamadan sonrası artık çekirdeğin kontrolündedir.

Çekirdek veya Kernel

BSD sistemlerde standart olarak derlenmiş olan çekirdek /boot/kernel dizininde yer alır. Farklı bir yapılandırma söz konusu olmadığı sürece bu çekirdek kullanılır. Açılış sırasında ekranda göreceğiniz yazılardan bazıları beyaz, bazıları gri renklidir. Beyaz renkli olan metin çekirdek mesajlarıdır.

Çekirdeğin donanımınızı bulup, çalıştırması ve diğer durumlar ile ilgili olarak döndürdüğü mesajlardır. Bu mesajları okumanız, ekrandan geçiş hızları nedeni ile pek olanaklı olmayacaktır. Sisteme giriş yaptığınızda dmesg | less komutu ile bu mesajları okuyabilirsiniz.

Uzun süre açık olan sistemlerde ise dmesg çıktısı sistemin çalışma

süresine bağlı olarak sadece son olayları kapsayabilir. Dmesg sadece çekirdek mesaj tamponundaki -kernel message buffer- mesajları döndürür ve bu mesajlar tamponun kapasitesi ile sınırlıdır. Bundan dolayı açılış mesajının bir kopyası /var/run/dmesg.boot dosyasında saklanır. Özellikle çekirdek tarafından belirlenen donanım ile bu donanıma atanmış olan isimleri belirlemek istiyorsanız /var/run/dmesg.boot dosyası bakmanız gereken yerdir.

Çekirdek mesajlarını okumak sıradan bir kullanıcı için bir anlam ifade etmeyecektir. Öte yandan sistemin işletiminden sorumlu olan sistem yöneticileri ile power-user olarak nitelendirilen kullanıcılar için son derece yararlı olmaktadır. Bir sorunun çözümü için gereken bilgi çekirdek mesajlarında bulunabilir.

Aşağıdaki örnekte kendi bilgisayarıma ait olan çekirdek mesajını görüyorsunuz. Çerkirdek mesajını inceleme kolaylığı sağladığı için parçalar halinde koydum

```
Copyright (c) 1992-2009 The FreeBSD Project.
Copyright (c) 1979, 1980, 1983, 1986, 1988, 1989, 1991, 1992,
1993, 1994
      The Regents of the University of California. All rights
reserved.
FreeBSD is a registered trademark of The FreeBSD Foundation.
FreeBSD 7.2-RELEASE-p2 #0: Wed Jun 24 00:14:35 UTC 2009
    root@amd64-
builder.daemonology.net:/usr/obj/usr/src/sys/GENERIC
```

İlk dört satır telif haklarını belirten satırlardır. İşletim sistemi ile çekirdek sürümünü ve çekirdeğin derlendiği tarihi belirtir. Çerkirdeği kendiniz yeniden derlerseniz, çekirdek sürüm numarası her derleme sonucunda bir artar, yani çekirdeği dördüncü derleyişinizde çalıştırmayı başardıysanız #3 görünecektir. Beşinci satırın baş tarafı çekirdeği derleyen kişinin yerel e-posta adresi ile kullanıcı adını belirtir. Yukarıdaki örnekte gördüğünüz root@amd64-builder .daemonology.net adresi BSD geliştirme ekibine ait olan bir adrestir. Kendi çekirdeğinizi derlerseniz burada görünecek olan ise kendi bilgisayarınızın adı olacaktır. Örneğin kendi derlediğim çekirdek ile sistemi başlatmış olsaydım root@droideka olarak görünecekti. Yine aynı satırın son kısmında ise derleme işlemi sırasında kullanılan kodun yeraldığı dizin görüntülenecektir.

```
CPU: Mobile AMD Sempron(tm) Processor 3500+ (1808.24-MHz K8-
class CPU)
  Origin = "AuthenticAMD"  Id = 0x40fc2  Stepping = 2

Features=0x78bfbff<FPU,VME,DE,PSE,TSC,MSR,PAE,MCE,CX8,APIC,SEP
,MTRR,PGE,MCA,CMOV,PAT,PSE36,CLFLUSH,MMX,FXSR,SSE,SSE2>
  Features2=0x2001<SSE3,CX16>
  AMD
Features=0xea500800<SYSCALL,NX,MMX+,FFXSR,RDTSCP,LM,3DNow!+,3D
Now!>
  AMD Features2=0x19<LAHF,ExtAPIC,CR8>
usable memory = 992002048 (946 MB)
avail memory  = 954560512 (910 MB)
```

Mesajın sonraki kısmında ise çekirdek tarafından belirlenen işlemci türü ve işlemcinin özellikleri belirtilmektedir. Ayrıca sistemde bulunan bellek miktarı belirtiliyor.

Aşağıdaki kısımda FreeBSD tarafından tanınmış olan bilgisayarımdaki donanımın listesini görüyorsunuz. ACPI ve ardından anakartımda bulunan donanımlar, USB hub, grafik, ağ kartı vs diğer donanım ile bu donanımlara çekirdek tarafından atanmış olan isimler ile en sonda sabit diskimdeki bölümler yer alıyor.

```
ACPI APIC Table: <HP           APIC  >
MADT: Forcing active-low polarity and level trigger for SCI
ioapic0 <Version 1.1> irqs 0-23 on motherboard
kbd1 at kbdmux0
acpi0: <HPQOEM SLIC-MPC> on motherboard
acpi0: [ITHREAD]
acpi0: Power Button (fixed)
Timecounter "ACPI-safe" frequency 3579545 Hz quality 850
```

```
acpi_timer0: <24-bit timer at 3.579545MHz> port 0x1008-0x100b
on acpi0
acpi_ec0: <Embedded Controller: GPE 0x10> port 0x62,0x66 on
acpi0
acpi_hpet0: <High Precision Event Timer> iomem 0xfed00000-
0xfed003ff on acpi0
Timecounter "HPET" frequency 25000000 Hz quality 900
acpi_button0: <Power Button> on acpi0
acpi_button1: <Sleep Button> on acpi0
acpi_acad0: <AC Adapter> on acpi0
battery0: <ACPI Control Method Battery> on acpi0
acpi_lid0: <Control Method Lid Switch> on acpi0
pcib0: <ACPI Host-PCI bridge> port 0xcf8-0xcff on acpi0
pci0: <ACPI PCI bus> on pcib0
pci0: <memory, RAM> at device 0.0 (no driver attached)
pci0: <memory, RAM> at device 0.1 (no driver attached)
pci0: <memory, RAM> at device 0.2 (no driver attached)
pci0: <memory, RAM> at device 0.3 (no driver attached)
pci0: <memory, RAM> at device 0.4 (no driver attached)
pci0: <memory, RAM> at device 0.5 (no driver attached)
pci0: <memory, RAM> at device 0.6 (no driver attached)
pci0: <memory, RAM> at device 0.7 (no driver attached)
pcib1: <ACPI PCI-PCI bridge> at device 2.0 on pci0
pci1: <ACPI PCI bus> on pcib1
pcib2: <ACPI PCI-PCI bridge> at device 3.0 on pci0
pci3: <ACPI PCI bus> on pcib2
vgapci0: <VGA-compatible display> mem 0xb2000000-
0xb2ffffff,0xc0000000-0xcfffffff,0xb1000000-0xb1ffffff irq 16
at device 5.0 on pci0
pci0: <memory, RAM> at device 9.0 (no driver attached)
isab0: <PCI-ISA bridge> port 0x1d00-0x1d7f at device 10.0 on
pci0
isa0: <ISA bus> on isab0
pci0: <serial bus, SMBus> at device 10.1 (no driver attached)
pci0: <processor> at device 10.3 (no driver attached)
ohci0: <OHCI (generic) USB controller> mem 0xb0004000-
0xb0004fff irq 19 at device 11.0 on pci0
ohci0: [GIANT-LOCKED]
ohci0: [ITHREAD]
usb0: OHCI version 1.0, legacy support
usb0: SMM does not respond, resetting
usb0: <OHCI (generic) USB controller> on ohci0
usb0: USB revision 1.0
uhub0: <nVidia OHCI root hub, class 9/0, rev 1.00/1.00, addr
1> on usb0
uhub0: 8 ports with 8 removable, self powered
ehci0: <EHCI (generic) USB 2.0 controller> mem 0xb0005000-
0xb00050ff irq 20 at device 11.1 on pci0
ehci0: [GIANT-LOCKED]
ehci0: [ITHREAD]
usb1: EHCI version 1.0
usb1: companion controller, 8 ports each: usb0
usb1: <EHCI (generic) USB 2.0 controller> on ehci0
usb1: USB revision 2.0
uhub1: <nVidia EHCI root hub, class 9/0, rev 2.00/1.00, addr
1> on usb1
uhub1: 8 ports with 8 removable, self powered
atapci0: <nVidia nForce MCP51 UDMA133 controller> port 0x1f0-
0x1f7,0x3f6,0x170-0x177,0x376,0x3080-0x308f at device 13.0 on
pci0
ata0: <ATA channel 0> on atapci0
ata0: [ITHREAD]
ata1: <ATA channel 1> on atapci0
ata1: [ITHREAD]
atapci1: <nVidia nForce MCP51 SATA300 controller> port 0x30c0-
0x30c7,0x30b4-0x30b7,0x30b8-0x30bf,0x30b0-0x30b3,0x3090-0x309f
mem 0xb0006000-0xb0006fff irq 21 at device 14.0 on pci0
atapci1: [ITHREAD]
ata2: <ATA channel 0> on atapci1
ata2: [ITHREAD]
ata3: <ATA channel 1> on atapci1
ata3: [ITHREAD]
pcib3: <ACPI PCI-PCI bridge> at device 16.0 on pci0
pci7: <ACPI PCI bus> on pcib3
hdac0: <NVidia MCP51 High Definition Audio Controller> mem
0xb0000000-0xb0003fff irq 22 at device 16.1 on pci0
hdac0: HDA Driver Revision: 20090329_0131
hdac0: [ITHREAD]
nfe0: <NVIDIA nForce 430 MCP13 Networking Adapter> port
0x30e0-0x30e7 mem 0xb0008000-0xb0008fff irq 23 at device 20.0
on pci0
miibus0: <MII bus> on nfe0
```

```
rlphy0: <RTL8201L 10/100 media interface> PHY 1 on miibus0
rlphy0:  10baseT, 10baseT-FDX, 100baseTX, 100baseTX-FDX, auto
nfe0: Ethernet address: 00:1b:24:35:e5:04
nfe0: [FILTER]
acpi_tz0: <Thermal Zone> on acpi0
atkbdc0: <Keyboard controller (i8042)> port 0x60,0x64 irq 1 on
acpi0
atkbd0: <AT Keyboard> irq 1 on atkbdc0
kbd0 at atkbd0
atkbd0: [GIANT-LOCKED]
atkbd0: [ITHREAD]
psm0: <PS/2 Mouse> irq 12 on atkbdc0
psm0: [GIANT-LOCKED]
psm0: [ITHREAD]
psm0: model IntelliMouse, device ID 3
cpu0: <ACPI CPU> on acpi0
powernow0: <PowerNow! K8> on cpu0
orm0: <ISA Option ROM> at iomem 0xcf800-0xd0fff on isa0
ppc0: cannot reserve I/O port range
sc0: <System console> at flags 0x100 on isa0
sc0: VGA <16 virtual consoles, flags=0x300>
sio0: configured irq 4 not in bitmap of probed irqs 0
sio0: port may not be enabled
sio0: configured irq 4 not in bitmap of probed irqs 0
sio0: port may not be enabled
sio0 at port 0x3f8-0x3ff irq 4 flags 0x10 on isa0
sio0: type 8250 or not responding
sio0: [FILTER]
sio1: configured irq 3 not in bitmap of probed irqs 0
sio1: port may not be enabled
vga0: <Generic ISA VGA> at port 0x3c0-0x3df iomem 0xa0000-
0xbffff on isa0
```

Yukarıdaki kısımda fareye ilişkin bir satır yer alıyor. Dizüstü bilgisayarımdaki touchpad tanınmış. Bir de ayrıca kablosuz USB fare kullanıyorum. Sistem bunu da belirliyor.

```
ums0: <Logitech USB Receiver, class 0/0, rev 2.00/5.00, addr
2> on uhub0
ums0: 16 buttons and Z dir.
```

```
uhid0: <Logitech USB Receiver, class 0/0, rev 2.00/5.00, addr
2> on uhub0

Timecounter "TSC" frequency 1808243961 Hz quality 800
Timecounters tick every 1.000 msec
acd0: DVDR <Slimtype DVD A DS8A1P/CH63> at ata0-master PIO4
ad4: 76319MB <Hitachi HTS541680J9SA00 SB2OC7BP> at ata2-master
SATA150
```

DVD+RW/+CD+RW kombo sürücü de acd0 olarak tanınmış. 80GB disk ad4 olarak tanımlı.

```
hdac0: HDA Codec #0: Conexant CX20549 (Venice)
pcm0: <HDA Conexant CX20549 (Venice) PCM #0 Analog> at cad 0
nid 1 on hdac0
pcm1: <HDA Conexant CX20549 (Venice) PCM #1 Digital> at cad 0
nid 1 on hdac0
```

Ses kartım ile fax/modem tümleşik olduğu için dizüstü bilgisayarlarda aynı yonga ile kontrol edilirler. Ses kartını ise kendim tanıtmış olduğum için çekirdek tarafından kullanılan sürücüsü yüklenmiş ve ses kartım hdac0 olarak tanımlanmış.

Bu aşamadan sonraki mesajlar ise kullandığım sabit diskin özelliklerini belirtiyor. GLABEL kullanıldığı için GEOM_LABEL ile başlayan satırlarda diskteki her bir bölüm belirtiliyor. Kullanılacak olan root bölümü çekirdek tarafından bağlanıyor ve GLABEL etiketleri kaldırılarak açılış sona eriyor.

```
GEOM_LABEL: Label for provider ad4s1a is
ufsid/4a4a3bf02aa49a41.
GEOM_LABEL: Label for provider ad4s1d is
ufsid/4a4a3bf4d20d3886.
GEOM_LABEL: Label for provider ad4s1e is
ufsid/4a4a3bf0a769ccea.
GEOM_LABEL: Label for provider ad4s1f is
ufsid/4a4a3bf026ec3d2b.
Trying to mount root from ufs:/dev/ad4s1a
```

```
GEOM_LABEL: Label ufsid/4a4a3bf02aa49a41 removed.
GEOM_LABEL: Label for provider ad4s1a is
ufsid/4a4a3bf02aa49a41.
GEOM_LABEL: Label ufsid/4a4a3bf0a769ccea removed.
GEOM_LABEL: Label for provider ad4s1e is
ufsid/4a4a3bf0a769ccea.
GEOM_LABEL: Label ufsid/4a4a3bf026ec3d2b removed.
GEOM_LABEL: Label for provider ad4s1f is
ufsid/4a4a3bf026ec3d2b.
GEOM_LABEL: Label ufsid/4a4a3bf4d20d3886 removed.
GEOM_LABEL: Label for provider ad4s1d is
ufsid/4a4a3bf4d20d3886.
GEOM_LABEL: Label ufsid/4a4a3bf02aa49a41 removed.
GEOM_LABEL: Label ufsid/4a4a3bf0a769ccea removed.
GEOM_LABEL: Label ufsid/4a4a3bf026ec3d2b removed.
GEOM_LABEL: Label ufsid/4a4a3bf4d20d3886 removed.
```

Sistemin bu aşamasından sonra ekranda göreceğiniz mesajlar gri renkli olacaktır. Gri renkli mesajlar çekirdeğin root olarak bağladığını ve bu aşamadan sonra 'init'i başlattığını göstermektedir.

init

init çekirdeğin başlattığı ilk süreçtir. BSD sistemlerde init süreç numarası 1 olarak tanımlıdır. init'in görevi çekirdek yüklendikten sonra diğer süreçleri başlatmaktır. Bu süreçler başlatılmadan önce öncelikle disklerin kontrol edilmesi gereklidir.

BSD sisteminizi doğru biçimde kapattıysanız, kapatılma işlemi sırasında sync adlı porgramın çalıştırıldığını görebilirsiniz. sync sistemde bulunan her diski kontrol eder, yazılmamış olan verileri diske yazar ve işi biten dosya sistemini/diski ayırır. başarılı biçimde veri yazılan ve ayrılan dosya sistemlerini "temiz/clean" olarak işaretler.

Dosya Sistemi Bütünlük Kontrolü

init ilk başlatıldığında yaptığı işlem dosya sisteminin temiz olarak işaretlenip işaretlenmediğini kontrol etmektir. Eğer dosya sistemi düzgünce ayrılmamış ise örneğin elektrik kesintisi veya kullanıcının sabırsız davranıp bilgisayarın düğmesine basıp sistemi kapatması gibi durumlarda dosya sistemi temiz olarak işaretlenmez. init bu durumda fsck programını çalıştırıp dosya sistemini hataya karşı tarar. Bulunan hataları fsck giderir. Eğer fsck, dosya sistemindeki hataları otomatik olarak gideremiyorsa bu durumda sistem tek kullanıcı kipine -single user mode- geçer. Böylece sistem yöneticisi gerekli olan işlemleri yapabilir. Dosya sistemi elle düzeltilmek durumundadır.

Dosya sisteminde bir hata yoksa, hata bulunduysa ama fsck tarafından otomatik olarak giderildiyse veya ele giderildi ise init /etc/fstab içinde tanımlı olan dosya sistemlerini bağlar.

Dosya sistemleri bağlandıktan sonra init, sistem yapılandırma dosyalarını çalıştırmaya başlar. İngilizce kaynaklarda system-configuration scripts olarak geçer. BSD ve UNIX kitaplarında ise Run Control, Resource Configuration scripts veya kısaltılmış adı ile rc scripts olarak geçer. Bu programlar /etc ve /etc/defaults altında yer alır. Bu dizinlerden başka /usr/local/etc/rc.d dizini de ayrıca kontrol edilir. Bu dizindekiler temel sistemin parçası olmayan sonradan kurulan uygulamalara ait olan dosyaları barındırır.

Burada bir noktayı açıklığa kavuşturmakta yarar var. Linux sistemlerindeki "çalışma düzeyi" ya da İngilizce adı ile "run level" BSD sistemlerinde bulunmaz. Tersine olarak tek kullanıcı ve çoklu kullanıcı kipi olarak tanımlanan iki farklı çalışma şekli bulunur. BSD sistemlerde Linux sistemlerde olan inittab dosyası da bulunmaz. Sistemin açılışı sırasında çalıştırılacak olan uygulamalar rc.conf dosyasında yeralır. Linux sistemelerinde olduğu gibi farklı çalışma düzeylerini temsil eden farklı dizinlerdeki dosyalar kullanılmaz. Bunu yazının ilerleyen bölümlerinde yeniden ele alacağız.

getty ve login

Sistem yapılandırması gereği çalıştırılacak servisler başlatıldıktan sonra, init, sanal terminalleri başlatır. Bu terminaller üzerinden sisteme giriş yapmak için kullanılan getty adlı programı başlatır. Bazı yapılandırmalarda getty yerine grafik arabirime giriş yapmak için kullanılan xdm yer alır. xdm aslında X ortamına giriş yapmanızı sağlar. X yapılandırmasını ele aldığım yazıda /etc/ttyvs dosyasının düzenlenmesi ile xdm dışında diğer seçeneklerin kullanılmasına yer vermiştim. Eğer X'i başlatmadıysanız veya X'i yapılandırmasıysanız girişi getty ile yapacaksınız demektir.

Ekranda en son satırda göreceğiniz sizden kullanıcı adı ve şifresi girmenizi bekleyen bir satır olacaktır.

```
login:
```

Kurulum sırasında tanımladığınız kullanıcı adınız ve şifreniz ile sisteme giriş yapabilirsiniz. Kullanıcı adını yazdıktan sonra sizden şifrenizi
istenecektir.

```
password:
```

Şifrenizi doğru yazdıysanız günün sözü "motto of the day-motd" ile karşılanacaksınız. Bunun ardından ise yapılandırmaya bağlı olarak tanımlı olan kabuğa giriş yapmış olacaksınız

```
[goksin@droideka:~]$
```

Kullanıcı adınızı ve şifrenizi yanlış yazdıysanız, sizden yeniden giriş yapmanız istenecektir. Bu işlem üç defa tekrarlandıktan sonra sistemin donmuş olduğu izlenimine kapılabilirsiniz. Sistem donmuş veya takılmış değil. Bu özellike BSD sistemlerde güvenlik amacı ile geliştirilmiştir. Kullanıcı adını ve şifrenizi üç defa hatalı girmeniz durumunda sistem girişi kısa bir süre için askıya alır. Bu süre sonunda yeniden giriş yapılmasına izin verir. Eğer yine üç defa hatalı giriş yaparsanız "donma" süresi uzayacaktır. Her üç hatalı girişten sonra sürenin uzaması BSD sistemlere ssh dışında farklı yollardan da giriş yapılabildiği için olası şifre kırma saldırılarının pratik olarak olanaksız duruma getirilmesi içindir. İsteyen seri bağlantı üzerinden deneyebilir.

Çıkış Yapmak - Logout

X11 ortamında olmadığınıza göre ekranda son yaptığınız işlemler görünecektir. Bunları ekrandan clear ile silebilirsiniz. Sistemden çıkış yapmak için hazır olduğunuza göre komut satırına exit yazıp entere basmanız yeterlidir. Çıkış işlemi tamamlandığında yeniden

```
login:
```

yazan satıra geri dönersiniz.

Sistemi Kapatmak

Yukarıda değindiğim gibi BSD veya diğer bir UNIX sistemi kapatacak iseniz bunun doğru biçimde yapılması gerekir. Aksi durumda oluşacak olan hatalar dosya sistemine zarar verip veri kayıplarına neden olabilir. Normal kullanıcıların sistemi kapatma yetkisi bulunmadığı için bu işlemin root olarak yapılması gereklidir. Bunun önüne geçmek için shutdown veya halt kullanmak gereklidir. Burada şu soru akla gelebilir: "İkisi arasındaki fark nedir?"

Halt veya reboot -adından anlaşılıyor ne olduğu- kullanılması önerilmeyen uygulamalardır. Halt ve reboot kullanıcılara uyarı vermeden sistemi sonlandırır.Bu istenmeyen bir durum olarak değerlendirilir. Asıl neden ise halt/reboot uygulamaları çalışmakta olan diğer uygulamalara SIGTERM sinyali yollar, veriyi diske yazar ve tanımlanan seçeneğin gereğini yerine getirir.

Shutdown ise kullanıcılara uyarı göndermek dışında farklı çalışma durumları için örneğin sisteme giriş yapılmasın ama ağ bağlantısı aktif olsun gibi seçenekler ile kullanılabilir. Bu nedenle halt veya reboot yerine shutdown kullanmak yerinde olacaktır.

Kaynak Yapılandırma Dosyaları

init başlatıldıktan sonra sistemin yapılandırılması gereği çalıştırılacak olan diğer servislerin başlatılması gerekir. Bu servisler bazı sunucu uygulamaları yanında işletim sisteminin kendisine ait olan bazı servisler de olabilir. Bu servisleri başlatan yapılandırma dosyaları güncel BSD sistemlerde bir arada bulunurken, eski sürümlerde işlevlerine göre gruplandırılarak /etc/rc.d dizini altında bulunurlar.

/etc/rc.d altında yer alan bu yapılandırma dosyaları aslında birer kabuk programıdır. Bu kabuk programları, sistem yapılandırmanıza göre BSD sistemin özel bir bileşenini başlatır. Bu programlardan bazıları diğer programlar tarafından çağrılarak çalıştırılır. Bu programlardan bazıları sistemi kurduğunuzda çalışmayacaktır. Bazıları ise özel yapılandırma seçenekleri için hazırlanmış olan programlardır. Bu dizinde yer alan tüm programlar init tarafından çalıştırılır ancak kontrol eden ise /etc/rc kabuk programıdır.

BSD sistemlerin yapısı gereği bu yapılandırma programlarını yeniden düzenlemenize gerek yoktur. Bu programlar gerekli servislerin ve 'daemon'ların çalıştırılmasının yanında gerekli diğer bilgilere ulaşacak biçimde yapılandırılmıştır. Bu dosyaların nasıl hazırlandığını öğrenmek isteyenler bir kopyalarını alıp üzerinde çalışabilir. Yapılandırmayı düzenlemek için ise yapılması gereken /etc/rc.conf dosyasının düzenlenmesidir. Söz konusu kabuk programları yapılandırma için bu dosyanın içeriğinden yararlanır.

Bazı kullanıcılar ve geliştiriciler Linux deneyimleri gereği kendi yazdıkları açılış scriptlerini /etc/rc.d altına koyarlar. Buraya kopyaladığınız açılış scriptleri çalışacaktır. Ancak BSD sistemlerin sistemetiğine uygun olarak bunları /usr/local/etc/rc.d dizinine kopyalamak daha uygun olacaktır. /etc/rc.d altındaki programlar temel sisteme ait olanlar iken /usr/local/etc/rc.d altındakiler ise sonradan kurulan programlara aittir. Bu dizinde yer alan programlar /etc/rc.d dizinidekilerin çalıştırılmasının ardından otomatik olarak çalıştırılır. Ayrıca bir düzenleme yapmaya gerek yoktur. Öte yandan olası bir sorun durumunda Linux dağıtımlarının aksine temel sisteme ait olan yapılandırma dosyalarını sonradan kurduğunuz programlara ait olanlardan kolaylıkla ayırabilirsiniz.

rcorder

/etc/rc dizininde yer alan programlar açılışta belirli bir sıra ile çalıştırılır. Bu sıralamayı rcorder ile öğrenebilirsiniz. Aşağıda benim sistemimdeki sıralama görülüyor:

```
[goksin@droideka /usr/home/goksin]$ rcorder /etc/rc.d/*
/etc/rc.d/dumpon
/etc/rc.d/ddb
/etc/rc.d/initrandom
/etc/rc.d/geli
/etc/rc.d/gbde
/etc/rc.d/encswap
/etc/rc.d/ccd
/etc/rc.d/swap1
/etc/rc.d/early.sh
/etc/rc.d/fsck
/etc/rc.d/root
/etc/rc.d/hostid
/etc/rc.d/mdconfig
/etc/rc.d/mountcritlocal
/etc/rc.d/zfs
/etc/rc.d/FILESYSTEMS
/etc/rc.d/var
/etc/rc.d/cleanvar
/etc/rc.d/random
/etc/rc.d/adjkerntz
```

```
/etc/rc.d/atm1
/etc/rc.d/hostname
/etc/rc.d/ipfilter
/etc/rc.d/ipnat
/etc/rc.d/ipfs
/etc/rc.d/kldxref
/etc/rc.d/sppp
/etc/rc.d/addswap
/etc/rc.d/auto_linklocal
/etc/rc.d/sysctl
/etc/rc.d/serial
/etc/rc.d/netif
/etc/rc.d/ip6addrctl
/etc/rc.d/atm2
/etc/rc.d/pfsync
/etc/rc.d/pflog
/etc/rc.d/pf
/etc/rc.d/isdnd
/etc/rc.d/ppp
/etc/rc.d/routing
/etc/rc.d/ip6fw
/etc/rc.d/network_ipv6
/etc/rc.d/devd
/etc/rc.d/ipsec
/etc/rc.d/ipfw
/etc/rc.d/nsswitch
/etc/rc.d/resolv
/etc/rc.d/mroute6d
/etc/rc.d/route6d
/etc/rc.d/mrouted
/etc/rc.d/routed
/etc/rc.d/netoptions
/etc/rc.d/NETWORKING
/etc/rc.d/mountcritremote
/etc/rc.d/devfs
/etc/rc.d/ipmon
/etc/rc.d/mdconfig2
/etc/rc.d/newsyslog
/etc/rc.d/syslogd
/etc/rc.d/savecore
/etc/rc.d/ldconfig
```

```
/etc/rc.d/archdep
/etc/rc.d/abi
/etc/rc.d/SERVERS
/etc/rc.d/named
/etc/rc.d/ntpdate
/etc/rc.d/rpcbind
/etc/rc.d/nisdomain
/etc/rc.d/ypserv
/etc/rc.d/ypxfrd
/etc/rc.d/ypupdated
/etc/rc.d/ypbind
/etc/rc.d/ypset
/etc/rc.d/yppasswdd
/etc/rc.d/wpa_supplicant
/etc/rc.d/accounting
/etc/rc.d/nfsclient
/etc/rc.d/amd
/etc/rc.d/atm3
/etc/rc.d/auditd
/etc/rc.d/tmp
/etc/rc.d/cleartmp
/etc/rc.d/dmesg
/etc/rc.d/ipxrouted
/etc/rc.d/kerberos
/etc/rc.d/kadmind
/etc/rc.d/keyserv
/etc/rc.d/kpasswdd
/etc/rc.d/quota
/etc/rc.d/nfsserver
/etc/rc.d/mountd
/etc/rc.d/nfsd
/etc/rc.d/statd
/etc/rc.d/lockd
/etc/rc.d/pppoed
/etc/rc.d/pwcheck
/etc/rc.d/virecover
/etc/rc.d/DAEMON
/etc/rc.d/watchdogd
/etc/rc.d/ugidfw
/etc/rc.d/timed
/etc/rc.d/apm
```

```
/etc/rc.d/apmd
/etc/rc.d/bootparams
/etc/rc.d/hcsecd
/etc/rc.d/bthidd
/etc/rc.d/local
/etc/rc.d/lpd
/etc/rc.d/motd
/etc/rc.d/mountlate
/etc/rc.d/nscd
/etc/rc.d/ntpd
/etc/rc.d/powerd
/etc/rc.d/rarpd
/etc/rc.d/sdpd
/etc/rc.d/rfcomm_pppd_server
/etc/rc.d/rtadvd
/etc/rc.d/rwho
/etc/rc.d/LOGIN
/etc/rc.d/syscons
/etc/rc.d/sshd
/etc/rc.d/sendmail
/etc/rc.d/cron
/etc/rc.d/jail
/etc/rc.d/localpkg
/etc/rc.d/securelevel
/etc/rc.d/power_profile
/etc/rc.d/othermta
/etc/rc.d/natd
/etc/rc.d/msgs
/etc/rc.d/moused
/etc/rc.d/mixer
/etc/rc.d/inetd
/etc/rc.d/idmapd
/etc/rc.d/hostapd
/etc/rc.d/geli2
/etc/rc.d/ftpd
/etc/rc.d/ftp-proxy
/etc/rc.d/dhclient
/etc/rc.d/bsnmpd
/etc/rc.d/bridge
/etc/rc.d/bluetooth
/etc/rc.d/bgfsck
```

```
[goksin@droideka /usr/home/goksin]$
```

Burada gördüğünüz sıralama programlarda tanımlanan anahtar sözcükler-KEYWORDS- ile öncelikle çalıştırılması gereken-PROVIDE, REQUIRE- diğer tanımlamalara dayanılarak belirlenmektedir. Bu tanımlamalar kabuk programlarının baş tarafında yer almaktadır. Diğer programlara bağımlı olmayanlar öncelikle çalıştırılır. Sonra da bağımlılıklara bakılarak sırası ile diğerleri çalıştırılır. Bu konuda ayrıntılı bilgiyi rc ve rcorder kılavuz -man rcorder ve man rc- sayfalarından edinebilirsiniz.

/etc/defaults/rc.conf Dosyası

Açılış sırasında çalıştırılan kabuk programları eğer sistem yöneticisi tarafından yapılandırılmadıysa genel yapılandırma dosyası esas alınarak çalıştırılır. Bu genel yapılandırma dosyası iki tanedir: /etc/defaults/rc.conf ve /etc/rc.conf. Birinci dosya sistem dosyasıdır ve BSD kurulumunda gelir. Bu dosya üzerinde değişiklik yapılmamalıdır. Yapılacak olan tüm değişiklikler ikinci dosyada /etc/rc.conf dosyasında yapılmalıdır.

FreeBSD kullanmaya başladığım 5.x sürümlerinde temel yapılandırma dosyası /etc/rc.conf idi. /etc/rc kullanılan değişkenler ile ilgili diğer kabuk programlarının yapılandırmasına ilişkin tüm değerler bu dosyada yer alır, sistem yöneticisi gerekli düzenlemeleri bu dosya üzerinde yapardı. Bu yaklaşımın olumsuz yanı olarak kısa zamanda dosyada yapılan düzenlemeler nedeni ile sorunlar baş gösterirdi. Buna rağmen zaman içinde yapılandırmada kullanılan değişkenler ile üstlendiği görevler de giderek arttı. Sistem yöneticisinin sistemi terfi etmesi durumunda bu dosyayının eski sürümü ile yeni sürümünü bir dosyada birleştirmesi gerekirdi. Bu ise dosyayı düzenlemekten daha zor olan bir işlem idi.

Bu sorunun aşılması için durumu geliştirici ekibi, /etc/rc.conf dosyasının bir kopyasını /etc/defaults dizini altına kopyalayarak

çözdüler. Sisteme ait olan standart yapılandırma /etc/defaults altındaki dosyada saklanırken, sistem yöneticisinin yapacağı değişikliklerin yer alacağı dosya ise /etc/rc.conf dosyasına bırakıldı. Bu dosya halen kurulum sırasında sysintsall kullanılarak yapılan yapılandırmayı da barındırır. /etc/rc.conf dosyasında yapacağınız değişiklikler /etc/defaults altında yer alan asıl yapılandırmayı değiştirmeden yeni yapılandırmayı kulanmanızı sağlar. Örneğin cupsd_enable="YES" satırının /etc/rc.conf dosyasına eklenmesi ile cups başlatılabilir. /etc/rc.conf dosyasının içeriğini silip sistemi yeniden başlatabilirsiniz. Bu durumda /etc/defaults yapılandırma dosyasındaki ayarlar kullanılır.

Örnek bir /etc/defaults/rc.conf dosyasını eniXma eki olarak inceleyebilirsiniz..

/etc/defaults/conf doayasına göz attığınızda BSD sistemde bulunan servislere ait olan temel yapılandırmayı görebilirsiniz. Her bir servis için gerekli olan ayarlar ve seçenekler bu dosyada tanımlanmıştır. Örneğin Linux emülasyonu için varsayılan ayar linux_enable="NO" olarak tanımlıdır. Linux emülasyonunu kullanmak için bu değerin /etc/rc.conf dosyasında linux_enable="YES" olarak eklenmesi gereklidir.

Yukarıdaki örnekte gördüğünüz gibi /etc/defaults/rc.conf içinde tek bir dosya içinde gruplanarak yer alır. Böylelikle dosya içinde ilgili değişkenler kolaylıkla bulunur ve /etc/rc.conf dosyasına istenen satırlar kopyalanarak kullanılabilir.

/etc/rc.conf Dosyası

BSD sistemler kurulduğunda sysinstall kurulum sırasında çalıştırılacak olan servislerin yapılandırması için seçenekler sunar. Bu seçenekler rc.conf dosyasına kayıt edilir. Aşağıdaki örnek kendi sistemime aittir.

```
# -- sysinstall generated deltas -- # Thu Jul 16 22:05:01 2009
# Created: Thu Jul 16 22:05:01 2009
```

```
# Enable network daemons for user convenience.
# Please make all changes to this file, not to
/etc/defaults/rc.conf.
# This file now contains just the overrides from
/etc/defaults/rc.conf.
defaultrouter="192.168.7.1"
hostname="droideka.elkotek"
ifconfig_nfe0="inet 192.168.7.4  netmask 255.255.255.0"
keymap="tr.iso9.q"
font8x16=iso09-8x16

# powerd ayarlar
powerd_enable="YES"
powerd_flags="-a adaptive -b adaptive"

# ntpd ayarlar
ntpd_enable="YES"
ntpd_sync_on_start="YES"

#hald ayarlar
hald_enable="YES"

#dbus ayarlar
dbus_enable="YES"

#cups ayarlar
cupsd_enable="YES"

#devfs.rules ayarlar
devfs_system_ruleset="kurallar"
```

Güncel işlemciler çalışma frekansını değiştirirerek farklı hızlarda çalışır. BSD sistemler işlemcinizi belirleyip gerekli çekirdek modüllerini kullanır. Dizüstü bilgisayarınız için güç yönetimi özelliğini kullanmak için powerd daemon kullanılır. Örneğin powerd daemon açılışta aktif olması için /etc/rc.conf içine aşağıdaki gibi bir düzenleme yapılabilir.

```
#
#powerd ayarlar
#
```

```
#
# Eklenme tarihi: 01/07/2009
#
#
# Kılavuz sayfası man powerd
#

powerd_enable="YES"
powerd_flags=" -a adaptive -b adaptive"
```

/etc/rc.conf dosyası üzerinde değişiklik yaparken bazen yapılandırmayı eski haline almak gerekebilir. Bu durumda yukarıdaki örnekte olduğu gibi yorum satırları ekleyerek yapılandırma tarihi, ve diğer bilgileri yazmak sonradan yapılacak düzenlemeler için bir referans oluşturabilir.

Bir diğer seçenek olarak değişiklik yaptığınız tarihteki asıl dosyanın bir kopyasını oluşturup sonradan bu dosyayı geri yüklemek de söz konusu olabilir.

Örneğin dosya üzerinde değişklik yaptığımız tarihi kullanarak bir yedek dosyası oluşturuabiliriz. İşlem yapmadan önce dosyanın bir yedeğini alıyoruz.

```
droideka# cp /etc/rc.conf /etc/rc.conf.yedek.01062009
```

Eski dosyayı geri almak için

```
droideka# mv /etc/rc.conf.yedek.01062009 /etc/rc.conf
```

Yukarıda verdiğim kendi /rc.conf yapılandırmam sistemin kurulumunda belirlediğiniz temel yapılandırma en başta yer alır. Bu kısımda TCP/IP ayarları, sistemin adı, ssh vs diğer yapılandırma seçenekleri yer alır. Sysinstall tarafından yapılan bu düzenlemeleri daha sonra da kendi tercihlerinize göre düzenleyebilirsiniz. Bu düzenlemeleri olduğu gibi bırakmak uygun bir çözüm olacaktır. Sonradan sysinstall ile yapacağınız düzenlemeler ilgili satırların değiştirilmesi ile olur. Eğer bu satırlar üzerinde düzenleme yaparsanız sysinstall bu satırları yeniden ekleyecektir. Böylelikle birbiri ile çelişen iki farklı yapılandırmanız olabilir. Sonradan düzenleme yaparken buna dikkat etmeniz yerinde olur.

/etc/rc.conf dosyalarını sistem yöneticisi düzenleyecektir. Sonradan port üzerinden kurulan uygulamalar kendi yapılandırma dosyalarını /usr/local/etc/rc.conf dosyasına ekleyebilir. Bu durumda sistem yöneticisi uygulamanın eklediği satır üzerinde gerekli gördüğü düzenlemeleri yapabilir. Öte yandan bu durum tüm portlar için geçerli olmayabilir. Bu durumda gereken yapılandırma satırı sistem yöneticisi tarafından eklenmelidir. Ayrıca kurulum sonrası uygulamayı başlatacak olan daemon olarak hazırlanmış olan kabuk programlarının da gerekli dizinlerde uygun izinlere sahip olduğundan emin olunmalıdır.

```
/usr/local/etc
```

Sistem yöneticisi program kurabilen tek kişi olduğu için kurduğu tüm programlar /usr/local dizinine kurulur. Bu durum açılışta çalıştırılan programlar için de geçerlidir. /usr/local/etc dizini bu programlar için /etc dizinin karşılığıdır denebilir. Bu dizine, temel sistemin parçası olmayan tüm programlara ait olan yapılandırma dosyaları kopyalanır.

/usr/local/etc dizini sistem yöneticilerinin kurdukları programlara ait yapılandırma dosyalarını bulacakları ve en önemlisi de düzenleyecekleri yapılandırma dosyalarının bulunduğu dizindir. Veritabanı, web, dosya, ftp vb sunuculara ait olan yapılandırma dosyaları bu dizinde yer alır. Hatta bu uygulamalara ait olan ve açılışta çalıştırılan kabuk programları da buradadır; /usr/local/etc/rc.d. /etc/rc.d dizininde yer alan programlar çalıştırıldıktan sonra bu dizinde bulunanlar çalıştırılır. Çalıştırılan programların uzantıları ".sh"dir. Bu dosyalar alfabetik düzende çalıştırılır. Örneğin sırayla apache.sh, mysql-server.sh vs olarak çalıştırılırlar. Bu kabuk programları kurulan paket veya port tarafından kopyalanır. Başlama ve sona erme komutlarını

kabul ederler. init bunları çalıştırırken "start" komutunu kullanır. Kendiniz de gerektiğinde bu servisleri başlatabilir ve durdurabilirsiniz.

```
# /usr/local/etc/rc.d/apache.sh start
```

Bazı paketlerin ve portların normal .sh uzantılı olmayan ama .sh.default veya .sh.sample uzantılı programlar kopyaladıklarını da görebilirsiniz. Bu dosyalar sistem yöneticisi tarafından düzenlenerek kullanılmak üzere bu farklı uzantılar ile kopyalanır. Sistem yöneticisi gereken düzenlemeleri yaparak bu dosyaların uzantısını .sh biçimine dönüştürüp kullanabilir. Örneğin apache doğrudan apache.sh dosyasını kopyalarken, samba.sh.sample dosyasının düzenlenerek samba.sh olarak kayıt edilip kullanılması gereklidir. Bu düzenleme yapıldıktan sonra çalıştırılabilir.

Bu dizin dışında başka bir dizini kullanmak yine sistem yöneticisinin veya programcının insiyatifinde olan bir durumdur. PCBSD, FreeBSD üzerinde çalışan kendi yapılandırmasını başlatmak için PCBSD init kullanmaktadır. Bunu da /usr/local/etc/rc.d yerine kendi tanımladıkları dizini kullanarak yapmaktadırlar. Benzer bir yapılandırma örneği aşağıda görülmektedir.

```
local_startup="/usr/local/etc/rc.d /herhangi/bir/dizin/rc.d" #
baslangic script dizini.
```

Bu şekidle ayrıca açılış programlarının yer aldığı bir dizin tanımlayabileceğinize göre Linux kullanıcılarının burada eski alışkanlıklarını kullanarak bazı düzenlemeler yapmaları bir güvenlik açığı meydana getirmektedir. Bazı Linux dağıtımları /etc/rc.local dizininde yer alan programları açılışta başlatmaktadır. Bu dizin BSD sistemdeki /usr/local/etc/rc.d karşılığıdır. FreeBSD 6.x öncesinde bu dizin sistemden kaldırıldı ve bugün sistemde yer almıyor.

Ancak bazı durumlarda örneğin uzak bir sistemdeki paylaşılan dizinlerin bağlanması, yedekleme sisteminin başlatılması gibi işlemler için bu dizini oluşturup kullanabilirsiniz. Burada yer alan programlar eski BSD sistemlerce ve Linux dağıtımlarınca açılışta otomatik olarak çalıştırılacaktır.

Bu şekilde olağan sistem yapılandırmasının dışında kalan uygulamaların çalıştırılmasının yaratabileceği sakıncaları gidermek için /usr/local/etc/rc.d içine kılavuz sayfalarında belirtildiği gibi doğru olarak yapılandırılmış programların kopyalanması yerinde olacaktır. Bugün için geriye doğru uyumluluğun sağlanması için halen /etc/rc.local dizinindeki uygulamaları çalıştıran /etc/rc.d/ dizininde bulunan local programı bulunmaktadır. Eğer rc.local kullanacak iseniz, rc.d dizinindekilerden önce bu dizinde yer alan programların çalıştırılacağını akılda bulundurmak gereklidir. Güvenlik ve olası sorunlardan kaçınmak için rc.d sistemetiğini kullanmak en doğrusu olacaktır.

FreeBSD 7.2 bir sistemde yer alan local kabuk programının içeriğini aşağıda görebilirsiniz.

```
[goksin@droideka /etc/rc.d]$ file local
local: Bourne shell script text executable
[goksin@droideka /etc/rc.d]$ cat local
#!/bin/sh
#
# $FreeBSD: src/etc/rc.d/local,v 1.6.10.1.4.1 2009/04/15
03:14:26 kensmith Exp $
#

# PROVIDE: local
# REQUIRE: DAEMON
# BEFORE:  LOGIN
# KEYWORD: shutdown

. /etc/rc.subr

name="local"
start_cmd="local_start"
```

```
stop_cmd="local_stop"

local_start()
{
        echo -n 'Starting local daemons:'
        if [ -f /etc/rc.local ]; then
                . /etc/rc.local
        fi
        echo '.'
}

local_stop()
{
        echo -n 'Shutting down local daemons:'
        if [ -f /etc/rc.shutdown.local ]; then
                . /etc/rc.shutdown.local
        fi
        echo '.'
}

load_rc_config $name
run_rc_command "$1"
[goksin@droideka /etc/rc.d]$
```

Sistem Açılışında Çalıştırılacak Bir Kabuk Programı Hazırlamak

Genel olarak temel sistem ile portlar üzerinden yaptığınız kurulumlar için açılışta çalışacak bir kabuk programı yazmak durumunda değilsinizdir. Temel sistem ve portlar gerekli programları sunar. Bazen çeşitli gerekliliklerden dolayı bu tür programları yazmak gerekebilir. Yazdığınız programın açılış sırasında çalışması için /usr/local/etc/rc.d dizinine kopyalamanız gerekir.

Aşağıdaki örnek man rc kılavuz sayfasından alınmıştır. Bu dosyayı esas alarak kendi gereksinmelerinize uygun bir kabuk programı yazabilirsiniz. init tarafından programınızın çalıştırılabilmesi için uzantısının .sh olması zorunludur.

```
#!/bin/sh
#

# PROVIDE: foo
# REQUIRE: bar_service_required_to_precede_foo
# BEFORE:  baz_service_requiring_foo_to_precede_it
# KEYWORD: FreeBSD

. /etc/rc.subr

name="foo"
rcvar=`set_rcvar`
command="/usr/local/bin/foo"
extra_commands="nop hello"
hello_cmd="echo Hello World."
nop_cmd="do_nop"

do_nop()
{
        echo "I do nothing."

}

load_rc_config $name
run_rc_command "$1"
```

Bu dosyadaki uygun yerleri kendi gereksinmelerinize göre düzenleyebilirsiniz. Yukarıdaki program init tarafından çağrıldığında başlatılırken ve sona erdirilirken kullanılacak olan seçenekleri /etc/rc.subr ile kontrol etmektedir. Bu dosyaya sh_program_00.sh adını verip çalıştırılabilir yaptığınızda kullanılabilecektir. (chmod +x sh_program_00.sh)

Açılışta çalıştırılan programlar kabuk programı olmak durumunda değildir. PERL, C/C++ veya tercih ettiğiniz bir diğer dilde de yazılabilir. Temel sistem tarafından çalıştırılabilecek olan dilleri tercih etmeniz menfaatiniz icabıdır. Ve en önemlisi dosyanızın uzantısının .sh olması da zorunludur. Uzantı belirtmeden yazdığınız programlar itinayla sorun çıkarır.

inetd Daemon inetd.conf Yapılandırması

Temel sistemin bünyesinde yeterli sayıda daemon bulunur. Bu daemonların başvurdukları yapılandırma dosyaları da *d.conf olarak /etc dizininde yer alır. Bu daemonlar dışında bir yapılandırma seçeneği daha bulunur: "Super-sunucu, inetd" inetd görevi tanımlanan ağ portları üzerinden gelene çağrıları dinleyerek gereken ağ servislerini, süreçlerini başlatmaktır.

Örneğin inetd telnet bağlantılarını kontrol eder. Eğer telnet çalıştırıyorsanız uzaktan sisteme telnet ile bağlanabilirsiniz. Bu istekte bulunduğunuzda telnetd başlamadan size bir login sunulur. Sisteme her yapılan telnet bağlantısı isteği (23 port) için ayrı bir telnetd süreci başlatılır. inetd dışındaki bu ve benzeri diğer daemonlar /usr/libexec altında yer alır. PATH/YOL değişkeni içinde tanımlı değildirler ve normal kullanıcılar tarafından çalıştırılamazlar. Bunun yerine diğer bir süreç tarafından -örneğin ağ bağlantısı veya soketler gibi- çağrılır, çalıştırılır ve çoğaltılırlar.

inetd'nin yararı buradadır. root tarafından çalıştırılan bir ana telnetd sürecine duyulan gereksinimi ortadan kaldırır. Özellikle de söz konusu süreçte ortaya çıkan bir güvenlik açığının neden olabileceği tehlikeyi ortadan kaldırır. Söz konusu telnetd dışında olan sshd, httpd,sendmail süreçleri kendileri çalışır ve inetd daemonuna gereksinim duymazlar. Asıl süreç root olarak olarak çalışır ve normal kullanıcılardan gelen istekleri dinleyerek gerektiğinde yeni süreçler oluşturur. Böylece bu süreç normal kullanıcı tarafından kullanılır ve işlem gerçekleşir. Bu yaklaşım esneklik ve hız sağlarkan güvenlik açısından bazı sorunlar yaratır. inetd aynı zamanda root olarak çalışır. Dolayısıyla da inetd kontrolünün yetkisiz birinin eline geçmesi durumu da güvenlik sorunu yaratır. Dolyısıyla ne kadar çok süreç root yetkileri çalışırsa güvenlik riski de o derece artar.

Eğer kurulum sırasında inetd daemonunu aktif kıldıysanız /etc/inetd.conf dosyasını düzenleyerek inetd'yi kendi tercihlerinize göre yapılandırabilirsiniz. Varsayılan yapılandırma bu dosyada yer alan tüm servislerin # ile etkisizleştirilmesidir. Güncel BSD sistemler inetd kullanımını teşvik etmez. telnetd ve ftp gibi güvenli olmayan servisler inetd tarafından çalıştırılır. Eğer bir sunucu olarak kurulumu yaptıysanız bu durumda bu servisleri gereği gibi yapılandırmanız ise size düşmektedir. Bu sunucu servislerini doğrudan güvensiz olarak çalıştırmak yerine port üzerinden daha güvenli olan diğer uygulamaları kullanarak yapılandırmanız yerinde olacaktır.

inetd'yi açılışta ekinleştirmediyseniz /etc/rc.conf dosyasına aşağıdaki satırı ekleyerek yapabilirsiniz.

```
inetd_enable="YES"
```

Aşağıda inetd tarafından kontrol edilen servisler görülmektedir.

```
Servis      Tanımı                  Kullanılan Kaynak/Port

ftp         File Transfer Protocol  Port 21/TCP

telnet      Uzak terminal           Port 23/TCP

comsat      "biff" sunucusu         Port 512/UDP

ntalk       Komut satırı chat       Port 518/TCP, UDP

ftp(IPv6)   File Transfer Protocol  IPv6

telnet(IPv6)      Uzak terminal     IPv6

pop3        Post Office Protocol    Port 110/TCP

imap4       Interim Mail Access Protocol(server-side mail)
Port 143/TCP

smtp        Qmail (SMTP server)     Port 25/TCP
```

```
netbios-ssn Samba dosya paylaşımı   Port 139/TCP

netbios-ns                          Port 137/TCP

finger      Kullanıcı bilgisi       Port 79/TCP
```

Bu servislerden birisini aktif kılmak için önündeki # işaretini kaldırıp inetd'yi yeniden başlatmanız yeterlidir.

inetd, BSD sistemlerin, sıkı biçimde kontrol ettiği ve otomatik olarak gerçekleşen yapılandırma işlemlerinin dışında kalmaktadır. Olası yapılandırma hataları vb sorunları kendiniz çözmeniz gerekecektir. Hatta ve hatta güvenlikten ödün verdiğinizi unutmamanız gereklidir. cvs servisi hatalı bir yapılandırma kullanılması durumunda güvenlik açığına neden olmaktadır. Samba için smbd ve nmbd programlarını /usr/local/sbin altında bulabilirsiniz. Bunlar Samba kurmadığınız sürece o dizinde bulunmayacaktır. Samba inetd ile çalıştırmanız durumunda netbios-ssn ve netbios-ns hizmetleri tamamen standart dışı bir yapılandırma ile çalışacaktır.

Benzer olarak POP3 servisi için gereken dosyalar da /usr/local/libexec dizininde bulunacaktır. Bu dizin /usr/local altında bulunduğuna göre, söz konusu uygulamayı siz kurmadığınız sürece burada olmayacaktır. Bu nedenle inetd aktif kıldığınızda farklı bir pop programı aynı dizine kurulacaktır.Kendiniz portlardan kurulum yaptığınızda popper kurulacaktır. inetd aktif kılıp kurulumun yapılmasını sağladıysanız bu durumda qpopper kurulur. Bu durumda yapılandırmanızın da değiştirilmesi gerekir.

```
pop3    stream    tcp     nowait    root
/usr/local/libexec/qpopper      qpopper
```

Burada yer verdiğimiz bazı olası sorunlar ile çözümleri işini bilen bir sistem yöneticisi tarafından kolaylıkla üstesinden gelinebilir. Ancak konuya hakim olmayan kişilerin yaptıkları kurulumlar ve yapılandırmalar ciddi sorunları da beraberinde getirecektir.

Sistem Kayıtları-syslogd ve syslog.conf Dosyası

Sistem mesajları syslogd-system logger daemon- tarafından /var/log dosyasına kayıt edilir. syslogd'un nasıl çalışacağını belirleyen /etc/syslog.conf dosyasıdır. Bu dosyada farklı servisler/süreçlere ait kayıtların hangi dosyada tutulacağı tanımlanır. Syslogd tarafından izlenen servislerin (auth, authpriv, console, cron, daemon, ftp, kern, lpr, mail, mark, news, ntp, security, syslog, user, uucp, ve local0, local1, ..,local7) hangi mesajların kayıt edileceği ise önem derecesi -severity- ile tanımlanır. Önem derecesi büyükten küçüğe doğru sıraladığımızda emerg (Acil), alert (Uyarı), crit (Kritik), err (Hata), warning(Uyarı), notice(Dikkat), info(Bilgi) ve debug(Hata ayıklama).

BSD sistemde çalıştırdığınız her daemon veya servis tanımladığınız araç üzerinden syslogd tarafından kaydı tutulur. Sendmail veya diğer bir e-posta programı syslodg() yordamlarını kullanarak farklı düzeyde öneme sahip olan mesajlar yollarlar. Bu mesajlar syslogd tarafından syslogd.conf dosyasında tanımlandığı üzere değerlendirilir.

syslogd.conf dosyasının standart içeriği aşağıdaki gibidir:

```
*.err;kern.debug;auth.notice;mail.crit          /dev/console
*.notice;authpriv.none;kern.debug;lpr.info;mail.crit;news.err
/var/log/messages
security.*
/var/log/security
auth.info;authpriv.info
/var/log/auth.log
mail.info
/var/log/maillog
lpr.info
/var/log/lpd-errs

ftp.info
/var/log/xferlog
```

```
cron.*
/var/log/cron
*.=debug
/var/log/debug.log
*.emerg                                         *
```

Bu dosyanın yorumu ise şudur:

Tüm hata mesajları, çekirdekten gelen hata ayıklama mesajları, yetkilendirme mesajları, kritik öneme sahip olan mesajlar konsolda görüntülenecektir. Güvenlik ile ilgili olan mesajlar /var/log/security dosyasına yazılacaktır. Posta programlarının çıktıları /var/log/mail dosyasına yazılacaktır. Geriye kalanlar ise /var/log/messages adlı genel sistem kaydı dosyasına yazılacaktır. Acil-emerg- olan mesajlar ise tüm konsollarda tüm kullanıcılara görüntülenir.

Söz konusu kayıt dosyaları sistemin çalışma süresi ile üzerinde çalışan servislere ve kullanıcı sayısına göre boyutu büyüyecektir. Bu nedenle dosyaların gereksiz yere büyümesini önlemek için kayıt dosyaları önceliklerine göre ya açılış sırasında yeniden oluşturulur veya dosyanın sonuna yeni kayıtlar eklenir. Örneğin /var/log/maillog dosyası periodic programı tarafından arşivlenir. Diğer kayıt dosyaları örneğin /var/log/cron ve /var/log/messages gibi üzerine yazan program tarafından yeniden üzerine yazılır. Arşivlenmiş olan kayıt dosyalarını bzcat ile açıp içerisinden aradığınızı grep ile bulabilirsiniz.

```
# bzcat /var/log/messages.2.bz | grep "rejected"
```

Aşağıdaki tabloda bazı sistem mesajlarının nasıl tanımlanarak syslogd tarafından ne şekilde işleneceğine dair örnekler görülmektedir.

```
------------------------------------------------------------
İmla                    İşlem
------------------------------------------------------------
@sistem.alanadı.net     Mesajlar söz konusu sisteme ağ
                        bağlantısı üzerinden yönlendirilir.

user1                   Mesajlar user1 kullanıcınısın giriş yaptığı
                        terminale yazılır

root,user1,user2        Tanımlı kullanıcılar mesajları alır

*                       Mesajlar sisteme giriş yapan tüm
                        kullanıcılara aktarılır

| "mail root"           Mesajlar root kullanıcısına postalanır.
------------------------------------------------------------
```

Konu ile ilgili ayrıntılı bilgi syslogd ve syslogd.conf kılavuz sayfalarından edinilebilir. (bilgi man syslogd ve man syslogd.conf)

Açılış Sürecini Güvenli Kılmak

Sistem yöneticileri için öncelikli olan konulardan birisi sistemin güvenliğini sağlamaktır. Bu sistem yöneticilerinin standart olarak yaptığı bir işlem olsa da BSD sistemlerin yapısı gereği bir çok sistem yöneticisi tarafından hatalı yapılandırmalar yapılmaktadır.

Bilgisayarınızı bir başka kişi ile paylaşıyorsanız, loader menüsünden komut satırına geçip boot -s komutu ile sistemi tek kullanıcı kipinde açılabilmesi olasılığı söz konusudur. Bu durum bir güvenlik açığı yaratmaktadır. Bu açığı ise kolaylıkla kapatabilirsiniz.

/etc/ttys dosyasında terminal ayarları tanımlanır. Bu ayarlar sisteme farklı kanallardan erişim için sınırlamaları belirtir. Seçtiğiniz erişim kanalı üzerinden erişimi kontrol altına almak için "secure" seçeneğini "insecure" yapmanız yeterlidir.

Standart yapılandırma:

```
ttyv8 "/usr/local/bin/kdm -nodaemon"  xterm   on  secure
```

Güvenlik önlemi alınmış durumu

```
ttyv8 "/usr/local/bin/kdm -nodaemon"  xterm   on  insecure
```

Bu yapılandırma sisteme söz konusu konsol üzerinden yapılacak olan erişimlerde root girişi için her zaman root şifresini istemesini tanımlar. Güvenli-secure- olarak tanımlı bir konsolda root şifresinin istenmeyeceğini anımsayın.

Tek kullanıcı kipi girişlerini kontrol altına aldığınızda bir noktanın daha kontrol altına alınması gereklidir. Sistemin normal olarak başlatılması durumunda sanal konsollar aktif durumda olur. Kullanıcılar bu sanal konsollar arasında ALT+F1, ALT+F2 vs. ile geçiş yapabilirler. Bu konsollardaki secure/insecure seçenekleri tanımlanarak doğrudan root girişi kapatılabilir. Bu sanal konsollar üzerinden kullanıcı adı ve şifresi ile giriş yapıldıktan sonra root girişi için "su" kullanılması sağlanır.

Açılışı Yavaşlatan Ne Olabilir?

Menü karşınıza geldikten sonra bir işlem yapmaz iseniz 10 sn sonra çekirdek başlatılacaktır. Elimdeki eski bilgisayarlada bile BSD sistemlerin açılması 30 sn'den daha uzun sürmemektedir. Bu sürenin daha uzun sürmesi söz konusu değildir, ancak ve ancak bir şeyler ters gitmediyse...

init yapısı gereği açılış için scriptleri kullanması ve bir sürecin bitmeden diğerinin başlatılamaması gereği sürecin dar boğazlarını ve sorunları belirlemeniz son derece kolaydır. En sık karşılaşılan yavaşlamalar Sendmail ve Apache kaynaklıdır. Her ikisinde de yavaşlamanın nedeni ağ yapılandırmasındaki bir hata veya ağdan kaynaklanan bir sorun olabilir.

Sendmail ile apache bilgisayara atanmış olan adı -hostname- bilmek durumundadır. Bunun için de /etc/resolv.conf dosyasında tanımlı olan DNS sunucularını kullanmak durumundadırlar. Bu sorgunun yapılması bazen ağdan kaynaklı nedenlerden dolayı açılış sürecini uzatabilir. Bunu aşmanın bir yolu /etc/resolv.conf dosyasında tanımlı olan DNS sunucusunun sisteminiz tarafından erişilebilen bir makina olmasını sağlamanızdır. Böylece Apache ve Sendmail gecikme olmaksızın sistemin adını belirleyip açılışı yavaşlatmadan devam etmesini sağlar.

Diğer olası sorun ise ağ üzerinden erişilen dosya paylaşımlarıdır. Bunlar NFS, Samba vb olabilir. Eğer bu dosya paylaşımları /etc/fstab içinde otomatik olarak bağlanacak şekilde yapılandırılmış ise açılış sırasında bağlanmaya çalışılacaktır. Bu dosya sistemlerinin bağlanma seçeneğinin noauto olarak tanımlanması ya da sonradan elle veya diğer bir yolla bağlanması uygun olur.

BSD - IX

Süreçlerin kontrolu

Gökşin Akdeniz
goksin@enixma.org

BSD sistemler -temelde sunucu işletim sistemi olarak değelendirilse de, ki masaüstünde de kullanabilirsiniz- süreçleri kontrol etmek için gereken tüm araçlara sahiptir. BSD sistemlerde süreçlerin kontolü ve izlenmesi sistem yönetiminin olağan işlerinden birisidir.

BSD sistemlerde süreçleri kontrol etmek için kullanılan araçlar, süreçlerin kullandığı sistem kaynağını, sürecin önceliğini, faklı sinyallerin süreçlere yollanmasını denetler ve gerekiyorsa sonlandırılıp yeniden başlatılmasına olanak verir. Sistemin bütününü kontrol edebilmek son kullanıcının hedeflendiği işletim sistemleri ile karşılaştırıldığında ilk bakışta BSD sistemlerinin "ilkel sistemler" olarak algılanmasına neden olur. Aslında işini bilen bir sistem yöneticisi için BSD sistemlerin sunduğu esneklik sistemin işleyişini olumsuz etkilemez. Tersine işleyişini kontrol edebilmek için son derece uygundur.

Daha basit olarak bir benzetim ile anlatacak olursak; BSD sistemler -özde bir UNIX sistemdir- şeffaf yapısı ile sistemi oluşturan tüm parçaların kolaylıkla görülebildiği, erişilebildiği sistemlerdir. Bu yapı bize herhangi bir parçaya müdahele etme ve değiştirme hatta ince ayar yapmamıza olanak sağlar. Bu BSD'nin -UNIX'in- doğasıdır.

Bu bölümde sistemin işleyişini izlemek, kontol etmek ve ince ayar yapabilmek için kullandığımız araçlar olan ps, top ve kill üzerinde duracağız. Bunun dışında sistemde bazı süreçlerin düzenli olarak işletilmesini sağlayan cron aracını da ele alacağız.

top ile Sistem Performasını İzlemek

Sistem performansı denildiğinde akla ilk gelen sistemin ne kadar çabuk işlemleri sonuçlandırdığıdır. Aslında performans kavramı yabancı kaynaklarda sistemin işleyişinin ölçülmesi anlamına gelir. Bu nedenle sistemin performansını ölçmek denildiğinde sistemin işleyişine ait sayısal bir değer anlaşılmaktadır. Top ve htop eniXma'nın 13'üncü sayısında kısaca ele alınmıştı. top adından da anlaşılacağı gibi sistemde işleyen süreç listesinin en üstünde yer alanları görüntülemek için geliştirilmiştir. Daha kesin olarak söylecek olursa süreç listesinin üst 10 satırındaki süreçleri listelemek için geliştirilmiştir yani TOP10. top ile çalışmanın yararı, gerçek zamanlı ve etkileşimli çalışmasıdır. Çalıştırıldığında her bir saniyede kendisini günceller ve sistemin o andaki durumuna ait bilgileri sunar. top sadece bilgi sunmaz aynı zamanda sizin verdiğiniz komutları da kabul eder ve gereğini yerine getirir. Bu komutlar dışında belirteceğiniz seçenekler ile döndüreceği süreçleri seçebilirsiniz de. Bu nedenle top, bazı süreçlerin işleyişine ince ayar yapmak için ve bir sunucuda ortaya çıkan bazı sorunların

kaynağını belirlemek için mükemmel bir araç durumuna gelir.

top Çıktısını Yorumlamak

Yeni kurduğunuz bir BSD sistemde top -çalıştırdığınız BSD sistemlerdeki top, Linux dağıtımlarındaki top isim aynı olsa da işleyişi faklıdır- çalıştırdığınızda size yaklaşık olarak 40 civarında sürece ilişkin bilgileri döndürecektir. Bu süreç sayısı çalıştıdığınız servisler ve uygulamalara bağlı olarak değişecektir.

top normal olarak çalıştırıldığında sistemdeki tüm süreçleri listeler. Bu durumu değiştirebilirsiniz. BSD yapılandırmasında bazı güvenlik araçları sistemin bir parçasıdır ve bunların kullanılması ile süreçlerin izlenmesi, kontrol edilmesi gibi bazı işlemler sadece sistem yöneticisinin yetkisine bırakılabilir. Aksi durumda normal bir kullanıcı hesabı ile sisteme giriş yaptığınızda sistemdeki tüm kullanıcılara ait olan süreçleri de top ile izleyebilirsiniz.

Sistemde çalışan süreçler -aktif, boşta veya zombie modunda olabilir- ve bu süreçlerin ne kadar işlemci zamanı tükettikleri standart olarak döndürülür. Top çıktısında göreceğiniz gibi ikinci satırda sistemdeki süreç sayısı gösterilir. Bu sayı sistemden sisteme farklılıklar gösterir. Ekranda ise bu süreçlerin tamamını göremeyebilirsiniz. "I" tuşuna basarak sadece aktif olan süreçleri görüntüleyebilirsiniz. Kullanabileceğimiz diğer seçeneklere ilerleyen bölümlerde yer vereceğiz.

Top kullandığınızda bir diğer ilgi çeken satır ise "load averages" satırıdır. Bu satırda yer alan bilgiler sistem üzerindeki yükü bir diğer deyişle sistemin ne kadar meşgul olduğunu belirtir. Gördüğünüz değerler 1, 5 ve 15 dk önce sistemde çalıştırılan süreçlerin sayısına bağlı olarak hesaplanır. Bu değerlere bakılarak "kesin" bir fikir edinmek zordur. Özellikle de farklı sistemlerde çalıştırılan farklı uygulamalar nedeni ile bu değerler değişkenlik gösterecektir. Sistem yükü 1 civarında olduğunda, sistem çalıştırılan her süreci başlatıldığında işlemekte olduğu anlamına gelir. Değer 1'den büyük ise süreçler başlatıldığında işleme alınmamaktadır, tersine olarak sıraya alınıp sırası gelen süreç işleniyor demektir.

```
last pid: 60325;  load averages:  2.46,  2.28,  2.21            up 0+12:15:36  12:00:13
132 processes: 4 running, 128 sleeping
CPU: 91.5% user,  0.0% nice,  7.3% system,  0.4% interrupt,  0.8% idle
Mem: 1106M Active, 1972M Inact, 587M Wired, 84M Cache, 399M Buf, 181M Free
Swap: 4096M Total, 52K Used, 4096M Free

  PID USERNAME    THR PRI NICE   SIZE    RES STATE  C   TIME   WCPU COMMAND
59897 root          1 106    0   319M   308M CPU0   0   0:14 45.65% cc1plus
  810 root          1  51    0   391M   318M select 1  18:18 12.50% Xorg
60147 goksin        1  47    0 85080K 26400K select 1   0:01  0.29% ksnapshot
  927 goksin        1  45    0   133M 40304K select 1   3:03  0.00% kdeinit
  970 goksin        1  44    0 87688K 25348K select 1   1:39  0.00% kdeinit
 1210 goksin        3  44    0   121M 41432K ucond  1   1:09  0.00% ktorrent
  939 goksin        1  44    0   117M 34088K select 1   1:02  0.00% kdeinit
 6422 goksin        1  44    0 93688K 30408K select 0   0:59  0.00% kdeinit
  971 goksin        1  44    0 91640K 27472K select 1   0:15  0.00% kdeinit
  958 goksin        1  60 r16F 58808K 11312K select 1   0:14  0.00% artsd
  705 root          1  44    0  5692K  1020K select 1   0:13  0.00% powerd
  997 goksin        1  44    0  9140K  2920K CPU1   1   0:12  0.00% top
  373 root          1  44    0  6820K  1044K select 0   0:09  0.00% moused
30880 goksin        1  44    0   116M 37760K select 0   0:09  0.00% kdeinit
  937 goksin        1  44    0 98100K 30096K select 0   0:07  0.00% kdeinit
  935 goksin        1  44    0 88664K 25180K select 1   0:06  0.00% kdeinit
  811 haldaemon     1  44    0 21756K  4536K select 1   0:06  0.00% hald
  843 root          1  44    0 11328K  1616K select 1   0:04  0.00% hald-addon-storage
  902 goksin        1  44    0 11824K  2636K select 1   0:04  0.00% gam_server
  943 goksin        1  44    0  9096K  1628K select 1   0:03  0.00% ksysguardd
  922 goksin        1  44    0 78592K 18676K select 1   0:02  0.00% kdeinit
11944 root          1   8    0 25736K 11800K wait   0   0:02  0.00% perl5.8.9
 1030 goksin        1  44    0 80756K 23408K select 0   0:02  0.00% kweatherservice
  832 root          1  44    0 11328K  1608K select 1   0:02  0.00% hald-addon-storage
  826 root          1  44    0 11328K  1608K select 1   0:02  0.00% hald-addon-storage
  835 root          1  44    0 11328K  1608K select 1   0:02  0.00% hald-addon-storage
  829 root          1  44    0 11328K  1608K select 1   0:02  0.00% hald-addon-storage
  840 root          1  44    0 11328K  1608K select 1   0:02  0.00% hald-addon-storage
  984 goksin        1  44    0 84728K 23412K select 0   0:02  0.00% kdeinit
 3540 goksin        1  44    0   105M 37440K select 0   0:02  0.00% kchmviewer
  698 root          1  44    0 10484K  1960K select 1   0:02  0.00% ntpd
 1038 goksin        1  44    0 83640K 24852K select 0   0:01  0.00% rssservice
  988 goksin        1  44    0   114M 30460K select 0   0:01  0.00% kdeinit
  919 goksin        1  44    0 80144K 19940K select 1   0:01  0.00% kdeinit
  990 goksin        1  44    0 92328K 24936K select 0   0:01  0.00% korgac
  976 goksin        1  44    0 79584K 22340K select 0   0:00  0.00% kwikdisk
 1102 goksin        1  44    0 84464K 23376K select 0   0:00  0.00% kttsd
```

Resim 1 - BSD'de top ekranı

Bu değerler, bir masaüstü sistem için bir çok grafik arayüz vb ile çeşitli

uygulamaların çalışması durumunda ortalama olarak sistem yükü 2 veya 3 arasında değişebilir. Eğer bu değer 5 ve üzerine ulaşıyorsa ciddi bir sorun olduğuna işaret eder. Sistem yükü ortalamasının 5 civarında olması kararsız çalışma durumuna işaret eder. Bu durumda sistemde çalışan bazı daemonlar -örneğin sendmail ortalama yük 12 olduğunda gelen istekleri kabul etmez- gelen istekleri kabul etmeyecektir. Eğer yük ortalaması 20 veya 30 civarında ise bu durumda sistemde var olan süreçler tamamlanamadan yenileri başlatılıyor demektir. Bu duruma feedback-loop situation veya race condition adı verilir. Devam etmesi durumunda sistem çok yavaşlayacaktır. Bu derecede aşırı yüklenmiş sitem durumuna death-spiral denilir. Adından da anlaşılacağı üzere sistem çökme durumuna gelir.

Bu durumlar UNIX sistemlerde ender olarak görülür ancak ortaya çıkması durumunda süreçlerin bazılarının sona erdirilmesi veya yeniden düzenlenmesi ile sorun giderilemez. Hatta sisteme uzaktan bağlanıp sorunu çözme için ssh oturumu açsanız da sistem buna yanıt vermeyebilir. Sistemin tüm süreçleri tamamlayıp sonuçlandırmasını beklemek iyi niyetli bir yaklaşım olsa da bu süre çok uzun olacaktır. Bunların yerine sistemi yeniden başlatmak en kısa ve kolay çözümdür.

top'u çalıştırdığınızda terminal üzerinde sisteme ilişkin bilgiyi görebilirsiniz. Bu bilgi top çalıştığı sırada sistemde çalışmakta olan süreçler ile bu süreçlerin kullandıkları sistem kaynağı miktarıdır. Sistemdeki RAM miktarını üst bölümde yer alan dördüncü ve beşinci satırlarda görebilirsiniz. Bu bilgi UNIX sistemlere özgü biçiminde gösterilmektedir. Anlaşılmaz gelebilir. UNIX sistemlerde bellek ve kullanılan bellek arasında kesin bir ayrım yoktur. Bunun nedeni UNIX'in bellek yönetiminin sisteminizde bulunan fiziksel bellek miktarına değil, sistemin erişebildiği sanal bellek üzerine kurulu olmasıdır. Sanal bellek ise sisteme kurulum yaparken oluşturduğunuz takas alanıdır. Sanal belleğin işlevi ise aktif olarak kullanılmayan verinin RAM yerine takas alanında tutulmasını sağlamaktır.

Sistemde bulunan fiziksel belleğe ne kadar uygulama sığdırabiliriz diye düşünüp uygulamaların gereksinim duyduğu bellek miktarını teker teker belirleyip aritmetik olarak toplamakla uğraşmak durumunda değiliz. UNIX bu işi bizim yerimize başarılı bir şekilde yapmaktadır zaten. Aktif olarak kullanılan veri RAM'de tutulurken diğerleri fiziksel belleğin yetmediği durumlarda takas alanına aktarılır ve gerektiğinde buradan okunup yeniden fiziksel belleğe aktarılır. BSD sisteminizi tek sistem olarak kurduysanız takas alanı diskin okuma yazma hızının en yüksek olduğu disk plaklanının en uç kısmında oluşturulur. Bunu sizin için sysinstall yapar.

top çıktısının beşinci satırlarında gördüğünüz "free" ile gösterilen değer sistemde kullanılabilecek olan serbest bellek miktarını belirtir. Bu değer sistemin başlatılmasının ardından o ana dek kullanılmamış yeni erişilmemiş olan bellek miktarını verir. Genel olarak bir çok kullanıcı bu değere bakar. Bu değer öncelikli olarak bakılacak olan değer değildir. Onun yerine "active" ile belirtilen değer daha çok bilgi verir. Bu değer sistemde çalışan süreçler atrafından kullanılan bellek miktarını belirtir. Diğer alanlar da bellek kullanımına ilişkin diğer bilgiyi verir. Bu değerleri aritmetik olarak toplayarak sistemdeki bellek miktarını elde etmeniz söz konusu değildir. Büyük bir olasılıkla da bu toplam sistemde bulunan fiziksel bellekten büyük olacaktır.

Takas kullanımına ilişkin bilgiyi veren alan olan "Swap" daha kesin bilgi verir. Bu alandaki "Used" ve "Free" ile gösterilen değerler kullanılan takas alanı ile serbest olan takas alanını belirtir ve toplamları daima takas alanının büyüklüğüne eşittir. Takas ile ilgili olan bilgileri yorumlamak daha kolaydır ve sistemin işleyişine ilişkin önemli ipuçları verir.

Takas kullanımına baktığınızda kullanılan takas alanının yaklaşık olarak %50 veya üzeri olduğunu görebilirsiniz. Bu nadiren karşılaşılan bir durumdur. Fiziksel belleğinizin -RAM- dolu olduğu ve sık sık verinin

takas ve fiziksel bellek arasında aktarıldığını gösterir. Bu durumda sistemde bulunan RAM miktarını arttırmak gereklidir. Burada takas alanının sonuna dek kullanılıp kullanılmayacağı sorusu akla gelebilir. BSD sistemler takas alanının tamamını kullanarak tüketmezler. Fiziksel bellekten az olan takas alanı ancak sistemde bir coredump olması durumunda sorun yaratır. Bu konuya önceki yazılarda yer vermiştik. Takas alanının coredump dışındaki durumlarda tükenmesi durumunda BSD sistem size hata mesajı verecektir, sistem kararlılığında bir değişme ortaya çıkmayacak ama bu durumu yaratan koşulların devam etmesi durumunda çalışan uygulamalarda sorunlar ortaya çıkması söz konusu olacaktır. Bunun önüne geçmek için RAM'ın yetersiz kalmaya başladığı durumlarda yeni RAM eklemek zorunlu olacaktır. SWAP bir disk alanı olduğu için fiziksel belleğiniz-RAM- kadar hızlı değildir ve sistemin hızını düşüren bir etkiye neden olur.

Alt kısımda göreceğiniz tabloda ise, o sırada sistemde çalışmakta olan süreçlere ait olan bilgi görüntülenir. Bu bilgi liste halinde sunurlur ve WCPU değeri en büyük olandan en küçük olana doğru sıralanır. WCPU-Weighted CPU kısaltmasıdır. İşlemci hızı, frekans olarak belirtilen değerler ile tanımlanır. Kaynaklarda "CPU Cycle" olarak tanımlanır. örneğin 4 MHz'lik bir işlemci saniyede 4 milyon adet CPU cycle yapabilmektedir. Buradaki saniye ise sistemdeki saatinizin belirttiği saniye değeridir. AMD işlemciler, Intel işlemciler ile karşılaştırıldığında aynı saniye aralığında bazı işlemleri daha kısa sürede yapabilmektedir. Bir işlemcinin frekansı veya hızı yukarıda belirtildiği gibi bir saniyedeki cycle sayısıdır. Bazı süreçler daha az cycle ile tamamlanırken bazıları daha fazlasına gerek duyarlar. Bu nedenle bazı süreçler saniye bazında ölçülebilecek kadar işlemci frekansı kullandıkları için saniye bazında ölçülebilir. Bu değer "Time" sütununda görüntülenir. Göreceğiniz zaman değerleri xx:xx: formatında olacaktır. Bu gösterim saat:dakika olarak hatalı yorumlanır. Halbuki bu değer sistem:kullanıcı olarak yorumlanmalıdır. Yukarıda değinildiği gibi süreçler WCPU sütununda azalan şekilde sıralandığı için ağırlıklı olarak süreçlerin "resident-sahip" olarak kullandığı işlemci frekansına göre listelenir. Resident state adı verilen işlemciye yüklenmiş olan kodun çalıştırıldığı frekans sayısının değerini verir. Bu sütunda yer alan değerlerin aritmetik toplamının %100 olması zorunlu değildir. Alt kısımda yer alan "CPU:" satırında ise farklı durumlarda, system-idle -sistem boşta-, user-kullanıcı ve nice-düzenleme, kullanılan işlemci frekansı miktarları belirtilir. Bu değerler WCPU sütunundaki değerlerle ilişkilidir.

Diğer sütunlar olan SİZE ve RES sütunları dikkat edilmesi gereken diğer alanlardır. SIZE sütunu değeri bir sürecin kullandığı veri-data, text-metin ve stack-yığıt değerlerinin toplamıdır. Bu değerler sistem genelinde kullanılan kaynakların bileşimidir. Dolayısıyla da bir sürecin kullandığı bellek miktarını kesin olarak vermez. Bir diğer deyişle bu değerler sanal bellek- virtual memory değeridir. RES sütunu ise sürecin kullanmakta olduğu fiziksel bellek miktarını verir. Bu değer resident memory veya resident set size-RSS olarak da değerlendirilir.

C sütunu eğer sistemde birden fazla işlemciniz varsa, sürecin hangi işlemciyi kullanmakta olduğunu gösterir. PID, process ID kısaltmasıdır ve sürecin numarasını gösterir. USERNAME ise süreci başlatan kullanıcının adını belirtir. STATE sütunu sürecin durumunu belirtir. Bu sütundaki değer zomb/zombie olarak tanımlı ise süreç sona erdirilmiş ancak süreç tablosundan-process table çıkarılmamış demektir.

Etkileşimli top Kullanımı

top komutunu çalıştırdığınızda terminal üzerinde çok sayıda sürece ait olan veriyi listeleyecektir. Bu süreçlerden bazıları aktif olarak çalışmıyor olabilir. Bu durumda aktif olarak çalışan süreçleri listelemek daha uygun olacaktır. top eski sayıda yer verildiği gibi etkileşimli olarak kullanılabilir. I tuşuna basarak sadece aktif olan süreçleri

izlemeye alabilirsiniz. Bir kullanıcıya ait süreçleri izlemek için U tuşuna basarak kullanıcı adını tanımlayarak ilgili kullanıcının sistemde çalıştırdığı süreçleri izlemeye başlayalabilirsiniz. Eğer bir süreci sonlandırmak isterseniz K tuşuna basıp ardından ilgili sürecin PID değerini vermeniz durumunda girdiğiniz PID değerine sahip olan süreç sonlandırılır. Diğer seçenekler ise top kılavuz sayfasında belirtilmiştir.

Temel olarak sistemin genel durumuna ilişkin bilgi edinmek için top mükemmel bir araçtır. Süreçlerin işleyişi için gerekli olan yönetimsel araçları da sunar ama herşeyi yapan bir araç değildir. Ayrıca etkileşimli olması nedeni ile sıklıkla kullanılan kabuk programlarında kullanılması uygun bir araç olmamaktadır. Bu nedenle top yerine ps kullanılır. ps, top'un yetersiz kaldığı yerlerde devreye girer.

ps ile Süreçlerin İzlenmesi

top ile karşılaştırıldığında, ps etkileşimli olarak çalışmayan fakat o andaki sistem süreçlerine ait bilgiyi veren bir uygulamadır. top ile elde ettiğiniz bilgiler yanında diğer ayrıntıları da sunar. herhangi bir seçenek vermeden çalıştırdığınızda sadece komutu veren kullanıcıya ait olan terminaller üzerinde çalışan süreçlere ait bilgi veririr. Söz konusu terminal sisteme giriş yaptığınızda çalıştırdığınız terminaldir. ps'in sunduğu bilgileri ayrıntılandırmak veya bilgi edinmek istediğiniz süreçleri tanımlamak için ilgili seçenekler ile birlikte kullanılmalıdır. Bu seçeneklerin tamamını ps kılavuz sayfasında bulabilirsiniz. ps ile en yaygın kullanılan seçenekler aux'tur. "ps aux" sık olarak kullanacağınız komut olacaktır. Seçenekler ise; a tüm kullanıcılara ait olan süreçleri döndür, u süreçleri başlatan kullanıcı isimlerini döndür, x kendi terminaliniz dışındakileri de döndür demektir.

Komutun çıktısını ekranda okuyabilirsiniz ancak çıktı terminal genişliğiniz ile sınırlı olacak ve çıktının bir bölmünü okumanız söz konusu olmayacaktır. Bunu önlemek için ps aux komutunu w seçeneği ile birlikte kullanırsak bu durumda terminal boyutlarından bağımsız olarak çıktı 132 karakter genişlikte olacaktır. Bu durumda bile tüm bilgiyi göremeyebilirsiniz. Terminal boyutlarından bağımsız ve 132 karakter sınırlaması olmaksızın çıktı elde edebilmek, yani gerektiğinde alt satırlara geçen ve devam eden bir çıktı elde etmek için ise w seçeneği iki defa kullanılmalıdır. Aşağıdaki örnekte çıktının bir bölümü alt satırlara taşacak şekilde gösterilmektedir.

```
[goksin@droideka /usr/home/goksin]$  ps wwaux
USER    PID %CPU %MEM   VSZ   RSS  TT  STAT STARTED   TIME COMMAND
root     12 91,6  0,0     0    16  ??  RL   11:45PM 419:14,46 [idle: cpu0]
root     11 85,0  0,0     0    16  ??  RL   11:45PM 424:07,66 [idle: cpu1]
root    810  6,6  7,6 401388 315992  ??  R    11:45PM  36:42,00 /usr/local/bin/X
-br -nolisten tcp :0 -auth /var/run/xauth/A:0-KNE0Nq (Xorg)
root      0  0,0  0,0     0     0  ??  DLs  11:45PM   0:00,13 [swapper]
root      1  0,0  0,0  2180   368  ??  ILs  11:45PM   0:00,23 /sbin/init --
root      2  0,0  0,0     0    16  ??  DL   11:45PM   0:02,02 [g_event]
root      3  0,0  0,0     0    16  ??  DL   11:45PM   0:49,69 [g_up]
root      4  0,0  0,0     0    16  ??  DL   11:45PM   0:40,21 [g_down]
root      5  0,0  0,0     0    16  ??  DL   11:45PM   0:00,00 [xpt_thrd]
root      6  0,0  0,0     0    16  ??  DL   11:45PM   0:00,00 [kqueue taskq]
root      7  0,0  0,0     0    16  ??  DL   11:45PM   0:00,00 [acpi_task_0]
root      8  0,0  0,0     0    16  ??  DL   11:45PM   0:00,00 [acpi_task_1]
root      9  0,0  0,0     0    16  ??  DL   11:45PM   0:00,00 [acpi_task_2]
root     10  0,0  0,0     0    16  ??  DL   11:45PM   0:00,00 [audit]
root     13  0,0  0,0     0    16  ??  WL   11:45PM   0:00,64 [swi1: net]
root     14  0,0  0,0     0    16  ??  WL   11:45PM   1:57,71 [swi4: clock sio]
root     15  0,0  0,0     0    16  ??  WL   11:45PM   0:00,00 [swi3: vm]
root     16  0,0  0,0     0    16  ??  DL   11:45PM   0:05,24 [yarrow]
root     17  0,0  0,0     0    16  ??  WL   11:45PM   0:01,48 [swi2: cambio]
root     18  0,0  0,0     0    16  ??  WL   11:45PM   0:00,00 [swi5: +]
root     19  0,0  0,0     0    16  ??  DL   11:45PM   0:00,00 [thread taskq]
root     20  0,0  0,0     0    16  ??  WL   11:45PM   0:00,59 [swi6: Giant taskq]
root     21  0,0  0,0     0    16  ??  WL   11:45PM   0:02,54 [swi6: task queue]
root     22  0,0  0,0     0    16  ??  WL   11:45PM   0:00,00 [irq9: acpi0]
root     23  0,0  0,0     0    16  ??  WL   11:45PM   0:43,20 [irq23: hdac0 ohci0]
root     24  0,0  0,0     0    16  ??  DL   11:45PM   0:00,00 [usb0]
root     25  0,0  0,0     0    16  ??  DL   11:45PM   0:00,00 [usbtask-hc]
root     26  0,0  0,0     0    16  ??  DL   11:45PM   0:00,00 [usbtask-dr]
root     27  0,0  0,0     0    16  ??  WL   11:45PM   0:03,68 [irq22: ehci0]
root     28  0,0  0,0     0    16  ??  DL   11:45PM   0:00,01 [usb1]
```

```
root          29  0,0  0,0     0    16  ??  DL   11:45PM   0:28,82 [nfe0 taskq]
root          30  0,0  0,0     0    16  ??  WL   11:45PM   0:18,54 [irq20: atapci0]
root          31  0,0  0,0     0    16  ??  WL   11:45PM   0:00,00 [irq21: atapci1]
root          32  0,0  0,0     0    16  ??  WL   11:45PM   0:00,00 [irq16: hdac1 drm0]
root          33  0,0  0,0     0    16  ??  DL   11:45PM   0:01,56 [acpi_thermal]
root          34  0,0  0,0     0    16  ??  DL   11:45PM   0:00,03 [acpi_cooling0]
root          35  0,0  0,0     0    16  ??  WL   11:45PM   0:03,74 [irq1: atkbd0]
root          36  0,0  0,0     0    16  ??  WL   11:45PM   0:00,00 [swi0: sio]
root          37  0,0  0,0     0    24  ??  DL   11:45PM   0:00,00 [sctp_iterator]
root          38  0,0  0,0     0    16  ??  DL   11:45PM   0:01,70 [pagedaemon]
root          39  0,0  0,0     0    16  ??  DL   11:45PM   0:00,00 [vmdaemon]
root          40  0,0  0,0     0    16  ??  DL   11:45PM   0:00,00 [pagezero]
root          41  0,0  0,0     0    16  ??  DL   11:45PM   0:03,59 [bufdaemon]
root          42  0,0  0,0     0    16  ??  DL   11:45PM   1:15,66 [syncer]
root          43  0,0  0,0     0    16  ??  DL   11:45PM   0:02,29 [vnlru]
root          44  0,0  0,0     0    16  ??  DL   11:45PM   0:08,46 [softdepflush]
root         136  0,0  0,0  2568   816  ??  Is   11:45PM   0:00,00 adjkerntz -i
root         373  0,0  0,0  6820  1044  ??  Ss   11:45PM   0:19,83 /usr/sbin/moused -p
/dev/ums0 -t auto -I /var/run/moused.ums0.pid
root         421  0,0  0,0  2180   484  ??  Is   11:45PM   0:00,01 /sbin/devd
root         546  0,0  0,0  5692  1180  ??  Ss   11:45PM   0:00,08 /usr/sbin/syslogd -s
messagebus   620  0,0  0,0  6896  1936  ??  Is   11:45PM   0:00,12
/usr/local/bin/dbus-daemon --system
root         639  0,0  0,1 17700  2916  ??  Is   11:45PM   0:00,01 /usr/local/sbin/cupsd
root         698  0,0  0,0 10484  1960  ??  Ss   11:45PM   0:02,86 /usr/sbin/ntpd -g -c
/etc/ntp.conf -p /var/run/ntpd.pid -f /var/db/ntpd.drift
root         705  0,0  0,0  5692  1012  ??  Ss   11:45PM   0:23,15 /usr/sbin/powerd -a
adaptive -b adaptive
root         739  0,0  0,1 10700  3056  ??  Ss   11:45PM   0:00,65 sendmail: accepting
connections (sendmail)
smmsp        743  0,0  0,1 10700  2808  ??  Is   11:45PM   0:00,02 sendmail: Queue
runner@00:30:00 for /var/spool/clientmqueue (sendmail)
root         749  0,0  0,0  6748  1232  ??  Ss   11:45PM   0:00,12 /usr/sbin/cron -s
root         803  0,0  0,0 14480  1736  ??  I    11:45PM   0:00,01 /usr/local/bin/kdm-bin
-nodaemon ttyv8
haldaemon    811  0,0  0,1 21756  4480  ??  Ss   11:45PM   0:08,56
/usr/local/sbin/hald
root         814  0,0  0,1 22104  3260  ??  Is   11:45PM   0:00,02
/usr/local/sbin/console-kit-daemon
root         815  0,0  0,1 15540  2216  ??  I    11:45PM   0:00,04 hald-runner
root         823  0,0  0,0 17380  1884  ??  I    11:45PM   0:00,01 hald-addon-mouse-
sysmouse: /dev/ums0 (hald-addon-mouse-sy)
root         826  0,0  0,0 11328  1612  ??  S    11:45PM   0:03,79 hald-addon-storage:
/dev/da0 (hald-addon-storage)
root         829  0,0  0,0 11328  1612  ??  S    11:45PM   0:03,52 hald-addon-storage:
/dev/da1 (hald-addon-storage)
root         832  0,0  0,0 11328  1612  ??  R    11:45PM   0:04,10 hald-addon-storage:
/dev/da2 (hald-addon-storage)
root         835  0,0  0,0 11328  1612  ??  S    11:45PM   0:03,91 hald-addon-storage:
/dev/da3 (hald-addon-storage)
root         840  0,0  0,0 11328  1612  ??  S    11:45PM   0:03,36 hald-addon-storage:
/dev/da4 (hald-addon-storage)
root         843  0,0  0,0 11328  1620  ??  S    11:45PM   0:07,41 hald-addon-storage:
/dev/cd0 (hald-addon-storage)
root         854  0,0  0,1 22948  2316  ??  I    11:45PM   0:00,01 kdm-bin: :0 (kdm-bin)
goksin       868  0,0  0,0  7064  1604  ??  Is   11:45PM   0:00,01 /bin/sh
/usr/local/bin/startkde
goksin       902  0,0  0,1 11824  2628  ??  R    11:45PM   0:07,20
/usr/local/libexec/gam_server
goksin       919  0,0  0,5 80144 19844  ??  Ss   11:45PM   0:02,12 kdeinit: kdeinit
Running... (kdeinit)
goksin       922  0,0  0,4 78592 18620  ??  S    11:45PM   0:05,32 kdeinit: kdeinit:
dcopserver --nosid (kdeinit)
goksin       925  0,0  0,5 82880 21324  ??  S    11:45PM   0:00,70 kdeinit: kdeinit:
klauncher --new-startup (kdeinit)
goksin       927  0,0  1,1 140504 44740  ??  S    11:45PM   5:37,47 kdeinit: kdeinit:
kded --new-startup (kdeinit)
goksin       932  0,0  0,0  5840   972  ??  S    11:45PM   0:00,66 kwrapper ksmserver
goksin       934  0,0  0,5 83496 22128  ??  S    11:45PM   0:00,42 kdeinit: kdeinit:
ksmserver (kdeinit)
goksin       935  0,0  0,6 91064 26840  ??  S    11:45PM   0:13,97 kdeinit: kdeinit:
kwin -session
101651b01ab1a000012500786160000000995000_1250282167_4337 (kdeinit)
goksin       937  0,0  0,7 98116 30224  ??  S    11:45PM   0:13,58 kdeinit: kdeinit:
kdesktop (kdeinit)
goksin       939  0,0  0,9 122388 36408  ??  S    11:45PM   1:56,45 kdeinit: kdeinit:
kicker (kdeinit)
goksin       942  0,0  0,0  7064  1592  ??  I    11:45PM   0:00,00 /bin/sh -c
ksysguardd
goksin       943  0,0  0,0  9096  1604  ??  S    11:45PM   0:04,93 ksysguardd
goksin       958  0,0  0,3 58808 11340  ??  S    11:45PM   0:32,58 /usr/local/bin/artsd
-F 10 -S 4096 -s 60 -m artsmessage -c drkonqi -l 3 -f
```

goksin 967 0,0 0,5 84504 22128 ?? S 11:45PM 0:00,40 kdeinit: kdeinit: kaccess (kdeinit)
goksin 970 0,0 0,6 87688 25292 ?? S 11:45PM 3:00,72 kdeinit: kdeinit: kmix -session 101651b01ab1a000012500786380000000995001 5_1250282156_977919 (kdeinit)
goksin 971 0,0 0,7 91640 27656 ?? S 11:45PM 0:27,19 kdeinit: kdeinit: konsole -session 101651b01ab1a000012500831640000000846001 2_1250282156_978029 (kdeinit)
goksin 976 0,0 0,5 79584 22228 ?? S 11:45PM 0:00,61 kwikdisk -session 101651b01ab1a000012502781550000000934003 8_1250282156_981329
goksin 984 0,0 0,6 85752 23320 ?? S 11:45PM 0:03,12 kdeinit: kdeinit: klipper (kdeinit)
goksin 988 0,0 0,7 116800 30496 ?? S 11:45PM 0:02,18 kdeinit: kdeinit: knotify (kdeinit)
goksin 990 0,0 0,6 92328 24796 ?? S 11:45PM 0:00,81 korgac --miniicon korganizer
goksin 1030 0,0 0,6 80772 23332 ?? I 11:46PM 0:04,63 kweatherservice
goksin 1038 0,0 0,6 83656 24532 ?? I 11:46PM 0:02,13 rssservice
goksin 1102 0,0 0,6 84464 23252 ?? I 11:46PM 0:00,70 kttsd
goksin 1173 0,0 0,0 14416 1908 ?? I 11:47PM 0:00,00 dbus-launch --autolaunch 0a56d7e66b20277e076e0dd74a7c1524 --binary-syntax --close-stderr
goksin 1174 0,0 0,0 6896 1624 ?? Is 11:47PM 0:00,02 /usr/local/bin/dbus-daemon --fork --print-pid 5 --print-address 7 --session
goksin 1212 0,0 0,6 87192 25400 ?? I 11:47PM 0:00,61 kdeinit: kdeinit: kio_uiserver (kdeinit)
goksin 42636 0,0 0,3 63044 11784 ?? I 12:37AM 0:00,00 /usr/local/bin/kdesud
goksin 53913 0,0 2,0 221716 82888 ?? I 8:10PM 0:47,80 kontact
goksin 53937 0,0 0,5 81404 20256 ?? S 8:10PM 0:00,58 kdeinit: kdeinit: kio_file file /tmp/ksocket-goksin/klauncherQSS5SY.slave-socket /tmp/ksocket-goksin/KWeatherService4fyvNU.slave-socket (kdeinit)
goksin 54069 0,0 0,6 106888 24328 ?? I 8:11PM 0:00,18 kdeinit: kdeinit: kio_pop3 pop3 /tmp/ksocket-goksin/klauncherQSS5SY.slave-socket /tmp/ksocket-goksin/kontact03SbN7.slave-socket (kdeinit)
goksin 54378 0,0 0,9 180680 35784 ?? S 8:16PM 0:04,02 pidgin
goksin 54446 0,0 0,9 117828 36884 ?? S 8:17PM 0:05,04 kdeinit: kdeinit: kate (kdeinit)
goksin 54458 0,0 0,9 102968 36488 ?? I 8:17PM 0:00,83 kchmviewer
goksin 54611 0,0 0,0 7064 1596 ?? I 8:19PM 0:00,00 /bin/sh /usr/local/bin/firefox3
goksin 54615 0,0 0,0 7064 1608 ?? I 8:19PM 0:00,00 /bin/sh /usr/local/lib/firefox3/run-mozilla.sh /usr/local/lib/firefox3/firefox-bin
goksin 54619 0,0 1,7 179056 69028 ?? I 8:19PM 0:03,84 /usr/local/lib/firefox3/firefox-bin
goksin 54623 0,0 0,1 21508 3924 ?? I 8:19PM 0:00,01 /usr/local/libexec/gconfd-2
goksin 54727 0,0 1,1 127608 44816 ?? R 8:21PM 0:14,49 ktorrent -icon ktorrent -miniicon ktorrent -caption KTorrent
goksin 54735 0,0 0,6 94236 23056 ?? I 8:21PM 0:00,04 kdeinit: kdeinit: kio_http http /tmp/ksocket-goksin/klauncherQSS5SY.slave-socket /tmp/ksocket-goksin/ktorrentmn3ZdS.slave-socket (kdeinit)
goksin 54737 0,0 0,6 94236 23036 ?? S 8:21PM 0:00,04 kdeinit: kdeinit: kio_http http /tmp/ksocket-goksin/klauncherQSS5SY.slave-socket /tmp/ksocket-goksin/ktorrent3pJo2b.slave-socket (kdeinit)
goksin 54738 0,0 0,6 94236 23024 ?? I 8:21PM 0:00,03 kdeinit: kdeinit: kio_http http /tmp/ksocket-goksin/klauncherQSS5SY.slave-socket /tmp/ksocket-goksin/ktorrentBWRhMH.slave-socket (kdeinit)
goksin 54764 0,0 1,4 215128 57308 ?? I 8:21PM 0:06,38 amarokapp
goksin 54770 0,0 0,5 82548 21496 ?? I 8:21PM 0:00,01 kdeinit: kdeinit: kio_file file /tmp/ksocket-goksin/klauncherQSS5SY.slave-socket /tmp/ksocket-goksin/amarokZsP0K3.slave-socket (kdeinit)
goksin 54774 0,0 0,1 12068 3964 ?? I 8:21PM 0:00,02 ruby /usr/local/share/apps/amarok/scripts/score_default/score_default.rb
goksin 55128 0,0 0,6 106376 24028 ?? I 8:27PM 0:00,10 kdeinit: kdeinit: kio_pop3 pop3s /tmp/ksocket-goksin/klauncherQSS5SY.slave-socket /tmp/ksocket-goksin/kontactVbehrE.slave-socket (kdeinit)
goksin 55131 0,0 0,7 90648 27200 ?? S 8:27PM 0:01,97 kaudiocreator
goksin 55140 0,0 0,5 82612 21824 ?? I 8:27PM 0:00,02 kdeinit: kdeinit: kio_media media /tmp/ksocket-goksin/klauncherQSS5SY.slave-socket /tmp/ksocket-goksin/kio_systemrentuW.slave-socket (kdeinit)
goksin 55147 0,0 0,6 108412 26256 ?? I 8:27PM 0:00,32 kdeinit: kdeinit: kio_audiocd audiocd /tmp/ksocket-goksin/klauncherQSS5SY.slave-socket /tmp/ksocket-goksin/kio_mediahVNMFr.slave-socket (kdeinit)
goksin 55170 0,0 0,6 106760 24064 ?? I 8:27PM 0:00,10 kdeinit: kdeinit: kio_pop3 pop3 /tmp/ksocket-goksin/klauncherQSS5SY.slave-socket /tmp/ksocket-goksin/kontactoy8uH7.slave-socket (kdeinit)
goksin 55178 0,0 0,8 115580 31920 ?? DL 8:27PM 0:01,22 kdeinit: kdeinit: kio_audiocd audiocd /tmp/ksocket-goksin/klauncherQSS5SY.slave-socket /tmp/ksocket-goksin/kaudiocreatorhQpjWP.slave-socket (kdeinit)
goksin 55410 0,0 0,7 91672 27528 ?? S 8:31PM 0:00,27 kdeinit: kdeinit: konsole --type shell (kdeinit)
root 795 0,0 0,0 5688 1028 v0 Is+ 11:45PM 0:00,00 /usr/libexec/getty Pc

```
ttyv0
root       796  0,0  0,0  5688  1028  v1  Is+  11:45PM   0:00,01 /usr/libexec/getty Pc
ttyv1
root       797  0,0  0,0  5688  1028  v2  Is+  11:45PM   0:00,00 /usr/libexec/getty Pc
ttyv2
root       798  0,0  0,0  5688  1028  v3  Is+  11:45PM   0:00,00 /usr/libexec/getty Pc
ttyv3
root       799  0,0  0,0  5688  1028  v4  Is+  11:45PM   0:00,00 /usr/libexec/getty Pc
ttyv4
root       800  0,0  0,0  5688  1028  v5  Is+  11:45PM   0:00,00 /usr/libexec/getty Pc
ttyv5
root       801  0,0  0,0  5688  1028  v6  Is+  11:45PM   0:00,00 /usr/libexec/getty Pc
ttyv6
root       802  0,0  0,0  5688  1028  v7  Is+  11:45PM   0:00,00 /usr/libexec/getty Pc
ttyv7
goksin      974  0,0  0,1  9020  2296  p0  Is   11:45PM   0:00,01 /usr/local/bin/bash
goksin      997  0,0  0,1  9140  2908  p0  S+   11:46PM   0:38,19 top
goksin    55412  0,0  0,1  9020  2396  p2  Ss    8:31PM   0:00,01 /usr/local/bin/bash
goksin    55447  0,0  0,0  6792  1476  p2  R+    8:32PM   0:00,00 ps wwwaux
[goksin@droideka /usr/home/goksin]$
```

Bu çıktı içinden aradığını bulmak için | grep "aradığınız_her_ne_ise" ile birlikte kullanılması gerekir. Bu şekilde aradığınız kullanıcıya veya uygulamaya ilişkin olan çıktıyı elde edersiniz.

ps çıktıları top'un aksine PID değerlerine göre küçükten büyüğe doğru sıralanır. top ile elde ettiğiniz bilgileri size ps'de sağlar. top ile karşılaştırıldığında kısmen daha fazla bilgi sunar ve kabuk programları veya tek satırlık komut dizilerinde pipe (|) ile birlikte kullanılarak eş zamanlı sistem kontrolünde etkin olarak kullanılabilir. Özellikle de bazı süreçleri kontrol altına almak ve iyileştirmek veya sonlandırmak için bu tür tek satırlık komut dizileri mükemmel olmaktadır.

ps ne zaman kullanılmalı?

Sistemin performansına ilişkin genel bir değerlendirmeye gereksinim varsa top doğru araçtır. ps ise belli bir andaki sistemdeki süreçlerin bir görüntüsünü elde etmek için uygundur. PID kullanarak belli bir sürece veya süreçlere ait bilgi edinecek iseniz ps doğru araç olacaktır. Kabuk programları ile birlikte veya yukarıda söz ettiğimiz gibi tek satırlık komut dizileri ile kullanacağınız araç ise yine ps olacaktır. Örneğin belli bir sürecin önceliğini değiştirmek, sona erdirmek vb. gibi işlemler için ps kullanılmalıdır.

Süreçlerin Sonlandırılması

BSD sistemleri sunucu dışında masaüstü sistemlerde de kullanırken sıklıkla karşılaşılabilecek bir durum kullanıcılardan gelen talepler doğrultusunda port üzerinden bazı uygulamaların ağ üzerinden uzak sistemlere kurmak durumunda kalınmasıdır. Bu durumlarda kullanıcıların root yetkileri olmadığı için sistemdeki root kullanıcı hesabının kontrolü yöneticide olur. En sık karşılaşılan durum kullanıcıların sistem üzerinde kendi yazdıkları uygulamaları kullanarak sistemi yavaşlatmalarıdır. Sisteme uzaktan bağlanıp port üzerinden uygulama kurarken bu tür durumlarda sorun çıkaran uygulamaları sonlandırmak ve gerekiyorsa kullanıcının PATH değişkenlerinin yeniden düzenlenmesi de gerekebilir. Aşağıdaki örnekte bir kullanıcının python kullanarak yazdığı gkrellm benzeri bir uygulamanın neden olduğu kaynak kullanımının top ile gözlenmesi görülüyor. Komutun çıktısının sadece ilgili bölümünü aşağıda görebilirsiniz.

```
last pid: 67469;  load averages:  8.32,  5.49,  2.47   up 53+01:04:22  20:34:42
90 processes:  1 running, 88 sleeping, 1 zombie
CPU states:  93.2% user,  0.0% nice,  0.2% system,  0.8% interrupt, 5.8% idle
Mem: 253M Active, 53M Inact, 128M Wired, 124M Cache, 35M Buf, 5112K Free
Swap: 1024M Total, 244M Used, 780M Free, 24% Inuse

PID USERNAME THR PRI NICE SIZE RES STATE  C   TIME   WCPU COMMAND
19460 murat        1   2     0 25908K  2816K poll   0 131:15 92.43% krelm.py
67468 root         1  28     0  2036K  1024K CPU0   0   0:00  0.20% top
```

Sistemi yavaşlatacak kadar kaynak tüketen uygulama krelm.py adlı uygulama. Python ile program yazanların en sık düştükleri hata budur ve genel olarak sistem kaynaklarını tüketen uygulamalar yazarlar.

Bunu düzgün olarak yazabilirsiniz ancak deneme yanılmadan daha iyi bir öğretmen olamaz.

Yukarıdaki çıktıda gördüğünüz durumda, programın kullanması gerekenden daha fazla belleği kullanmaya çalışması ile ortaya çıkmış bir durum olması da söz konusu olabilir. Hatta yazılan kod içindeki bir döngü sonsuz döngü durumundadır ve sona ermeyeceği için program daha fazla kaynak tüketmeye başlamış olabilir. Bu sürecin sona ereceğini beklemek diğer uygulamaların çalışmasını zora sokacak veya hatta tamamem engelleyecek bir duruma gelmesine de neden olabilir. Bu durumda krelm.py'nin sona erdirilmesi gereklidir.

top ile Sürecin Sonlandırılması

Bu süreci sonlandırmak üzere K tuşuna bastığımızda kill çalıştırılacaktır. kill kullanmak için doğrudan program PID değerini vermek gerekir. Yukarıdaki örnekte 19460 numaralı süreci root olarak sona erdirebiliriz. Süreci sonlandırdıktan sonra "murat" kullanıcısına durumu açıklayan bir e-posta mesajı yollayabiliriz, telefon ile uyarabilir eğer daha da kızarsak kullanıcıyı silebiliriz. İşin şakası bir yana bu tür işler yapacak kullanıcılarınız bulunuyorsa bu durumda önlem almak yerinde olacaktır.

top üzerinden kill kullanırken, asıl kill aracını kullanmıyoruz. Sadece top kendi içinde yer alan ve kill işlevlerinin sadece bir kısmına sahip olan bir kill komutu kullanmaktadır. Bu nedenle top içinden kill'i çalıştırdığınızda her zaman süreç sonlandırılmayabilir. Bu durumda normal yöntemle kill kullanmak doğru olacaktır.

kill ile Süreçleri Sonlandırmak

kill'i anlatırken İngilizce bilen öğrenciler hep süreci sonlandırıyor diye atılırlar. Aslında kill süreci sonlandırır ama yaptığı sadece bu değildir. kill işlev olarak geniş bir yelpazede farklı sinyaller yollayarak süreçleri yönetmektedir. Normal kullanıcılar kendi süreçlerini kill ile sona erdirebilir ancak diğer kulanıcıların süreçlerine müdahale edemez. Bu yetki sadece root kullanıcısına aittir. Bir süreci sonlandırmanın en uygun yolu kill ile süreç numarasını tanımlayarak komutu vermektir.

```
# kill 12345
```

Bu komut doğrudan süreci sonlandırır. UNIX sistemlerde programların bildiği ve anladığı TERM sinyalini yollar. Diğer seçenekler yani sinyalleri aşağıdaki tabloda görebilirsiniz.

kill ile yaygın olarak kullanılan sinyallerler

Sinyal	İsim	İşlevi
1	HUP	Hang-up/sonlandır ve yeniden başlat
2	INT	Interrupt/kesinti
3	QUIT	Quit-Çık
6	ABRT	Abort-Vaz geç
9	KILL	Non-ignorable kill-Program kesin olarak sonlandırılır
14	ALRM	Alarm-Uyarı
15	TERM	Terminate cleanly-Düzgünce sonlandırır.

Bu sinyalleri doğrudan adı ile veya numarası ile birlikte kullanabilirsiniz.

```
# kill -HUP 1234
```

Bu komut 1234 numaralı sürece kendisini düzgünce sonlandırmasını ve sonrasında aynı seçenekler ile ve varsa yapılandırma dosyasını vb. okuyarak yeni PID alarak yeniden başlatılmasını sağlar. Bu komut özellikle yapılandırma dosyalarında yaptığınız değişikliğin geçerli kılıp yeni yapılandırma ile uygulamayı yeniden başlatmak için kullanılır.

```
# kill -9 4321
```

Bu komut ise en son çare olarak kullanılmalıdır. Kullandığınızda ise

süreç sonlandırılır ve geride artıkları kalır. Bu artıklar sürece ait olan soketler ve/veya süreç tarafından başlatılan alt süreçler olabilir. Bunlar sonlandırılmamış olacaktır. Bu nedenle en son çözüm olarak kullanılması gerekir. Öte yandan kill -9 <PID> bazen işe yaramayabilir. Bir süreç zombie durumunda olabilir. Her ne kadar kill -9 ile sonlandırmaya çalışsanız da sinyale yanıt vermeyebilir. Bu durumda denemeye devam edebilirsiniz ancak denemeler işe yaramıyoersa sistemi yeniden başlatmanız gerekecektir.

nice ve renice ile Süreçlerin Önceliğini Belirleyin

Süreçleri kontrol etmek için ps, top ve kill dışında nice ve renice da kullanabileceğiniz diğer araçlardır. nice ve renice bir sürecin önceliğini değiştirir. Sürecin önceliği -20 ile 20 arasında değişen tamsayı değerleridir. top çıktısında NICE sütunu döndürülen çıktıda görülen süreçlerin önceliğini belirtir. Normal koşullarda tüm süreçlerin önceliğinin 0 olduğunu görürsünüz. Bu standart değerdir. Normal koşullarda herhangi bir düzenlemeye gerek duyulmayacağı için sıfır standart değerdir. Bazı durumlarda ise bazı süreçlerin önceliği değiştirilerek ya ön sıralara ya da tersi olarak işlem sırasında gerilere kaydırılabilir.

Süreçlerin önceliğini değiştirmede root kullanıcısı en yetkili olanıdır. Normal kullanıcılar kendi süreçlerine 0 ile -20 arasındaki değerleri atayamaz ama 0 ile 20 arasındaki bir değeri atayabilir. Örneğin kendimize ait olan bir sürece renice ile 10 değerini atayabiliriz.

```
[goksin@droideka /usr/home/goksin]$  renice 10 12345
```

top üzerinden renice kullanmak için R tuşuna basmak gerekir. Bu durumda yetkiniz dahilinde atayacağınız değeri ve ardından sürecin PID'sini girmeniz gereklidir. Yukarıdaki örnekte gördüğünüz krelm.py uygulamasına top üzerinden renice ile 20 değerini atayabilir ve işlem sırasında en sona kaydırabiliriz de. Bu durumda top çıktısında NICE sütununda ilgili sürece ait olan değer root olarak atadığımız değeri gösterecektir.

nice, /usr/bin altında yer alan bir araçtır. Aynı zamanda csh ve tcsh kabuklarının da bir işlevidir. Herhangi bir komutun önceliğini belirterek çalıştırmanızı sağlar.

```
[goksin@droideka /usr/bin]$ ls -l nice
-r-xr-xr-x  1 root  wheel  7208  1 May 09:06 nice
[goksin@droideka /usr/bin]$
```

nice ile bir uygulamanın önceliğini tanımlayarak çalıştırmak için

```
[goksin@droideka /usr/home/goksin]$ nice -5 ls
```

Bu durumda ls, 5 öncelikle çalışmaya başlayacaktır. -5 olarak yazılmış olmasına rağmen +5 olarak yorumlanacaktır. Bu normal kullanıcı hesapları için olağandır. root olarak bir sistemde çalışırken nice kullanarak bir sürecin önceliğini yüksek tutarak başlatmak için komutun aşağıdaki gibi çift "-" ile yazılması gereklidir.

```
# nice --5 ls
```

Bu kullanım şekli ise sadece /usr/bin/nice için geçerlidir. BSD sistemlerde root hesabı için standart kabuk csh olduğu için bu durumu göz önüne alarak

```
# /usr/bin/nice --5 ls
```

olarak çalıştırılması gereklidir.

cron ile Süreç Otomasyonu

Windows kullanıcıları "zamanlanmış görevler" ile tanıştıklarında bunun aslında var olan bir aracın hem de 1988'de ilk yazıldığı günden yaklaşık olarak 10 yıl sonra Microsoft tarafından kopyalanarak yeni bir araçmış gibi sunulması ile tanışmış oldular. Aslında bu "zamanlanmış

görevler" uygulması UNIX sistemlerde Paul Vixie tarafından yazılmış ve kullanılmaya başlanmıştı. Sistem yöneticilerinin günlük olarak gerçekleştirdikleri işlemleri bir düzen içerisinde otomatik olarak gerçekleştirilmesini sağlayan araç olan cron sadece sistem yöneticisinin kullanımına sunulmuş bir araç değildir. Aynı zamanda sistemde tanımlı olan kullanıcılar tarafından da kullanılabilir.

BSD sistemlerde kurulumu tamamladığınızda cron sisteminizde hazır ve çalışır durumdadır. FreeBSD'de cron önceden hazırlanarak sisteme dahil edilmiş bazı uygulamalara ait olan takvimi işletmeye başlar. Bu takvim dahilinde güvenlik kontrolü, sistem durumu, sistemdeki tanımlı yolların güncellenmesi vb. diğer görevler bulunur. Bu görevler ise sistem yöneticisi tarafından yapılacak rutin işlerdir ve root yetkileri ile gerçekleştirilip /var/mail/root dosyasına sonuçları raporlanır. BSD ve UNIX-benzeri sistemlerde cron bir sistem servisi-daemon- olarak çalışır. Çalıştırılacak olan uygulama veya uygulamaları önceden belirlenen tarihte çalıştırır. cron için çalıştırılacak olan programlar ile belirlenen takvim cron tarafından crontab dosyasından okunur. cron bir sistem servisi olduğu için crontab dosyasına eklenen bir girdi için cron yeniden başlatılmak durumunda değildir. crontab dosyası düzenli olarak cron tarafından kontrol edilerek çalıştırılacak olan bir uygulama olup olmadığı kontrol edilir. Bu nedenle cron'un yeniden başlatılması gerekli değildir.

crontab dosyası dediğimizde /etc/crontab dosyasından söz ediyoruz demektir. Bu dosya sistem tarafından çalıştırılacak olan süreçlerin/uygulamaların tanımlandığı dosyadır. Bu dosyasının içeriği üzerinde düzenleme yetkisi root kullanıcısına aittir. Bu dosyaya erişim yetkisi root'a ait olduğu için birçok kullanıcı ve acemi sistem yöneticisi doğrudan bu dosyaya kendi girdilerini eklemek ister. Bu işlemde /etc/rc.local yerine /usr/etc/local/rc.local kullanılması gerektiği için /etc/crontab dosyasında bir işlem yapılmamalıdır. Sistemi sonradan terfi ettiğinizde sistem yapılandırması bu dosyaya yazılacaktır. Bunun yerine root olarak çalıştırılmasını istediğiniz süreçler/uygulamalar için /var/cron/tabs dosyasının kullanılması gereklidir.

crontab Dosyası Nasıl Düzenlenir?

cron, çalıştırılacak olan süreçler için crontab dosyasının içeriğini okur. Aşağıda /etc/crontab dosyasının içeriğini görüyorsunuz:

```
# /etc/crontab - root's crontab for FreeBSD
#
# $FreeBSD: src/etc/crontab,v 1.32.34.1 2009/04/15 03:14:26 kensmith Exp $
#
SHELL=/bin/sh
PATH=/etc:/bin:/sbin:/usr/bin:/usr/sbin
HOME=/var/log
#
#minute hour    mday    month   wday    who     command
#
*/5     *       *       *       *       root    /usr/libexec/atrun
#
# Save some entropy so that /dev/random can re-seed on boot.
*/11    *       *       *       *       operator /usr/libexec/save-entropy
#
# Rotate log files every hour, if necessary.
0       *       *       *       *       root    newsyslog
#
# Perform daily/weekly/monthly maintenance.
1       3       *       *       *       root    periodic daily
15      4       *       *       6       root    periodic weekly
30      5       1       *       *       root    periodic monthly
#
# Adjust the time zone if the CMOS clock keeps local time, as opposed to
# UTC time.  See adjkerntz(8) for details.
1,31    0-5     *       *       *       root    adjkerntz -a
```

Sistem yapılandırması gereği atrun, entropy, log dosyalarının yedeklenmesi, günlük, haftalık ve aylık bakım işlemleri ile sistem saatinin yapılandırması yer alıyor. Bu dosya üzerinde işlem yapmayacağız ancak dosyanın içeriğindeki düzenleme /var/cron/tabs /kullanıcı_adı dosyasının içeriğini oluşturmakta kullanmak için esas

alınacağı için üzerinde durmakta yarar var.

Dosyanın içeriğinde gördüğünüz aşağıdaki satır düzenleme biçimi hakkında bilgi veriyor.

```
#minute hour    mday    month   wday    who     command
```

ilk değer olan minute-dakika, hour-saat, mday-ayın hangi günü, month-ay, wday-haftanın hangi günü, who-kim ve command-komut çalıştırılacak olan komutu tanımlamaktadır. Mantık son derece basittir: dakika, saat, ay-gün, ay, haftanın-günü. Bu tanımlamalarda kullanılan değerler ise belli bir aralıkta olacak olan sayısal değerlerdir:

Minute(Dakika)	0-59
Hour(saat)	0-23
Day of month(ayın günü)	1-31
Month (ay)	1-12
Day of week(haftanın günü)	0-7 (Pazar = 0)

Kullanıcılar tarafından kullanılacak olan bir örnek dosyada ise bu satırlar ve bir sütun yer almayacaktır. Çünkü her kullanıcı kendisine ait bir crontab dosyası kullanacaktır. Sisteme ait olan /etc/crontab dosyası sisteme aittir ve çalıştırılacak olan komutun sahibi ve komutu çalıştıracak olan kullanıcı tanımlıdır. Sistemdeki kullanıcılara ait olan dosya ise yukarıda değindiğimiz gibi kullanıcı_adı ile tanımlı olduğu için bu sütün yer almaz. Aşağıdaki örnek dosya /var/cron/tabs/goksin dosyasının içeriğidir.

```
droideka# cat /var/cron/tabs/goksin
# DO NOT EDIT THIS FILE - edit the master and reinstall.
# (/tmp/crontab.c3RcgBomoL installed on Tue Aug 18 19:03:07 2009)
# (Cron version -- $FreeBSD: src/usr.sbin/cron/crontab/crontab.c,v 1.24.8.1
2009/04/15 03:14:26 kensmith Exp $)
# deneme
0 10,20 * * *   /usr/home/goksin/bin/kontrol.pl
droideka#
```

Bu dosyayı sonradan düzenlerken işimizi kolaylaşacaktırmak için aşağıdaki gibi yeniden düzenliyoruz.

```
# DO NOT EDIT THIS FILE - edit the master and reinstall.
# (/tmp/crontab.c3RcgBomoL installed on Tue Aug 18 19:03:07 2009)
# (Cron version -- $FreeBSD: src/usr.sbin/cron/crontab/crontab.c,v 1.24.8.1
2009/04/15 03:14:26 kensmith Exp $)
#
#minute hour mday month wday command
#
#dakika saat ayin_gunu ay haftanin_gunu komut
#
# deneme
0 10,20 * * *   /usr/home/goksin/bin/kontrol.pl
```

Bu dosyada tanımlanan kontrol.pl dosyası her gün saat 10:00 ve 20:00'de çalıştırılacaktır. Ancak bunu her gün yerine örneğin Pazartesi, Çarşamba ve Cuma günleri çalıştırmak istersek şu şekilde düzenlememiz gereklidir:

```
0 10,20 * * 1,3,5   /usr/home/goksin/bin/kontrol.pl
```

Hatta bu görevi Sadece Eylül ayı boyunca çalıştıralım:

```
0 10,20 * 8 1,3,5   /usr/home/goksin/bin/kontrol.pl
```

Gördüğünüz "*" karakteri, söz konusu zaman değişkeninin her değerini temsil eder. (Düzenli ifadelerdeki *) Eğer çalıştırılacak olan komut her beş dakikada bir çalıştırılacak ise dakika sütunu

```
0,5,10,15,20,25,30,35,40,45,50,55
```

olarak tanımlanmalıdır. Aynı zaman dilimine ait olan birden fazla değeri "," ile ayırarak tanımlayabilirsiniz. Her beş dakikada bir tekrarlanan işlemler için ise uzun uzun yazmak yerine "/" kullanılabilir. Bu durumda her beş dakikada bir ifadesi "*/5" olarak tanımlanabilir. Bu ve diğerleri için cron kılavuz sayfasına gözatabilirsiniz.

Zaman sütunundaki değerleri sayısal olarak girmek yerine bazı kısaltmalar ile aynı tanımlamalar yapılabilir. Bu değerleri aşağıda görebilirsiniz:

Dizge	Karşılığı
@reboot	Sistem her yeniden başladığında
@yearly	0 0 1 1 * (yılda bir defa)
@annually	@yearly ile aynıdır.
@monthly	0 0 1 * * (aylık)
@weekly	0 0 * * 0 (haftalık)
@daily	0 0 * * * (günlük)
@midnight	@daily ile aynıdır.
@hourly	0 * * * * (saatlik)

Zaman sütunlarında olduğu gibi komut sütununda yer alan komutlar da birden fazla olabilir. Komutları ayırmak için ise ";" kullanılmalıdır. /etc/crontab olanın aksine /var/cron/tabs/kullanıcı_Adı dosyalarında dizinler tanımlı değildir. Bu nedenle çalıştırılacak olan uygulama /komut vs. için tam yolu tanımlamanız gereklidir. Bu çalıştırılan komutların bir çıktısı oluyorsa, çıktıları dosyanın sahibi olan kişiye e-posta ile iletilir. /etc/crontab dosyası için bu root kullanıcısıdır. Örneğin tam yolun tanımlanmayıp sadece kontrol.pl olarak tanımlandığında cron işlemi sonuçlandıramayacak ve aşağıdaki e-posta mesajını yollayacaktır:

```
You have mail.
droideka# cd /var/mail
droideka# cat goksin
From goksin@droideka.elkotek Tue Aug 18 20:00:07 2009
Return-Path: <goksin@droideka.elkotek>
Received: from droideka.elkotek (localhost [127.0.0.1])
        by droideka.elkotek (8.14.3/8.14.3) with ESMTP id n7IH05G1036571
        for <goksin@droideka.elkotek>; Tue, 18 Aug 2009 20:00:07 +0300 (EEST)
        (envelope-from goksin@droideka.elkotek)
Received: (from goksin@localhost)
        by droideka.elkotek (8.14.3/8.14.3/Submit) id n7IH04Aj036557;
        Tue, 18 Aug 2009 20:00:04 +0300 (EEST)
        (envelope-from goksin)
Date: Tue, 18 Aug 2009 20:00:04 +0300 (EEST)
Message-Id: <200908181700.n7IH04Aj036557@droideka.elkotek>
From: goksin@droideka.elkotek (Cron Daemon)
To: goksin@droideka.elkotek
Subject: Cron <goksin@droideka> kontrol.pl
X-Cron-Env: <SHELL=/bin/sh>
X-Cron-Env: <HOME=/home/goksin>
X-Cron-Env: <PATH=/usr/bin:/bin>
X-Cron-Env: <LOGNAME=goksin>
X-Cron-Env: <USER=goksin>

kontrol.pl: not found

droideka#
```

crontab Dosyalarını Oluşturmak ve Düzenlemek

/etc/crontab dosyasını düzenlememeniz gerektiğini biliyorsunuz. Bu nedenle kendi sisteminiz için veya kendi kullanıcılarınız için özel bir crontab dosyası oluşturmak gerekecektir. Kullanıcıya ait olan crontab dosyası /var/cron/tabs/kullanıcı_adı adlı dosya olduğu için sadece kullanıcısı tarafından düzenlenebilir. Root olarak /var/cron/tabs dizinine gözattığınızda aşağıdakine benzer bir çıktı görebilirsiniz:

```
droideka# ls -al
total 6
drwx------  2 root  wheel  512 18 Ağu 19:03 .
drwxr-x---  3 root  wheel  512  1 May 10:27 ..
-rw-------  1 root  wheel  466 18 Ağu 19:03 goksin
droideka#
```

goksin kullanıcısına ait olan dosya sadece goksin kullanıcısı tarafından düzenlenebilir. Root kullanıcısı kullanıcıya ait olan crontab dosyasını düzenleyebilir. Kullanıcı yani goksin kullanıcısı da bu dosyayı düzenleyebilir ama sistemdeki diğer kullanıcılar bu dosyaya yazamaz. Kullanıcıya ait olan dosya için izin 0600 olarak tanımlıdır ve her kullanıcı kendi dosyasına yazabilir ve okuyabilir. Bu BSD sistemler

için standart bir uygulamadır. Kullanıcılar kendi crontab dosyalarını düzenlemek için $VISUAL değişkeninde tanımlı olan düzenleyiciyi kullanmak durumundadır. BSD sistemlerde eğer aksine bir düzenleme yapılmadıysa bu düzenleyici vi metin editörüdür.

at ile Tek Sefer Çalışacak Süreç Tanımlamak

cron ile sürekli olarak çalıştırılacak olan programlar yanında tek bir sefer çalıştırılacak olan programları da belirli bir tarihte çalışacak şekilde düzenleyebilirsiniz. Bu düzenlemede uygulama çalıştırıldıktan sonra dosya içinden ilgili satırın silinmesi gerekir. Bunun yerine aslında belli bir zamanda veya tarihte bir seferde çalıştırılmak üzere herhangi bir uygulamayı çalıştırmak için "at" kullanabiliriz.

at bir çok komuttan oluşur. Çalıştırılacak olan komutu oluşturan at, atq beklemekte olan diğer komutlar, atq ile sorgulanan ve beklemede olan komutları atrm ile silebilirsiniz. at ile kullanılabilecek diğer komutlar için kılavuz sayfasında ayrıntılı bilgi bulabilirsiniz.

at ile ileri bir tarihte çalıştırılacak olan komutu tanımlamak kabuk ortamında program yazmaktan farklı değildir. çalıştırılacak olan komutu veya komutlar dizisini tanımlamak için öncelikle tarihi tanımlamalı ve ardından çalıştırılacak olan komutları satır satır girmeniz gerekir. İşlemi sonlandırmak için yeni bir satıra geçiş yapıp CTL+D basarak girişi sonlandırarak komut satırına dönebilirsiniz. Aşağıda saat 16:00'da çalıştırılmak üzere ayarlanan bir örnek görüyoruz.

```
droideka# at -f komut_seti at teatime tomorrow
```

at ile çalıştırılacak olan komutları bir dosyadan okutabilirsiniz. Bu, komutları teker teker yazmak durumunda kalmadan işlemin yapılmasını sağlar. Öte yandan birden çok komut çalıştırmayacak iseniz bu durumda dosya kullanmak yerine doğrudan tanımlayabilirsiniz. (^D, CTRL+D anlamına gelmektedir.)

```
droideka# at 15:35
echo "merhaba" > /dev/ttyp2
^D
droideka#
```

at için SS:DD veya SSDD formatını kullanarak saat ve dakika olarak zamanı tanımlayabilirsiniz. Tanımladığınız saat geldiğinde komut çalıştırılacaktır. Tarih tanımlamak için ss:dd örneğindeki SS:DD AY GÜN YIL olarak kesin bir tanımlamaya gidebilirsiniz. Bunların dışında ise bazı bildik zamanlamalar hazır olarak tanımlıdır. Örneğin yukarıdaki örnekte görülen "at teatime tomorrow" yarın saat 16:00'oa çalıştırılacağını bildirmektedir.

Önceden tanımlanan olan komutları atq ile öğrenebiliriz:

```
droideka#
droideka# atq
Date                          Owner         Queue   Job#
19 Ağu 2009 Çar EEST 00:11:00   root          c       1
droideka#
```

Zamanlanmış olan 1 numaralı işlemi sonlandırmak için

```
droideka# atrm 1
droideka#
```

at komutu ile çalıştıracağınız komutlar atrun tarafından /etc/crontab dosyasında belirtilen aralıklar ile kontrol edilir. Bu kontrol her beş dakikada bir yapılır. Zamanı geçen komutlar işlenir. Bu kontrol aralığını kendi tercihlerinize göre düzenleyebilirsiniz ancak pek gerek duyulmayacaktır.

cron ve at İçin Erişim Kontolü

cron ve at ile birçok komutu veya programı belirli bir tarihte çalıştırmak üzere zamanlamasını yapabiliriz. Bu sistemdeki kullanıcılara kolaylık

sağlarken öte yandan sistem üzerindeki yükü arttırabilir. Bu durumda kullanıcıların hepsine cron ve at erişimi sağlamak sorun yaratabilir. Bu sorunların önüne geçebilmek için kullanıcılar arasında ayrım yapılarak erişim yetkilerinin kontrol altına alınması gerekebilir. Bunun için izin verilen-verilmeyen kullanıcıları tanımlamanız yeterlidir. Kullanıcı yetkilerini tanımlamak için ilgili dosyaların oluşturulması ve düzenlenmesi gereklidir. Standart BSD sistem kurulumlarında bu dosyalar bulunmaz. Standart olarak kullanıcılar "cron" kullanabilir ama "at" kullanamazlar.

Sistem yöneticisi bu yapılandırma dosyalarını kendisi oluşturup gerektiği gibi düzenlemelidir. cron ve at'nin potansiyel olumsuz kullanımlarına karşı izin verilecek olan kullanıcı adları belirlendikten sonra cron için /var/cron/allow ve /var/cron/deny dosyaları oluşturulup bu dosyalara kullanıcı isimleri eklenmelidir. Dosyalar oluşturulduktan sonra, sadece root tarafından erişilebilir kılınmalıdır. (chmod 600). Eğer /var/cron/deny dosyasını oluşturduysanız bu durumda /var/cron/allow dosyasına gerek yoktur. İzin verilmeyen kullanıcılar dışında kalan kullanıcılar cron kullanabilir. Ancak her iki dosya varsa bu durumda allow dosyasında yer alan kullanıcı adı, deny dosyasında bulunsa dahi izin verilir. Diğer bir deyişle allow dosyası deny dosyasına göre üstündür.

Benzer yapılandırma at için de yapılabilir. /var/at/at.allow dosyasına at kullanabilecek olan kullanıcılar, /var/at/at.deny dosyasına da izin verilmeyen kullanıcılar eklenmeli ve cron'da olduğu gibi sadece root tarafından erişilebilir kılınmaları gereklidir. (chmod 600) Eğer bu dosyalar bulunmuyorsa standart yapılandırma gereğidir.

Periodic Görev Zamanlayıcısı

BSD sistemlerde /etc/crontab dosyası içinde gördüğüniz üç satır farklı dizinlerdeki kabuk programlarını çalıştırmaktadır. crontab dosyasında tenımlanan zamanlarda sistem bakımının yapılmasını sağlar.

```
# Perform daily/weekly/monthly maintenance.
1       3       *       *       *       root    periodic daily
15      4       *       *       6       root    periodic weekly
30      5       1       *       *       root    periodic monthly
```

Periodic komutu günlük-daily, weekly-haftalık ve aylık-monthly olmak üzere üç farklı zamanda çalıştırılmaktadır. Bu üç farklı satır /etc/periodic altında bulununa üç ayrı dizine işaret etmektedir. Bu dizinlerde kabuk programları yer alır. Bu kabuk programları belirtilen zamanlarda bakım çalışmalarını gerçekleştirir. /etc/periodic altında şu dizinleri görebilirsiniz:

```
droideka# ls
daily           monthly         security        weekly
droideka# cd daily
droideka# pwd
/etc/periodic/daily
droideka# ls
100.clean-disks         140.clean-rwho          300.calendar            404.status-zfs
408.status-gstripe      440.status-mailq        480.status-ntpd
110.clean-tmps          150.clean-hoststat      310.accounting          405.status-ata-
raid    409.status-gconcat      450.status-security     500.queuerun
120.clean-preserve      200.backup-passwd       330.news                406.status-
gmirror     420.status-network      460.status-mail-rejects 999.local
130.clean-msgs          210.backup-aliases      400.status-disks        407.status-graid3
430.status-rwho         470.status-named
droideka# cd ../monthly/;ls
200.accounting  999.local
droideka# cd ../security/;ls
100.chksetuid           300.chkuid0             410.logincheck          510.ipfdenied
550.ipfwlimit           800.loginfail           security.functions
200.chkmounts           400.passwdless          500.ipfwdenied          520.pfdenied
700.kernelmsg           900.tcpwrap
droideka# cd ../weekly/;ls
310.locate      320.whatis      330.catman      340.noid        400.status-pkg  999.local
droideka#
```

Bu dizinlerin içeriğinde yer alan kabuk programları zamanı geldiğinde

çalıştırılacak ve gereken bakım çalışmaları gerçekleştirilecektir. Bu kabuk programları isimlerinin önüne konulan sayılara sahiptir. Bu düzenleme söz konusu kabuk programlarının belli bir düzende çalıştırılmasını sağlamak içindir. Bu sıralama matemetikte leksicografiksel düzen veya alfebetik sıralama olarakda adlandırılır. Bu nedenle /etc/peridoic/daily dizininde yer alan 999.local en son çalıştırılacak olan kabuk programı olacaktır.

Bu kabuk programlarının çalıştırılmasının ardından, çıktıları standart olarak root kullanıcısına /var/mail/root dosyasına kayıt edilir. Bu dosya BSD sisteminizde gece yarısından sonra sabah 03:00 sıralarında raporlanarak /var/mail/root dosyasına kayıt edilir. Aşağıda örnek bir çıktı görülmektedir.

```
From root@droideka.elkotek Tue Aug 18 03:04:17 2009
Return-Path: <root@droideka.elkotek>
Received: from droideka.elkotek (localhost [127.0.0.1])
        by droideka.elkotek (8.14.3/8.14.3) with ESMTP id n7I04HPc006319
        for <root@droideka.elkotek>; Tue, 18 Aug 2009 03:04:17 +0300 (EEST)
        (envelope-from root@droideka.elkotek)
Received: (from root@localhost)
        by droideka.elkotek (8.14.3/8.14.3/Submit) id n7I04Hiq006313
        for root; Tue, 18 Aug 2009 03:04:17 +0300 (EEST)
        (envelope-from root)
Date: Tue, 18 Aug 2009 03:04:17 +0300 (EEST)
From: Charlie Root <root@droideka.elkotek>
Message-Id: <200908180004.n7I04Hiq006313@droideka.elkotek>
To: root@droideka.elkotek
Subject: droideka.elkotek daily run output



Removing stale files from /var/preserve:

Cleaning out old system announcements:

Removing stale files from /var/rwho:

Backup passwd and group files:
droideka.elkotek passwd diffs:
29d28
< cups:(password):193:193::0:0:CUPS Owner:/nonexistent:/usr/sbin/nologin
31a31
> cups:(password):193:193::0:0:CUPS Owner:/nonexistent:/usr/sbin/nologin
droideka.elkotek group diffs:
37d36
< cups:*:193:
40a40
> cups:*:193:

Verifying group file syntax:
/etc/group is fine

Backing up mail aliases:

Disk status:
Filesystem     Size    Used   Avail Capacity  Mounted on
/dev/ad4s2a    496M    225M    231M    49%    /
devfs          1.0K    1.0K      0B   100%    /dev
/dev/ad4s2e    496M     40M    416M     9%    /tmp
/dev/ad4s2f    240G     42G    179G    19%    /usr
/dev/ad4s2d    4.8G    271M    4.2G     6%    /var
procfs         4.0K    4.0K      0B   100%    /proc

Last dump(s) done (Dump '>' file systems):

Network interface status:
Name    Mtu Network       Address              Ipkts Ierrs    Opkts Oerrs  Coll
nfe0   1500 <Link#1>      00:23:7d:c6:22:71   560560     0   454943     0     0
nfe0   1500 192.168.7.0   droideka            556371     -   454741     -     -
lo0   16384 <Link#2>                           0     0       0     0     0
lo0   16384 fe80:2::1     fe80:2::1             0     -       0     -     -
lo0   16384 localhost     ::1                   0     -       0     -     -
lo0   16384 your-net      localhost             0     -       0     -     -

Local system status:
 3:01AM  up 10:47, 0 users, load averages: 0.03, 0.09, 0.07

Mail in local queue:
/var/spool/mqueue is empty
                Total requests: 0
```

```
Mail in submit queue:
/var/spool/clientmqueue is empty
        Total requests: 0

Security check:
   (output mailed separately)

Checking for rejected mail hosts:

Checking for denied zone transfers (AXFR and IXFR):

-- End of daily output --
```

/etc/periodic dizininde yer alan kabuk programlarının çalışması /etc/defaults/periodic.conf dosyasında tanımlıdır. Bu dosyayı inceleyerek söz konusu kabuk programlarının nasıl işlediğini öğrenebilirsiniz. Kendi yapılandırmanızı tanımlamak isterseniz /etc/perdiodic .conf dosyasını kullanabilirsiniz. Bu dosya da /etc/rc.conf dosyasında olduğu gibi işlemektedir. Böylece standart yapılandırma yerine kendi tercih ettiğiniz yapılandırmayı kullanabilirsiniz; farklı bir dosyaya kayıt alma veya farklı zamanlarda çalışma gibi. Burada dikkat edilecek olan nokta /etc/periodic/security dizininde yer alan güvenlik bakımı yapan kabuk programları için hazırlanmış olan programlardır. Bunların işleyişi kritik bilginin istenmeyen birilerinin eline geçebileceği düşünülerek diğerlerinden farklıdır. Bu dosyalar da yine /etc /defaults/periodic.conf dosyasında tanımlandığı gibi işletilir ama sistem yöneticisinin tercihi gereği /etc/periodic.conf dosyası düzenlenerek farklı biçimde işletilebilir.

Sistem yöneticileri, kendi hazırladıkları kabuk programlarını ilgili periodic dizinleri altına kopyalayabilirler. Ancak /etc/periodic yerine /usr/local/etc/periodic dizinine kopyalanmaları ve oradan çalıştırılmaları uygundur. /etc/crontab dosyasında tanımlı olan işlemler cron tarafından çalıştırıldıktan sonra /usr/local/etc/ dizinine dönerek buradakileri çalıştıracaktır. Bu çalıştırma işlemi yine /usr/local/etc /periodic/daily,monthly,weekly,security dizinlerinde yukarıda anlatıldığı gibi alfebetik sırada gerçekleşecektir. Sistem yöneticisinin gerçekleştireceği ve sık sık tekrarlanan işlemlerin örneğin ports güncellemesi vb. işlemler /usr/local/etc/ altına alınarak cron ile gerektiği gibi yapılabilir. Bunları tek bir crontab dosyasına toplamaktansa sistem ile sonradan kurulan uygulamalar ve diğer servislerin asıl sistemden ayrı tutulması en uygun çözüm olacaktır.

BSD - X

Port Sistemi

Gökşin Akdeniz
goksin@enixma.org

UNIX sistemlerde temel sistemin parçası olmayan yazılımlar sistem kurulum kaynaklarında yer almaz. Bu ek yazılımların sistem yöneticisi tarafından kurulması gerekir. Yazılım kurulumu UNIX sistemler için geçmişte ve bugün de kaynak koddan derlenerek yapılmaktadır. Kaynak kod dediğimiz aslında bilgisayara programının yapacağı işlemlerin biz ölümlüler tarafından okunabilir ve anlaşılabilir biçimde yazıldığı bir dizi komutun yazıldığı bir metin dosyasıdır. Bu dosyadaki komutlar derleyici tarafından işleme tabi tutularak bilgisayarın anlayıp işleyebileceği 1 ve 0 dizilerine dönüştürülür.

UNIX sistemlerde yazılım kurmak kaynak koddan derlenerek yapılagelmiştir. Bunun istisnası ise RPM yazılım yönetim aracının UNIX sistemlere aktarılması ve kullanılmasıdır. UNIX sistem yöneticilerinin genel olarak kaynak koddan derleyerek yazılım kurmak durumunda olmaları bir UNIX sistem yöneticisi için zor bir durum değildir. Sorurn olursa, derleme sırasında derleyicinin döndürdüğü hata mesajlarına bakarak gereken çözüm uygulanarak sorunu çözebilir. Ancak bu durum UNIX sistem yöneticisinin bundan otuz yıl önce doktora tezini tamamlamakta olan bilgisayar mühendisleri veya yirmi yıl önce yüksek lisans derecesine sahip olan bir bilgisayar mühendisi olduğunu göz önüne alırsak hata mesajını okuyup, yorumlayıp çözüm sunan sistem yöneticisine hayran olmamak elde değildir. Hele hele UNIX'in kapalı kaynak kodlu ticari sürümlerinin ortaya çıkması ile her sistem yöneticisinin çalıştığı sistemi en iyi şekilde bilmek durumunda olması gerektiğini düşünecek olursak. Kaynak koddan yararlanarak yazılım derlenirken, eğer derleme süreci başarıyla sonuçlanırsa derlenen yazılım çalışacaktır. İşlemin başarı ile sonuçlanması için derlenecek kaynak kodun derleme işlemi yapılacak sistem için yazılmış olması gerekir. Aksi durumda derleme işlemi başarısızlıkla sonuçlanacaktır. Bu durumda döndürülen hata mesajlarına bakarak soruna çözüm bulunması gerekir. Platform farklılıklarının neden olduğu sorunların aşılması için birçok çözüm geliştirilmiştir. Bunlardan autoconf farklı platformlarda kaynak kod derlenirken olabilecek sorunları ortadan kaldırmaya yönelik önemli bir adım olsa da tüm problemleri çözmekten uzak kalmaktadır. Platform farklılıklarını gidermenin en basit yolu her bir platform için kaynak kodu yeniden yazmak olarak düşünebilir ancak bu çözümün gerektirdiği kaynak, zaman, emek vb pratik olarak bu durumu olanaksız kılmaktadır. Öte yandan tüm yazılımlar autoconf gibi araçları kullanarak derlenememektedir. Sistem yöneticisinin bir UNIX sistemde yetkili olan tek kişi olarak döndürülen hata mesajına bakarak kaynak kodu yeniden ele alıp yamaması veya makefile dosyasını yeniden düzenlemesi gerekebilir.

BSD sistemlerde kaynak kod kullanılarak yazılımın derlenerek kurulması ve kullanılması yukarıdaki sürece göre çok daha kolaydır. Bunun için FreeBSD -ports collection/ports/ports

tree- port kolleksiyonu/port ağacı/port olarak adlandırılan sistemi kullanırken, NetBSD ve DragonflyBSD ise pkgsrc kullanırlar. Bu bölümde önce FreeBSD'de kullanılan 'port'u ele acağız. Gelecek ay ise pkgsrc üzerinde duracağız.

FreeBSD Ports

Linux dağıtımları derleme sürecinde yaşanan sorunları aşmak için yazılımın önceden derlenerek kullanıma hazır hale getirilmiş olarak sunarlar. Bu sunulan hazır derlenmiş yazılımlar "paket" adı altına kullanıcılara sunulur. Paketler ya merkezi bir depoda ya da CD/DVD setlerinde sunulur. Yazılımın doğru biçimde çalışması için gereken kütüphaneler ile diğer derlenmiş uygulamalar paketlerin içinde yer alır ve kurulum işlemi sırasında hangi dizin altına kopyalanacağı da bu paketin içinde yer alır. "Paket" yaklaşımı Linux dağıtımlarının farklı çözüm yaklaşımları nedeni tek tip olmaktan uzaktır. En yaygın olarak kullanılan paket sistemleri olan RPM ve DEB bugün farklı paltformlarda kullanılmaktadır. RPM, sadece Linux dağıtımlarında değil aynı zamanda ticari UNIX sistemlerde de kullanılmıştır. DEB, bildiğiniz Debian paketlerinin adıdır ve Debian, ardından türevi olan Ubuntu vs dağıtımlarda da kullanılmıştır. Ama Debian paketlerini *buntu'larda kurmanız söz konusu değildir. Tersi olarak her ne kadar paketlerin uzantısı DEB olsa da *buntu paketleri de Debian'da kurulmaz. RPM ve DEB vs make kullanarak yazılım derlemek ve ardından yine make kullanarak derlenmiş yazılımı kaldırmak gibi zahmetli ve sorunlu bir sürecin sakıncalarını ortadan kaldırmıştır. Paket sistemini kullanmak yerine derlemeyi de tercih edebilirsiniz. Ancak bir sistem yöneticisi açısından konuya bakarsak ve elinize geçen bir tar.gz veya tar.bz2 dosyası ile dağıtılan bir kaynak kodu alıp sistem yöneticinize gidip " bunu derleyelim, ne duruyoruz?" dediğinizde sizi nazikçe red etmeyecek bir sistem yöneticisine rastlayacağınızı sanmıyorum. O halde neden halen make var sorusunu soracaksınız. RPM, DEB vs. hangi paket yöneticisini kullanırsanız kullanın kurduğunuz paketler bir tar.gz/tar.bz2 dosyasından farklı değildir. Sadece adı farklıdır. Bu dosyaların içinde make kullanılarak derlenmiş hazır dosyalar ile gerekli yapılandırma dosyaları ile diğer kabuk programı vs barındırırlar. Peki neden derlemeyelim sorusu her zaman bu açıklamanın ardından gelen ikinci sorudur. Derleme işlemi kaynak kodun bir derleyici ile işlenip bilgisayarın işleyebileceği hale dönüştürülmesidir. Bu işlem make ile yapılır. (bkz: man make) make derleme işlemi için makefile gereklidir. Derlemeden önce yapılandırma/configure işlemi yapılır, gereken kütüphaneler vs. aranır, bulunur, gerekiyorsa yamalar yapılır. Ardından make derleme işlemine başlar ve derlemenin bitmesinin ardından ise dosyaların uygun yere koyalanması gereklidir. Daha sonrasında yazılımın güncellenmesi ve kaldırılması gibi işlemler için ise -tabi öncelikle eski yazılımınn sistemden sorunsuzca kaldırılması gerekir- bu sürecin yeniden gerçekleştirilmesi zorunludur.

BSD sistemler ise yazılım yönetimi sürecini biraz daha ileri taşımıştır. Yazılımı derleyebilirsiniz, paketleyebilirsiniz, kaldırabilirsiniz, terfi veya bir eski sürüme indirgeyebilirsiniz. Bunu yapmak için tek bir araç yerine her bir işlem için tasarlanmış olan aracı kullanırsınız, yani pkg_* ve ports. pkg_* araçları hazır derlenmiş olan paketler ile çalışmanızı sağlarken, ports kaynak kodu tek bir komut ile derleyerek kurmanıza, terfi etmenize ve kaldırmanıza olanak sağlar. Linux dağıtımları da yapıyor diyecekler için anımsatalım: İlk paket yönetim aracı 1995'te Debian ile gelen dpkg ile hemen hemen aynı zamanda BSD sistemlerde pkg_* ile birlikte PORTS kullanılmaya başlanmıştır.

Bu arada, ports konusunu ele aldığımızda RPM yeniden karşımıza gelecek. Neden diyen olursa, BSD sistemler Linux emülasyonu sunar. %100 LSB-Linux Standart Base uyumlu olan bu emülasyon bu nedenle Fedora ve doğal olarak RPM desteği sunar. Fedora için sunulan RPM paketlerini kullanabilirsiniz. Diğer bir dağıtım kullanabilir miyiz diye soranlar olacaktır, bu dağıtımlar PORTS üzerinde yer alır, desteği azdır veya kaldırılmıştır. Kullanıp kullanmamak size kalmış. Bu konuya ileride yeniden döneceğiz.

BSD sistemlerde kullanılan derlenmiş ve kullanıma hazır olan yazılımları barındıran paket sisteminin altyapısını geliştiren kişi Jordan Hubbard'dır.[1] Paket sisteminin altyapısının hazırlanmasında ve geliştirilmesinde büyük emeği olmuştur. FreeBSD bu paket sistemini kullanan ilk BSD sistem olmuş daha sonra netBSD bu sistemi kullanmaya başlamıştır. İzleyen süreçte birçok açık kaynak kodlu yazılım projesi Jordan Hubbard'ın geliştirdiği araçları ve altyapıyı kendi bünyesinde kullanmaya başlamıştır.

Paketi tanımlamak gerekirse, yapılandırma dosyaları, paylaşımlı kütüphaneler, belgelendirme, çalıştırılabilen dosyalar vb. bir arada bulunduğu, hazırlandığı bilgisayardan başka bir bilgisayarda kullanıldığında, barındırdığı programların, kütüphaneleri, belgelerin vs. sorunsuzca kullanılmasına, çalıştırılmasına olanak veren sistemdir. Paket sistemi şu işlemleri de yapabilir:

-Sistemde kurulu olan paketlerin/portların bir listesini veritabanında saklar.

-Uzak bir sunucudaki paketleri kurabilir.

-Kurmakta olduğunuz paketlerin gereksinim duyduğu diğer paketleri -bağımlılıkları- sizin için kurar.

-Var olan paketleri terfi eder, günceller ve geride artık bırakmadan kaldırır.

Bu işlemler Linux dağıtımlarının paket araçları tarafından da yapılır. BSD sistemlerdeki farkı ise paketler ve port aynı sistematik ile çalışır ve temel sisteme ait olan tüm uygulamalar, bunlara sizin kendi kurduklarınız da dahil, /usr/local altında bulunur. Diğer bir deyişle, /usr/local aslında /usr ile aynı dizin hiyerarşisine sahiptir. Bunun istisnası /usr/local/man dizininin kılavuz sayfalarına ayrılmış olmasıdır. /usr/local/share/man dizini bu nedenle yoktur. Böylelikle temel sistemin parçası olmayan herşey /usr/local altında bulunur. Böylelikle kullanıcı tarafından kurulan ve sisteme ait olamayan yazılımlar kesin olarak ayrılır. Linux dağıtımları ise yaygın olarak hepsini aynı dizine yerleştirir ve /usr/local genellikle "boştur", hemen hemen herşey aynı dizinin altına gider.

Aşağıda /usr/local dizin hiyerarişini görüyoruz:

```
[goksin@droideka /usr/home/goksin]$ cd /usr/local
[goksin@droideka /usr/local]$ ls -l
total 160
drwxr-xr-x    3 root  wheel  34816 29 Ağu 00:08 bin
drwxr-xr-x    9 root  wheel    512 15 Ağu 00:50 diablo-
jdk1.6.0
drwxr-xr-x    2 root  wheel    512  8 Ağu 14:07 env
drwxr-xr-x   29 root  wheel   1536 26 Ağu 23:00 etc
drwxr-xr-x  202 root  wheel  31232 26 Ağu 23:00 include
drwxr-xr-x    3 root  wheel   2048 26 Ağu 11:05 info
drwxr-xr-x   52 root  wheel  69632 29 Ağu 00:08 lib
drwxr-xr-x    8 root  wheel    512 13 Ağu 18:02 libdata
drwxr-xr-x    8 root  wheel   1536 15 Ağu 00:19 libexec
drwxr-xr-x   33 root  wheel   1024 15 Ağu 04:15 man
drwxr-xr-x    8 root  wheel    512 15 Ağu 14:10
openoffice.org-2.4.2
drwxr-xr-x    2 root  wheel   1536 17 Ağu 16:59 sbin
drwxr-xr-x  124 root  wheel   2560 26 Ağu 23:00 share
drwxr-xr-x    2 root  wheel    512  7 Ağu 17:45 translations
drwxr-xr-x    2 root  wheel    512  6 Ağu 15:58 www
[goksin@droideka /usr/local]$
```

Gördüğünüz dizinler sizin kullandığınız sistemde farklı olabilir. Ancak değişmeyen ve standart olarak bulunan dizinleri açıklamaları ile birlikte aşağıda görebilirsiniz.

bin	Derlenmiş olan uygulamaları
etc	Yapılandırma dosyaları
include	Yeni yazılım kurmak için gerekecek olan C include

dosyaları	
info	Belgelendirme hazırlamak için kullanılan dosyalar
lib	Paylaşımlı kütüphaneler
libexec	Diğer uygulamalar tarafından kullanılan programlar
man	Kılavuz sayfaları
sbin	Sistem tarafından kullanılan programlar
share	Platformdan bağımsız dosyalar.

Bağımlılıklar ve Paylaşımlı Kütüphaneler

Paylaşımlı kütüphaneler veya paylaşılan kütüphaneler, diğer uygulamalar tarafından kullanılan bazı işlevleri barındıran derlenmiş dosyalardır. Paylaşımlı kütüphaneleri kullanarak bir program çeşitli işlevlere/fonksiyonlara erişip bunları kullanabilir. Ayrıca dinamik bağlama -dynamic linking- olarak da adlandırılır. Bu tekniğin yararı statik bağlamaya -static linking- göre yazılacak olan kodu azalatarak programların boyutlarını küçültmesidir. Statik olarak bağlama tekniği ise tersine olarak bir programın derlenmesi sırasında diğer kodu dahil eder ve dosya boyutunu büyütür. Bu teknik -statik olarak- derleyeceğiniz her programın kendi bünyesine paylaşılan kodun eklenmesini sağlar, derlenmiş programın boyutu büyür. Paylaşımlı kütüphaneler, tüm platformlarda bulunur ama farklı isimlere sahiptir. örneğin Windows için bu Dynamic Link Libraries (DLL) olarak tanımlanır.

BSD sistemlerde temel sistem gerekli olan tüm paylaşımlı kütüphaneleri barındırır. Böylece hem asıl sistem hem de sonradan kurulacak olan uygulamaların gereksinim duyacağı tüm -hemen hemen hepsi- kütüphaneler sistemde bulunur. Ender de olsa bazı kurmak isteyeceğiniz programlar hazır bulunmayan kütüphanelere gereksinim duyabilir ve bunların sonradan kurulması gerekir.

BSD sistemlerde paylaşılan/paylaşımlı kütüphaneler /usr/lib dizini altında bulunur. Sonradan kurulan paylaşımlı kütüphaneler ise /usr/local/lib altında barındırılır. BSD sisteme kurduğunuz tüm yazılımlar bu dizinleri kullanarak kütüphanelere erişir. İlk olarak /usr/lib altında arama yapılır bulunmaz ise /usr/local/lib ve bir kaç başka dizin altında aranır. Kütüphanelerin bulunduğu dizinleri /etc/rc.conf yapılandırma dosyasını düzenleyerek değiştirebilirsiniz.

BSD sisteme kurduğunuz her paket doğru biçimde çalışmak için diğer paketlere gereksinim duyar. Buna bağımlılık denir. Kurmak istediğiniz paket için gereken diğer paketler vs. sistemde bulunmuyorsa, paket kurulumu sırasında diğer gerekli olan paketler kurulur, ardından asıl kurmak istediğini paket kurulur. Daha sonra kaldırmak istediğiniz bir paket diğer paketlerce gereksinim duyuluyorsa bu durumda paket kaldırma işlemi gerçekleşmez.

FreeBSD'de Kurulmuş Paketler Hakkında Bilgi Edinmek

Paket sistemini kavramının en kolay yolu sistemde kurulu olan paketler hakkında bilgi edinmektir. Sisteminizde yaptığını kuruluma göre farklı paketler bulunur. Bu paketler arasında paket yönetim araçları da yer alır. Paketleri kurmak, kaldırmak, güncellemek gibi işlemler için kullanacağınız pkg_* araçları; pkg_add, pkg_delete, pkg_info, pkg_update, pkg_version ve pkg_create uygulamalarıdır. Diğer paket yönetim araçlarının aksine her iş için ayrı bir araç yazılmıştır. Böylece tek bir araçta birden çok seçeneği kullanmak ve anımsamak durumunda kalmazsınız. Bu araçların hepsi paketlerin kayıtlarını tutan /var/db/pkg sizininde yer alan paket veritabanını ile bütünleşik olarak çalışır. Burada veritabanı dediğimizde ilk akla gelen bir tane büyük bir dosya olduğudur. Aksine, sistemde kurulu olan tüm paketlerin isimlerinin bulunduğu dizinleri görürsünüz. Bu dizinler ilgili olduğu paketin adını taşır ve içinde adı geçen pakete ilişkin bilgiler yer alır. pkg_info bu bilgiyi kullanarak pakete ilişkin bilgiyi döndürür. pkg_info'yu tek başına kullanarak sistemde kurulu olan tüm paketler hakkında bilgi edinebilirsiniz. Aşağıdaki örnekte sistemimde kurulu olan paketlerden bir bölümünü görüntülüyoruz:

```
droideka# pkg_info | head -15
ORBit2-2.14.17      High-performance CORBA ORB with support
for the C language
OpenEXR-1.6.1_1     A high dynamic-range (HDR) image file
format
aalib-1.4.r5_4      An ascii art library
akode-2.0.2,1       Default KDE audio backend
akode-plugins-mpc-2.0.2,1 Musepack decoder plugin for akode
akode-plugins-mpeg-2.0.2,1 MPEG audio decoder plugin for akode
akode-plugins-oss-2.0.2,1 OSS output plugin for akode
akode-plugins-resampler-2.0.2,1 Resampler plugin for akode
akode-plugins-xiph-2.0.2_2,1 FLAC/Speex/Vorbis decoder plugin
for akode
amarok-1.4.10_5     Media player for KDE
amspsfnt-1.0_5      AMSFonts PostScript Fonts (Adobe Type 1
format)
apache-ant-1.7.1    Java- and XML-based build tool,
conceptually similar to mak
appres-1.0.1        Program to list application's resources
apr-ipv6-gdbm-db42-1.3.8.1.3.9 Apache Portability Library
arts-1.5.10_2,1     Audio system for the KDE integrated X11
desktop
droideka#
```

pkg_info'da yukarıdaki örnekte gördüğünüz gibi paketlere ilişkin bilgi aynı zamanda mimari ve diğer bilgileri barındırmamaktadır. Linux sistemler ile çalıştıysanız bu durum size garip gelebilir ama paket sistemi ilk olarak x86 üzerinde geliştirildiği için bu bilgileri barındırmaz. Hatta BSD sistemlerin ftp sitelerinde göreceğiniz üzere paketler paket_adı.tbz dosyası olarak ilgili mimari için ayrılmış dizin altında bulunmaktadır. Eski sürümlere ait olan paketlere rastlarsanız bunların tgz yani tar.gz dosyaları olduğunu görebilirsiniz. tbz dosyaları ise bildiğiniz tar.bz2 dosyalarıdır. gzip kullanmaya göre bzip2 kullanılması sadece bzip2'nin daha iyi sonuçlar vermesinden dolayıdır. Yukarıdaki örnekte gördüğümüz paketlere ait bilgiler bir satır uzunluğunda olan kısa tanımlamalardır. Biraz daha ayrıntılı bilgi edinebiliriz. Daha fazla ayrıntı ise tekil olarak paket bazında döndürülür. Örneğin zip-3.0 paketi hakkındaki bilgi aşağıda görülmektedir.

```
droideka# pkg_info zip-3.0
Information for zip-3.0:

Comment:
Create/update ZIP files compatible with pkzip


Required by:
firefox-3.5.2,1


Description:
Zip is a compression and file packaging utility.  It is
compatible with
PKZIP 2.04g (Phil Katz ZIP) for MSDOS systems.  There is a
companion to zip
called unzip (of course) which you can also install from the
ports/package
system.

WWW: http://www.info-zip.org/Zip.html



droideka#
```

Daha fazla ayrıntı ise -v seçeneği yani verbose seçeneği ile edinilebilir. Bu durumda paket hakkındaki bilgi dışında pakette yer alan tüm dosyalar vs. listelenir. Aşağıdaki örnekte çıktının tamamı aktarılmamıştır.

```
droideka# pkg_info -v zip-3.0
Information for zip-3.0:

Comment:
Create/update ZIP files compatible with pkzip
```

```
Description:
Zip is a compression and file packaging utility.  It is
compatible with
PKZIP 2.04g (Phil Katz ZIP) for MSDOS systems.  There is a
companion to zip
called unzip (of course) which you can also install from the
ports/package
system.

WWW: http://www.info-zip.org/Zip.html


Packing list:
        Comment: PKG_FORMAT_REVISION:1.1
        Package name: zip-3.0
        Package origin: archivers/zip
        CWD to /usr/local
File: bin/zip
        Comment: MD5:d31771c2411be351f1556f4097575d32
File: bin/zipcloak
        Comment: MD5:25fdfe79ad1de6626092be50a13e2c35
File: bin/zipnote
        Comment: MD5:1bdb97d6435465e983f817b50a464826
File: bin/zipsplit
        Comment: MD5:1def400d2cfd19834b7f27b694cebbda
File: man/man1/zip.1.gz
        Comment: MD5:ccd5f71cfb25c09e0d5a59e125f4cf5f
File: man/man1/zipcloak.1.gz
        Comment: MD5:a585a8020b0a0f030e59b0274299c497
File: man/man1/zipnote.1.gz
        Comment: MD5:f71924cc4f2cce23806f386984c2a3bc
File: man/man1/zipsplit.1.gz
        Comment: MD5:1720d6f12a3407bae4a319c16d322757
        UNEXEC 'rm -f %D/man/cat1/zip.1.gz %D/man/cat1/zip.1
%D/man/cat1/zip.1.gz %D/man/cat1/zip.1.gz.gz
%D/man/cat1/zip.1.gz.bz2'
        UNEXEC 'rm -f %D/man/cat1/zipcloak.1.gz
%D/man/cat1/zipcloak.1 %D/man/cat1/zipcloak.1.gz
%D/man/cat1/zipcloak.1.gz.gz %D/man/cat1/zipcloak.1.gz.bz2'
        UNEXEC 'rm -f %D/man/cat1/zipnote.1.gz
%D/man/cat1/zipnote.1 %D/man/cat1/zipnote.1.gz
%D/man/cat1/zipnote.1.gz.gz %D/man/cat1/zipnote.1.gz.bz2'
        UNEXEC 'rm -f %D/man/cat1/zipsplit.1.gz
%D/man/cat1/zipsplit.1 %D/man/cat1/zipsplit.1.gz
%D/man/cat1/zipsplit.1.gz.gz %D/man/cat1/zipsplit.1.gz.bz2'
        CWD to .
File: +COMMENT (ignored)
        Comment: MD5:482c7883e97f4796b330265d21664958
```

Yukarıdaki çıktıda gördüğünüz bölümde, paketin tanımı, paketin içinde yer alan dosyalar, varsa paket kurulumu sonrasında çalıştırılacak olan komutlar, kaldırma işleminde silinecek olan dosyalar vb. ile kaldırma işlemi sonrası yapılacak işlemler tanımlıdır. Bazı paketlerde örnek standart bir yapılandırma dosyası da pakette yer alır. Bu dosya kullanılarak kendi yapılandırmanız hazırlanır. Eğer varsa bu örnek dosya kullanılarak kendi yapılandırma dosyanızı oluşturabilirsiniz. Sonradan oluşturduğunuz yapılandırma dosyaları ile bu temel dosyalar karşılaştırılır ve ikisi arasında fark bulunmuyorsa paketi kaldırdığınızda kendi hazırladığınız yapılandırma dosyası da silinir, aksi halde kendi yapılandırma dosyanız saklanır.

Bir diğer nokta ise kullandığımız paketlerin gücel olup olmadıklarıdır. Bir paketin güncel olup olmadığını nasıl bilebiliriz? pkg_version aracı bu iş için geliştirilmiştir. pkg_version ancak ve ancak ports kurulu ise işe yarayacaktır. Sistemde kurulu olan paketlerin sürüm numaralarını ports üzerindeki değerleri ile karşılaştırarak bir paketin güncel olup olmadığına ilişkin bilgiyi döndürür.

```
droideka# pkg_version | head -15
ORBit2                                  =
OpenEXR                                 =
aalib                                   =
akode                                   =
akode-plugins-mpc                       =
akode-plugins-mpeg                      =
akode-plugins-oss                       =
akode-plugins-resampler                 =
akode-plugins-xiph                      =
```

```
amarok                                =
amspsfnt                              =
apache-ant                            =
appres                                =
apr-ipv6-gdbm-db42                    =
arts                                  =
droideka#
```

İki sütun olarak gördüğünüz çıktıda sol tarafta paket isimleri sağda ise port sürümü ile karşılaştırması yer almaktadır. Güncel paketler "=", eski olan paketler ise "<" ile işaretlenir. Çıktıyı daha açıklayıcı kılmak için "--verbose" yani "-v" seçeneğini kullanabiliriz. Böylece paketlerin durumunu daha kolay görebiliriz. pkg_version sistemde kurulu olan tüm paketlerin durumunu listeler.

```
droideka# pkg_version -v | head -15
ORBit2-2.14.17                        =   up-to-date with port
OpenEXR-1.6.1_1                       =   up-to-date with port
aalib-1.4.r5_4                        =   up-to-date with port
akode-2.0.2,1                         =   up-to-date with port
akode-plugins-mpc-2.0.2,1             =   up-to-date with port
akode-plugins-mpeg-2.0.2,1            =   up-to-date with port
akode-plugins-oss-2.0.2,1             =   up-to-date with port
akode-plugins-resampler-2.0.2,1       =   up-to-date with port
akode-plugins-xiph-2.0.2_2,1          =   up-to-date with port
amarok-1.4.10_5                       =   up-to-date with port
amspsfnt-1.0_5                        =   up-to-date with port
apache-ant-1.7.1                      =   up-to-date with port
appres-1.0.1                          =   up-to-date with port
apr-ipv6-gdbm-db42-1.3.8.1.3.9        =   up-to-date with port
arts-1.5.10_2,1                       =   up-to-date with port
droideka#
```

Paket sayısının yüzlerce olması durumunda grep kullanarak çıktıyı sınırlayabilirsiniz.

```
droideka# pkg_version | grep "<"
droideka#
```

Paket Kurmak: pkg_add

Temel sistemi kurduğunuzda kurulum kaynağından ilgili dağıtım seçimine göre paketler kurulur. Bunlara ek olarak gereksinim duyduğunuz yazılımlar paket olarak kurulum kaynağınızda, CD/DVD setleri ya da yerel/uzak sunucular, yer alır. Paketler içinde açık kaynak kod camiasınca geliştirilen hemen hemen tüm yazılımlar yer alır. Her bir yazılım kendisi için ayrılmış bir porta sahiptir ve port ağacı üzerinde yer alır. Bunlar içinde kapalı kaynak kodlu derlenmiş ve dağıtımı serbest olan Linux yazılımları da dahildir. Port ağacı üzerinde tüm yazılımlar bulunmadığı için "hemen hemen" dedik. Zira BSD port ağacı üzerinde yer alan yazılımların lisansları konusunda BSD geliştiricileri çok hasass davranır. Bazı yazılımların derlenmiş olarak dağıtılması olanaklı iken bazıları da sadece kaynak kod olarak dağıtılabilir. Bunun dışında platformunuza uygun bir sürümü olmayan yazılımlar paket olarak bulunmaz. Bu konuya port ağacı ve port konusunu ele alırken yeniden döneceğiz.

Sysinstall ile Paket Kurulumu

Sisteminizde port ağacını kurduysanız -Bazı BSD kaynaklarında bu PORTS COLLECTION olarak geçer. Bazı kaynaklarda ise PORTS TREE adı verilir. Benim tercihim ise PORTS TREE/PORT AĞACI terimi- kullanabileceğiniz paketler /usr/ports dizini altında yer alır.

```
droideka# cd /usr/ports
droideka# pwd
/usr/ports
droideka# ls
.cvsignore      Tools           distfiles       mail
security
.portsnap.INDEX UIDs            dns             math
shells
CHANGES         UPDATING        editors         mbone
sysutils
COPYRIGHT       accessibility   emulators       misc
```

```
textproc
GIDs            arabic          finance         multimedia
ukrainian
INDEX-5         archivers       french          net
vietnamese
INDEX-6         astro           ftp             net-im
www
INDEX-7         audio           games           net-mgmt
x11
INDEX-7.db      benchmarks      german          net-p2p
x11-clocks
KNOBS           biology         graphics        news
x11-drivers
LEGAL           cad             hebrew          palm
x11-fm
MOVED           chinese         hungarian       polish
x11-fonts
Makefile        comms           irc             ports-mgmt
x11-servers
Mk              converters      japanese        portuguese
x11-themes
README          databases       java            print
x11-toolkits
README.html     deskutils       korean          russian
x11-wm
Templates       devel           lang            science
droideka#
```

Göreceğiniz her dizin kendi içinde alt dizinler barındırır ve alt dizinlerin her birine port adı verilir. Port işaret ettiği paket komut satırından veya diğer araçlar kullanılarak kurulabilir. Ports ağacı üzerinde yazılım aramak sysinstall ile daha kolaydır. Sysinstall'ı çalıştırın. Configure ssçip enter basın. gelen ekranda Packages seçerek devam edin. Kurulum için kaynak seçimi ekranına geldiğinizde ilk seçenek DVD/CD setleridir. Bu seçeneği kulanarak elinizde bulunan DVD/CD setlerini kullanabilirsiniz. Eğer CD/DVD Setleri yok ise bu durumda FTP seçeneğini kullanmak gerekiyor. FTP seçeneğinde dünya üzerinde bulunan birçok BSD FTP sunucusu adresi yer alıyor. Coğrafi olarak size yakın olan bir sunucuyu tercih etmeniz uygun olacaktır. Seçtiğiniz sunucuya yapılan bağlantı sayısı fazla ise bu durumda kurulum süresi uzayabilir. başka bir sunucu seçerek yeniden denemeniz veya ports ağacını kullanmanız önerilir. Paketlerin listesine erişildiğinde aralarından seçim yapabilirsiniz. Paketleri seçerken ekranın alt kısmında pakete ilişkin tek satırlık bir açıklama görüntülenir. Bu açıklama satırı pkg_info ile görüntülenen pakete ait tek satırlık açıklamadır. Paketleri seçtikten sonra install ile kurulumu tamamlayabilirsiniz. Bazı durumlarda ise aradığını paket CD/DVD setlerinde veya ftp sunucularında bulunmaz. Bu durumda sysinstall'dan çıkmak için bir seçim yapmadan doğrudan install seçimi devam edin. karşınıza gelecek olan mesaj herhangi bir paketin seçilmediğini belirtecektir Bunuda evet diyerek sysinstall'dan çıkabilirsiniz. CD/DVD setleri boyut olarak tüm ports ağacındaki dosyaları barındıramaz. Bu yazının yazıldığı tarihte ports ağacı üzerinde yer alan port/paket sayısı 20613 olmuştu. Bunların boyutlarını düşünecek olursanız CD/DVD setlerinin sınırlı sayıda paketi barındırması da anlaşılabilir.

Sysintall ile kurulum yapılırken öncelikle seçilen paketler ve bağımlılıkları kontrol edilir. Seçilen paket ile varsa bağımlılıkları indirilir, ardından /usr/tmp dizinine açılır ve pkg_add ile kurulurlar. İşlemin tamamlanmasının ardından yeniden sysinstall arayüzüne dönersiniz. Kurulan paketlerde bulunan ikili dosyalar /usr/local/bin'de, kılavuz sayfaları /usr/local/man'da, yapılandırma dosyaları /usr/local/etc altında yerlerini almıştır. Eğer kurduğunuz bir paketi bulamıyorsanız çıkıp yeniden giriş yapın yeterli olur. Kurulumun ardından yapılandırma dosyalarını düzenlemeniz gerekebilir. Yapılandırma dosyaları /usr/local/etc dizininde parogram_adı.conf.sample benzeri bir ad ile yer alırlar. Bu dosya yapılandırmada esas alınacak olan dosyadır. Bu dosyayı program_adı.conf olarak kopyalamanız gerekir ve ardından sevdiğiniz metin düzenleyicisini kullanarak bu yapılandırma dosyalarını tercihlerinize göre düzenleyebilirsiniz. Bu işlemin ardından da rc.d dizininde yer alan başlangıç programlarına da göz atmanızda ve gerekiyorsa düzenlemenizde yarar var.

pkg_add ile Paket Kurulumu

Sysinstall ile paket kurulumu işleminde sunduğu seçenek sayısı azdır. Yapılandırma, ikili dosyalar ile kütüphaneler kopyalanacağı dizinler standart olarak tanımlıdır. Bu nedenle diğer işletim sistemlerindekilerin aksine kurulum süreci daha kontrollüdür. Bazı durumlarda ise kurulum sırasında dizin vb. seçenekleri kontrol etmek isteyebilirsiniz. Bunun için pkg_add kullanılması gerekir.

pkg_add aracı, tbz dosyaları ile kullanılmak üzere tasarlanmıştır. Dolayısıyla da bir paket ile kullanılmak durumundadır. pkg_add ile paket kurmak için iki yol söz konusudur: Birinci seçeneğimizde belirli bir dizine indirilmiş olan paketleri belirterek pkg_add ile bu paketler kurulur. İkinci seçenek ise paketlerin bulunduğu sunucu ve sunucudaki dizini tanımlayarak paketi kurmaktır. Bu yöntem ile pkg_add aynı sysinstall ile paket kuruluyormuşçasına gereken bağımlılıkları da dikkate alarak ve gerekiyorsa önce bağımlılıkları kurarak ardından kurmak istediğiniz paketi kuracaktır. Böylece kurduğunuz paket sorunsuzca çalışacaktır. Bu işlem sırasında uyarı mesajları döndürülebilir. Bunlar kurmaya çalıştığınız paketin bağımlı olduğu paketlerin eski sürümlerinin bulunduğu ancak programın bu paketlerin yeni sürümlerine gereksinim duyduğu mesajıdır. Bu mesajları aldığınızda sözkonusu paketleri de üst sürümlerine güncellemeniz gereklidir. Böylelikle kurulan paketin içinde yer alan uygulamaların da sorunsuzca çalışması sağlanacaktır. Bazı durumlarda paketlerin kurulumunun tamamlanmasının ardından yapılandırma ile ilgili bir mesaj görebilirsiniz. Bu mesaj, sözkonusu paketin kurulmasının ardından ilgili programın/programların doğru olarak çalışabilmesi için sistem yöneticisinin yapması gereken yapılandırma işlemlerini belirtmektedir. Böylelikle sistem yöneticisi yapılandırma dosyalarını gerektiği gibi düzenleyerek işlemi tamamlayacaktır.

pkg_add ile uzak sunuculardan kurulum yaparken varsayılan sunucu adresi kullandığınız BSD sistemin ftp sunucusudur. Bu sunucudan başka bir sunucu adresi kullanılmaz. Bunun istisnası ise paket adresinin tam yolunu yazarak paketin tanımlanmasıdır. "pkg-add sunucudaki_dizin/paket_adı.tbz" biçimindeki bir tanımlama ile pkg_add kullanabilir. Ancak pkg-add -r ile adres tanımlanamaz, pkg_add gereken sunucu adresi çözümlemesi işlemini kendisi yaparak paketi kurar.

pkg_add ile paket kurulumu yaparken bir noktaya dikkate etmek gerekiyor. BSD sistemlerde genel olarak PORT AĞACI üzerinde yer alan portlar yeni sürümün çıkışı sırasında paket olarak hazırlanır ve sunuculara konur. Bu paketler hazırlandıkları zamanki port ağacını esas alırlar. Bu nedenle eski olmaları ve yukarıda söz ettiğimiz lisans, mimari vb kısıtlamalar gereği tüm portlar paket olarak bulunmaz. Ancak paketlerin güncel port ağacı kullanılarak bir paket hazırlama sisteminde hazırlanması durumunda yerel bir ağ üzerindeki diğer BSD sitsemlere de bu paketlerin kurulması için pkg_add kullanılabilir.

Paketleri Güncellemek ve Kaldırmak

Sysinstall sadece paket kurar ama paketi kaldırmak için bir seçenek sunmaz. Bir paketi kaldırmak için doğrudan paket yönetim araçlarını kullanarak -pkg_delete <paket_adı> ile kaldırılabilir. Paketin kaldırılması için paketin adının doğru tanımlanması gereklidir. Eğer kaldırmak istediğiniz paketin adını anımsayamıyorsanız bu durumda pkg_info kullanılabilir. Aşağıdaki örnekte iki farklı Openoffice.org sürümünden 2.4.x kaldırmak için kullanılacak olan paket adının bulunmasını görüyoruz:

```
droideka# pkg_info | grep "openoffice.org"
openoffice.org-3.1.1 Integrated
wordprocessor/dbase/spreadsheet/drawing/chart/br
tr-openoffice.org-2.4.2_3 Integrated
wordprocessor/dbase/spreadsheet/drawing/chart/br
droideka#
```

Bu sistemden 2.4.2 sürümünü kaldırmak için ise

```
droideka# pkg_delete tr-openoffice.org-2.4.2_3
```

komutunu verdiğimde tr-openoffice.org-2.4.2_3 paketi sistemden kaldırılacaktır. kaldırmak istediğiniz paket sistemde kurulu olan başka paketler için gerekli ise yani diğer paketler buna bağımlı ise söz konusu paket kaldırılmayacaktır. Ancak zorlayarak -f seçeneğini kullanarak paketi yine de kaldırabilirsiniz.

Kurulu Paketleri Terfi Etmek/Güncellemek

Bir paketin eski sürümünü yeni sürümüne pkg_add ile terfi edebilirsiniz. Bunun yolu ise öncelikle eski paketin sistemden kaldırılması ile ardından yeni paketin kurulması ile gerçekleşir. Doğrudan pkg_add ile paketin güncel sürümünü kurmaya çalışırsanız sistemde eski sürümün bulunduğunu belirten bir hata mesajı döndürülecek ve işlem gerçekleşmeyecektir. Dolayısıyla önce eski sürümü kaldırıp ardından yeni sürümü kurmak gerekir.

Port Ağacı ve Yapısı

UNIX sistemler için yazılım kurmak her zaman için kaynak koddan derlenerek yapıla gelmiştir. Sistem yöneticisinin zamanını fazlasıyla alan bir işlem ola gelmiştir. Sistem yöneticisi öncelikle kurmak istediği programın kaynak kodlarını bulmak, indirmek, geçici bir dizine açıp benioku-README.TXT dosyasının içeriğini okuduktan sonra, bu dosyada belirtilen kütüphane vb kontrol ettikten sonra yapılandırma komutunu - configure - çalıştırırdı. Bu yapılandırma komutu sistemi kontrol ederek gereken kütüphaneler vb. uygunluğu denetlenirdi. Sistem yöneticisi bu yapılandırma aşamasında yapılandırma işleminin döndürdüğü mesajları inceleyip gereken işlemleri yapmak durumunda idi. Derlenecek olan program aynı olsa bile farklı UNIX sistemlerin yapılandırmalarının farklı olması nedeni ile -örneğin IBM-AIX ile HP-UX veya SUN SOLARIS sistemlerin döndürecekleri mesajlar farklı olduğu için- sistem yöneticisinin sorumlu olduğu sistemi çok iyi bilmesi gerekli idi. Bazı sistemlerde derleyici programları olmadığı için sistem yöneticisinin ayrıca derleyici kurması ve gerekli olan make dosyalarını da kendi hazırlaması söz konusu olabiliyordu. yapılandırma aşamasının başarılı bir şekilde aşılmış olması durumunda sistem yöneticisi ya kendi yazdığı makefile veya şanslı ise tar.gz içinden çıkan makefile dosyasını kullanarak derleme işlemini gerçekleştirirdi. Bu aşamada eğer bir hata mesajı döndürülmediyse şanslı gününüzdesiniz demektir. Derleme işleminin gerçekleşmesi kurulumun tamamlandığı anlamına gelmez. Sistem yöneticisi derleme işleminin yapıldığı dizinde oluşan derlenmiş dosyaları daha sonra uygun dizinlere örneğin /usr/local/bin vb kopyalamak durumundadır. Eğer yine şanslı gününüzde iseniz ve tar.gz içinden çıkan makefile içinde tanımlı olan bir kurulum-install dizini var ise bu durumda make ile yaptığınız derleme işlemini make install ile sonuçlandırıp elle dosyaları kopyalama yükünden kurtulmuş olabilirdiniz.

Bu şekilde paket kurmak sistem yöneticilerinin kabusu olmaktan öteye gidemedi. Bunun temel nedeni yazılımı geliştiren kişinin kullandığı sistem ile kurulumun gerçekleştirileceği sistemlerin farklı, yazılım geliştiricisinin veya geliştiricilerinin destek sunmayışı, derleme için gereken makefile dosyasının ya sistem ile uyumsuz olması ya da makefile bulunmaması ve sistem yöneticisinin bilgi ve deneyimine dayanarak makefile yazmak durumunda olması, derleyicilerin sistem ile uyumsuz olması, derleme sırasında ortaya çıkan hatalar, derleme sonrasında ise kurulumda karşılaşılan sorunlar oldu. DEB, RPM ortaya çıkmasının ardından ve izleyen süreçteki gelişmeler sonrasında RPM'in UNIX sisstemlere aktarılmasına dek başka bir yöntem geliştirilmediği için de UNIX sistemlerin çağ dışı olduğu düşüncesi yaygınlaştı.

BSD sistemlerde yazılım kurma/kaldırma/güncelleme vb işlemler için kaynak koddan derleme geleneksel bir yöntem olarak yerini korumaktadır. Ancak yukarıdaki problemler bugün söz konusu değildir. Kaynak koddan yine yazılım kurulmaktadır ve problemlerin

ortadan kaldırılması için gereken araçlar elinizin altındadır. Yazının ilk bölümünde port ağacından kısaca söz ettik. BSD sistemlerde yazılım kurmak, kaldırmak ve güncellemek için pkg_* gerekli olan araçları sunar. Paketler ise sürümün çıkış tarihindeki port ağacının paketlenmesi ile oluşturulur. Paketler aslında port ağacının belli bir tarihteki paketlenmiş halidir demek yanlış olmaz. Paketler dolayısıyla port ağacını esas aldığı için BSD sistemlerde yazılım kurmak için port ağacı asıl kaynaktır diyebiliriz. Port ağacı, kaynak koddan derleme yaparak program kurduğunuz ama bu sürecin kontrollü bir şekilde otomatik olarak gerçekleştirilip kurulumun kayıt altına alınarak paket veritabanına eklenmesini sağlar. Bu işlemi yaparken de kaynak kodlar tanımlanan sunucudan indirilir, derlenir ve gerekli diğer işlemler yapılır.

BSD sistemler hazır paketler sunmakla birlikte kaynak koddan derlemeye de izin vermektedir. Aslında opkg_* ile kullandığınız paketler sadece PORT AĞACI kullanılarak hazırlanmış olan paketlerdir. BSD sistemlerde kullanılan PORT AĞACI, sadece BSD sistemler ile sınırlı değildir. Birçok farklı işletim sistemine aktarılmıştır. Bunlar arasında Gentoo ve Mac OS X/Darwin sayılabilirler.

Port nedir?

Port, tanımlanan kaynak kodun FreeBSD'de kurulması için gereken yamalar ile derleme sürecine ait tüm adımların tanımlandığı yönergeler topluluğudur. FreeBSD ekibi, ports'un sorunsuzca işlemesi için gereken tüm işlemleri kontrol altına alır ve bunları belgeler. Böylelikle sistem yöneticisi yazılımın yapılandırılması işlemleri ile uğraşmak durumunda kalmaz ve asıl işine odaklanabilir.

Port Ağacı/Ports Tree Kurulumu

Eğer FreeBSD sisteminizi kurarken dağıtım seçiminde tümünü seçtiyseniz port ağacı/port koleksiyonu/ports kurulmuş demektir. /usr/ports dizini altında bulabilirsiniz. Eğer freeBSD son sürümünü kurduysanız port ağacı kurulum DVD/CD setlerinin oluşturulduğu güne ait olacaktır. FreeBSD port ağacı sürekli olarak güncellenir bu nedenle elinizdeki port ağacının güncellenmesi gereklidir. FreeBSD port ağacını kurmak, güncellemek için cvs veya portsnap kullanılabilir. Hangi aracı kullanacağınız ise tercihinize ait. portsnap cvs'e göre daha hızlı olduğu için portsnap kullanılması daha kolaydır. FreeBSD kurulumunun ardından port ağacını güncellemek uygun olacaktır. Portağacını güncellemek için öncelikle güncel port ağacının bilgisini edinmeniz gerekir. Bu işlem portsnap fetch ile gerçekleştirilir. Bu işlemin ardından da port ağacının güncel sürümüne terfi işlemi yapılmalıdır, yani portsnap extract. Bu işlemleri ayrı ayrı yapmak yerine bir tek seferde portsnap fetch update ile yapmak zaman kazandıracaktır. Eğer sistemde port ağacı kurulmadıysa portsnap ile kurabilirsiniz. portsnap fetch extract kullanılmalıdır. kurulu olan bir port ağacını güncellemek için portsnap fetch update kullanılmalıdır. Aşağıdaki örnekte port ağacı güncellenmektedir.

```
droideka# cd /usr/ports
droideka# portsnap fetch update
Looking up portsnap.FreeBSD.org mirrors... 3 mirrors found.
Fetching snapshot tag from portsnap1.FreeBSD.org... done.
Fetching snapshot metadata... done.
Updating from Sat Sep 12 01:44:13 EEST 2009 to Tue Sep 15
09:42:35 EEST 2009.
Fetching 4 metadata patches... done.
Applying metadata patches... done.
Fetching 0 metadata files... done.
Fetching 274
patches.....10....20....30....40....50....60....70....80....9
0....100....110....120....130....140....150....160....170....
180....190....200....210....220....230....240....250....260..
..270.. done.
Applying patches... done.
Fetching 19 new ports or files... done.
Removing old files and directories... done.
Extracting new files:
/usr/ports/LEGAL
```

```
/usr/ports/MOVED
/usr/ports/Mk/bsd.gcc.mk
/usr/ports/Mk/bsd.linux-apps.mk
/usr/ports/UPDATING
/usr/ports/archivers/zipios++/
/usr/ports/astro/nightfall/
/usr/ports/audio/abraca/
/usr/ports/audio/calf/
/usr/ports/audio/creox/
/usr/ports/audio/ezstream/
/usr/ports/audio/gmm/
/usr/ports/audio/gnome-media/
/usr/ports/audio/linuxsampler/
/usr/ports/audio/mp3c/
/usr/ports/audio/ncmpcpp/
/usr/ports/audio/pulseaudio/
/usr/ports/audio/raul/
/usr/ports/benchmarks/himenobench/
/usr/ports/benchmarks/hpl/
/usr/ports/biology/crux/
/usr/ports/biology/molden/
/usr/ports/biology/ortep3/
/usr/ports/biology/platon/
/usr/ports/biology/psi88/
/usr/ports/biology/tinker/
/usr/ports/cad/calculix/
/usr/ports/cad/feappv/
/usr/ports/cad/gmsh/
/usr/ports/cad/opencascade/
/usr/ports/cad/p5-Verilog-Perl/
/usr/ports/cad/pdnmesh/
/usr/ports/cad/scotch/
/usr/ports/cad/tochnog/
/usr/ports/comms/wsjt/
/usr/ports/comms/wspr/
/usr/ports/converters/p5-Convert-UUlib/
/usr/ports/databases/Makefile
/usr/ports/databases/maatkit/
/usr/ports/databases/mytop/
/usr/ports/databases/p5-DBD-LDAP/
/usr/ports/databases/p5-DBIx-Class-Schema-Loader/
/usr/ports/databases/phpmyadmin/
/usr/ports/databases/sqlite-ext-inet/
/usr/ports/deskutils/Makefile
/usr/ports/deskutils/dolphin-plugins-mplayerthumbs/
/usr/ports/deskutils/gtg/
/usr/ports/deskutils/vboxgtk/
/usr/ports/devel/Makefile
/usr/ports/devel/icu/
/usr/ports/devel/kdesvn-kde4/
/usr/ports/devel/libU77/
/usr/ports/devel/libmonetra/
/usr/ports/devel/libsigc++20/
/usr/ports/devel/monodevelop-boo/
/usr/ports/devel/monodevelop-database/
/usr/ports/devel/monodevelop-java/
/usr/ports/devel/monodevelop-vala/
/usr/ports/devel/p5-Algorithm-Evolutionary/
/usr/ports/devel/p5-DateTime-Locale/
/usr/ports/devel/p5-DateTime-TimeZone/
/usr/ports/devel/p5-Log-Dispatch/
/usr/ports/devel/papi/
/usr/ports/devel/ptlib26/
/usr/ports/devel/py-mwlib.rl/
/usr/ports/devel/py-pyutil/
/usr/ports/devel/py-yaml/
/usr/ports/editors/Makefile
/usr/ports/editors/emacs22/
/usr/ports/editors/kile-kde4/
/usr/ports/emulators/virtualbox/
/usr/ports/finance/opentaxsolver/
/usr/ports/games/Makefile
/usr/ports/games/d2x-xl/
/usr/ports/games/linux-worldofgoo-demo/
/usr/ports/games/neverball/
/usr/ports/games/powwow/
/usr/ports/games/quake2forge/
/usr/ports/games/xmoto/
/usr/ports/graphics/Makefile
/usr/ports/graphics/cimg/
/usr/ports/graphics/dataplot/
/usr/ports/graphics/f-spot/
```

```
/usr/ports/graphics/f90gl/
/usr/ports/graphics/gplot/
/usr/ports/graphics/ipe/
/usr/ports/graphics/libraw/
/usr/ports/graphics/libwpcg/
/usr/ports/graphics/linux-dri74/
/usr/ports/graphics/linux-f10-libGLU/
/usr/ports/graphics/linux-f8-libGLU/
/usr/ports/graphics/linux-libGLU/
/usr/ports/graphics/linux_dri-devel/
/usr/ports/graphics/linux_dri/
/usr/ports/graphics/p5-Geometry-Primitive/
/usr/ports/graphics/pfstools/
/usr/ports/graphics/pgplot/
/usr/ports/graphics/png/
/usr/ports/graphics/robot/
/usr/ports/graphics/ruby-pgplot/
/usr/ports/graphics/tiff/
/usr/ports/graphics/xd3d/
/usr/ports/graphics/yagf/
/usr/ports/irc/bitlbee/
/usr/ports/irc/epic4/
/usr/ports/irc/inspircd/
/usr/ports/irc/kvirc-devel/
/usr/ports/irc/quassel/
/usr/ports/japanese/Makefile
/usr/ports/japanese/man/
/usr/ports/japanese/tdiary-devel/
/usr/ports/japanese/tdiary/
/usr/ports/java/phpeclipse/
/usr/ports/lang/gcc43/
/usr/ports/lang/libhx/
/usr/ports/lang/ratfor/
/usr/ports/mail/enigmail-thunderbird/
/usr/ports/mail/postfix/
/usr/ports/mail/proxsmtp/
/usr/ports/math/R/
/usr/ports/math/algae/
/usr/ports/math/arpack/
/usr/ports/math/atlas-devel/
/usr/ports/math/atlas/
```

```
/usr/ports/math/bihar/
/usr/ports/math/blacs/
/usr/ports/math/blas/
/usr/ports/math/blocksolve95/
/usr/ports/math/clp/
/usr/ports/math/eispack/
/usr/ports/math/elmer-umfpack/
/usr/ports/math/femlab/
/usr/ports/math/fftw/
/usr/ports/math/freemat/
/usr/ports/math/gnuplot/
/usr/ports/math/gotoblas/
/usr/ports/math/gretl/
/usr/ports/math/jags/
/usr/ports/math/jakarta-commons-math/
/usr/ports/math/kaskade/
/usr/ports/math/lapack++/
/usr/ports/math/lapack/
/usr/ports/math/lapack95/
/usr/ports/math/libRmath/
/usr/ports/math/linpack/
/usr/ports/math/metis-edf/
/usr/ports/math/mumps/
/usr/ports/math/nsc2ke/
/usr/ports/math/octave-devel/
/usr/ports/math/octave/
/usr/ports/math/p5-Geo-Distance/
/usr/ports/math/p5-Statistics-Basic/
/usr/ports/math/petsc/
/usr/ports/math/plplot/
/usr/ports/math/py-numpy/
/usr/ports/math/py-symeig/
/usr/ports/math/qd/
/usr/ports/math/qrupdate/
/usr/ports/math/rkward/
/usr/ports/math/scalapack/
/usr/ports/math/scilab/
/usr/ports/math/sdpa/
/usr/ports/math/sdpara/
/usr/ports/math/suitesparse/
/usr/ports/math/superlu/
```

```
/usr/ports/math/superlu_mt/
/usr/ports/math/taucs/
/usr/ports/misc/Makefile
/usr/ports/misc/compat5x/
/usr/ports/misc/compat6x/
/usr/ports/misc/freebsd-doc-all/
/usr/ports/misc/freebsd-doc-en/
/usr/ports/misc/freebsd-doc-hu/
/usr/ports/misc/gnuls/
/usr/ports/misc/heyu2/
/usr/ports/misc/ipa_conv/
/usr/ports/misc/mc/
/usr/ports/misc/sloccount/
/usr/ports/multimedia/kbtv/
/usr/ports/multimedia/libtuner/
/usr/ports/multimedia/motion/
/usr/ports/multimedia/vlc/
/usr/ports/net-im/telepathy-farsight/
/usr/ports/net-im/telepathy-gabble/
/usr/ports/net-im/telepathy-salut/
/usr/ports/net-mgmt/Makefile
/usr/ports/net-mgmt/cowpatty/
/usr/ports/net-mgmt/fetchconfig/
/usr/ports/net-mgmt/nagiosagent/
/usr/ports/net-mgmt/p5-Net-Abuse-Utils/
/usr/ports/net-p2p/giftoxic/
/usr/ports/net-p2p/p5-Net-BitTorrent/
/usr/ports/net-p2p/transmission-daemon/
/usr/ports/net/blam/
/usr/ports/net/cnupm/
/usr/ports/net/dictd-database/
/usr/ports/net/libbgpdump/
/usr/ports/net/liveMedia/
/usr/ports/net/mpich2/
/usr/ports/net/netatalk/
/usr/ports/net/opal3/
/usr/ports/net/openmpi/
/usr/ports/net/p5-Geo-IPfree/
/usr/ports/net/pear-Net_SMTP/
/usr/ports/net/pvm/
/usr/ports/net/ruby-mpi/
```

```
/usr/ports/net/vnstat/
/usr/ports/net/wmwifi/
/usr/ports/net/x11vnc/
/usr/ports/net/xmlrpc++/
/usr/ports/ports-mgmt/Makefile
/usr/ports/ports-mgmt/portmk/
/usr/ports/ports-mgmt/portrac/
/usr/ports/print/pear-File_PDF/
/usr/ports/science/2dhf/
/usr/ports/science/abinit/
/usr/ports/science/at/
/usr/ports/science/avogadro/
/usr/ports/science/cdf/
/usr/ports/science/cgnslib/
/usr/ports/science/dcl/
/usr/ports/science/elmer-eio/
/usr/ports/science/elmer-fem/
/usr/ports/science/elmer-hutiter/
/usr/ports/science/gamess/
/usr/ports/science/getdp/
/usr/ports/science/ghemical/
/usr/ports/science/hdf/
/usr/ports/science/hdf5-18/
/usr/ports/science/hdf5/
/usr/ports/science/isaac-cfd/
/usr/ports/science/libctl/
/usr/ports/science/libghemical/
/usr/ports/science/mbdyn/
/usr/ports/science/mpb/
/usr/ports/science/mpqc/
/usr/ports/science/ncs/
/usr/ports/science/netcdf/
/usr/ports/science/oases/
/usr/ports/science/peekabot/
/usr/ports/science/pnetcdf/
/usr/ports/science/psi3/
/usr/ports/science/py-scipy/
/usr/ports/science/silo/
/usr/ports/science/vis5d+/
/usr/ports/security/amavisd-new/
/usr/ports/security/vuxml/
```

```
/usr/ports/security/wipe/
/usr/ports/sysutils/cdrdao/
/usr/ports/sysutils/cdrtools-devel/
/usr/ports/sysutils/cdrtools/
/usr/ports/sysutils/di/
/usr/ports/sysutils/gsmartcontrol/
/usr/ports/sysutils/knutclient/
/usr/ports/sysutils/p5-Schedule-Cron/
/usr/ports/sysutils/p5-Sysadm-Install/
/usr/ports/sysutils/rsyslog4/
/usr/ports/sysutils/screen/
/usr/ports/sysutils/stress/
/usr/ports/sysutils/symon/
/usr/ports/sysutils/wmtop/
/usr/ports/textproc/discount/
/usr/ports/textproc/libxml2/
/usr/ports/textproc/libyaml/
/usr/ports/textproc/mdocml/
/usr/ports/www/Makefile
/usr/ports/www/dokeos/
/usr/ports/www/havp/
/usr/ports/www/ikiwiki/
/usr/ports/www/mod_flickr/
/usr/ports/www/nginx-devel/
/usr/ports/www/nginx/
/usr/ports/www/p5-Catalyst-View-TT/
/usr/ports/www/p5-REST-Google-Apps-Provisioning/
/usr/ports/www/p5-WWW-Plurk/
/usr/ports/www/php-templates/
/usr/ports/www/tdiary-devel/
/usr/ports/www/tdiary/
/usr/ports/www/webserver/
/usr/ports/www/webservices/
/usr/ports/www/xapian-omega/
/usr/ports/www/ziproxy/
/usr/ports/x11-clocks/wmbday/
/usr/ports/x11-clocks/wmtimer/
/usr/ports/x11-fm/gnome-commander2/
/usr/ports/x11-toolkits/Makefile
/usr/ports/x11-toolkits/gambas2-gb-qt/
/usr/ports/x11-toolkits/lesstif/
/usr/ports/x11-toolkits/py-traitsbackendwx/
/usr/ports/x11-wm/Makefile
/usr/ports/x11-wm/matwm2/
/usr/ports/x11/wmcliphist/
/usr/ports/x11/xfce4-clipman-plugin/
Building new INDEX files... done.
droideka#
```

Ports güncellenmesinin ardından paket varitabanı gibi port veritabanını da güncellenmesi işlemi yapılmalıdır. portsdb -u komutu port ağacı bilgilerini günceller.

```
droideka# portsdb -u
[Updating the portsdb <format:bdb_btree> in /usr/ports ... -
20685 port entries found
.........1000.........2000.........3000.........4000.........
5000.........6000.........7000.........8000.........9000.....
....10000.........11000.........12000.........13000.........1
4000.........15000.........16000.........17000.........18000.
........19000.........20000...... ..... done]
droideka#
```

Portsnap İşleyişi

Portsnap, FreeBSD portsnap sunucularını arar. Eriştiği sunucudan port ağacının güncel sürümne ait olan en son kaydı indirir. Bu kaydı bütünlük kontrolünden geçirir ve bütünlük kontrolünün başarıyla sonuçlanmasının ardından port ağacının güncel sürümü /usr/ports dizine açılır. Port ağacı üzerinde yaklaşık olarak 20700 civarında yazılım yer almaktadır. Bu nedenle port ağacını kullanabilmek FreeBSD'de yazılım kurmak ve kaldırmak için kritik öneme sahiptir. Aşağıda /usr/ports dizinindeki dizinler ve dosyalar görülmektedir.

```
droideka# ls -l
total 113674
-rw-r--r--     1 root  wheel        19 15 Tem  1997
.cvsignore
-rw-r--r--     1 root  wheel   1770805 27 Eyl 14:27
```

```
.portsnap.INDEX
-rw-r--r--     1 root  wheel     56211 15 Eyl 14:03 CHANGES
-rw-r--r--     1 root  wheel      1499  1 Oca  2009 COPYRIGHT
-rw-r--r--     1 root  wheel      2170 18 Eyl 12:17 GIDs
-rw-r--r--     1 root  wheel  17370624 27 Eyl 14:27 INDEX-5
-rw-r--r--     1 root  wheel  20007920 27 Eyl 14:27 INDEX-6
-rw-r--r--     1 root  wheel  20002585 27 Eyl 14:27 INDEX-7
-rw-r--r--     1 root  wheel  35136512 27 Eyl 14:42 INDEX-7.db
-rw-r--r--     1 root  wheel      4498 15 Eyl 16:32 KNOBS
-rw-r--r--     1 root  wheel     33702 14 Eyl 20:57 LEGAL
-rw-r--r--     1 root  wheel    312974 24 Eyl 23:56 MOVED
-rw-r--r--     1 root  wheel      6099 22 Ağu 22:32 Makefile
drwxr-xr-x     2 root  wheel      1024 26 Eyl 23:43 Mk
-rw-r--r--     1 root  wheel      1298 22 May  2006 README
-rw-r--r--     1 root  wheel      5119 18 Eyl 00:18
README.html
drwxr-xr-x     2 root  wheel       512  7 Ağu 13:51 Templates
drwxr-xr-x     4 root  wheel       512 15 Eyl 17:55 Tools
-rw-r--r--     1 root  wheel     11117 18 Eyl 12:17 UIDs
-rw-r--r--     1 root  wheel    307465 25 Eyl 17:24 UPDATING
drwxr-xr-x    26 root  wheel      1024 18 Eyl 00:18
accessibility
drwxr-xr-x    12 root  wheel       512 18 Eyl 00:18 arabic
drwxr-xr-x   189 root  wheel      4096 22 Eyl 17:11 archivers
drwxr-xr-x   109 root  wheel      2560 26 Eyl 23:43 astro
drwxr-xr-x   857 root  wheel     18432 26 Eyl 23:43 audio
drwxr-xr-x    66 root  wheel      1536 22 Eyl 17:11 benchmarks
drwxr-xr-x    98 root  wheel      2048 24 Eyl 18:06 biology
drwxr-xr-x    87 root  wheel      2048 24 Eyl 18:06 cad
drwxr-xr-x   139 root  wheel      3072 18 Eyl 00:20 chinese
drwxr-xr-x   161 root  wheel      3072 22 Eyl 17:11 comms
drwxr-xr-x   126 root  wheel      3072 22 Eyl 17:11 converters
drwxr-xr-x   687 root  wheel     16896 26 Eyl 23:43 databases
drwxr-xr-x   280 root  wheel      6144 26 Eyl 23:43 deskutils
drwxr-xr-x  2978 root  wheel     69632 27 Eyl 14:27 devel
drwxr-xr-x    21 root  wheel     10752 24 Eyl 18:12 distfiles
drwxr-xr-x   140 root  wheel      3072 26 Eyl 23:43 dns
drwxr-xr-x   251 root  wheel      5120 27 Eyl 14:27 editors
drwxr-xr-x   209 root  wheel      4608 26 Eyl 23:43 emulators
drwxr-xr-x    88 root  wheel      3072 22 Eyl 17:11 finance
drwxr-xr-x    30 root  wheel      1024 18 Eyl 00:27 french
```

```
drwxr-xr-x   121 root  wheel      2560 26 Eyl 23:43 ftp
drwxr-xr-x  1072 root  wheel     20992 26 Eyl 23:43 games
drwxr-xr-x    45 root  wheel      1536 22 Eyl 17:11 german
drwxr-xr-x   962 root  wheel     19456 26 Eyl 23:43 graphics
drwxr-xr-x    11 root  wheel       512 18 Eyl 00:30 hebrew
drwxr-xr-x    12 root  wheel       512 18 Eyl 00:30 hungarian
drwxr-xr-x   145 root  wheel      3072 26 Eyl 23:43 irc
drwxr-xr-x   391 root  wheel      8192 24 Eyl 18:06 japanese
drwxr-xr-x   164 root  wheel      4096 22 Eyl 17:11 java
drwxr-xr-x    61 root  wheel      1536 24 Eyl 21:31 korean
drwxr-xr-x   379 root  wheel      7680 27 Eyl 14:27 lang
drwxr-xr-x   735 root  wheel     16896 27 Eyl 14:27 mail
drwxr-xr-x   525 root  wheel     12288 26 Eyl 23:43 math
drwxr-xr-x    15 root  wheel       512 18 Eyl 00:33 mbone
drwxr-xr-x   614 root  wheel     13312 22 Eyl 17:11 misc
drwxr-xr-x   318 root  wheel      6656 26 Eyl 23:43
multimedia
drwxr-xr-x  1104 root  wheel     23552 26 Eyl 23:43 net
drwxr-xr-x   168 root  wheel      3584 26 Eyl 23:43 net-im
drwxr-xr-x   278 root  wheel      6144 26 Eyl 23:43 net-mgmt
drwxr-xr-x   143 root  wheel      3584 24 Eyl 21:31 net-p2p
drwxr-xr-x   106 root  wheel      2560 24 Eyl 18:06 news
drwxr-xr-x    44 root  wheel      1536 18 Eyl 00:37 palm
drwxr-xr-x    22 root  wheel      1024 18 Eyl 00:37 polish
drwxr-xr-x    65 root  wheel      1536 22 Eyl 17:11 ports-
mgmt
-rw-r--r--     1 root  wheel  20565544 27 Eyl 13:52
ports_print_index.txt
drwxr-xr-x    17 root  wheel      1024 18 Eyl 00:37
portuguese
drwxr-xr-x   348 root  wheel      7168 26 Eyl 23:43 print
drwxr-xr-x    46 root  wheel      1536 24 Eyl 18:06 russian
drwxr-xr-x   135 root  wheel      2560 27 Eyl 14:27 science
drwxr-xr-x   865 root  wheel     18432 26 Eyl 23:43 security
drwxr-xr-x    44 root  wheel      1024 22 Eyl 17:11 shells
drwxr-xr-x   912 root  wheel     17920 26 Eyl 23:43 sysutils
drwxr-xr-x  1236 root  wheel     28672 26 Eyl 23:43 textproc
drwxr-xr-x    11 root  wheel       512 18 Eyl 00:41 ukrainian
drwxr-xr-x    23 root  wheel      1024 18 Eyl 00:41
vietnamese
drwxr-xr-x  1855 root  wheel     47616 26 Eyl 23:43 www
```

```
drwxr-xr-x   495 root  wheel     10240 26 Eyl 23:43 x11
drwxr-xr-x    65 root  wheel      1536 18 Eyl 00:45 x11-clocks
drwxr-xr-x    71 root  wheel      2560 18 Eyl 00:45 x11-
drivers
drwxr-xr-x    44 root  wheel      1024 22 Eyl 17:11 x11-fm
drwxr-xr-x   145 root  wheel      3584 24 Eyl 18:06 x11-fonts
drwxr-xr-x    12 root  wheel       512 18 Eyl 00:45 x11-
servers
drwxr-xr-x   213 root  wheel      6144 18 Eyl 00:45 x11-themes
drwxr-xr-x   321 root  wheel      7168 26 Eyl 23:43 x11-
toolkits
drwxr-xr-x   146 root  wheel      3072 24 Eyl 18:06 x11-wm
droideka#
```

Dizinler dışında yer alan dosyalardan port_print_index.txt dışındakiler standart dosyalar ve dizinlerdir. Bu dosyalardan CHANGES, COPYRIGHT, GID, KNOBS, LEGAL, MOVED ve Makefile port ağacı kullanırken çok sık başvuracağınız dosyalardır.

CHANGES dosyası, FreeBSD port ağacı üzerinde yapılan değişiklikleri kapsar. Bu dosya port ağacının işleyişi hakkında bilgi edinmek için başlangıç noktasıdır.

COPYRIGHT dosyası port ağacının bütününe ilişkin lisansı tanımlar. Port ağacı üzerinde yer alan her bir tekil port'ta yer alan yazılım kendi lisansına sahip olmakla birlikte port ağacının lisansı BSD lisansısıdır.

GID dosyası port ağacında yer alan yazılımların grup ID değerini tanımlar. Birçok yazılımın doğru biçimde çalışması için gerek normal kullanıcı gerek özel bir kullanıcı olarak bir grup ID sahip olması gereklidir. Böylelikle sistemde kurulan yazılımların hepsinin sorunsuzca çalışması sağlanır.

KNOBS dosyası, port ağacı üzerinde yer alan tüm yazılımların derlenmesi sırasında yapılacak ince ayar için gereken değişkenleri tanımlar. Bu değişkenler her bir port için ayrıca tanımlanabileceği gibi /etc/make.conf dosyasından da yapılabilir.

LEGAL dosyası, port ağacı üzerinde yer alan yazılımlara ilişkin yasal düzenlemeleri tanımlar. Örneğin ticari kullanımının-no commercial use- yasak olması, yeniden dağıtım -redistribution- kısıtlaması, ücretli kullanım -no monetary gain- vb. kısıtlamaları tanımlar. Bu kısıtlamalar içerisinde örneğin LAME geliştiricilerinin belirttiği gibi sadece kaynak kod olarak dağıtılması gibi sınırlamalar tanımlanır. Port ağacı üzerinde yer alan yazılımların hemen hemen tamamınına yakını özgür yazılımlar olsa da bazılarının lisansları kısıtlamaları barındırmaktadır. Burada karşılaşabileceğiniz en yaygın kısıtlama yeniden dağıtımın -prohibition on redistribution- sınırlanmasıdır. FreeBSD geliştiricileri bu dosyaları kendi sunucularında barındırmaz. Elinizdeki CD/DVD setlerinde de bu dosyalar bulunmaz. Onun yerine bu dosyaları nasıl derleyeceğiniz ve kullanacağınıza ilişkin yönergeler port ağacı üzerinde yer alır. Böylelikle yazılımı kendiniz derleyerek kullanabilirsiniz. Örneğin uzun bir süre JAVA lisansının yapısı gereği FreeBSD'de yer almadı. FreeBSD geliştircileri FreeBSD ile kullanılabilecek olan derlenmiş bir JAVA paketi sunamadı. Ayrıca derlenebilecek olan bir JAVA kodu da sunamadı. Ancak SUN sitesinden indirilecek olan JAVA kodunun nasıl derlenerek sistemde kullanılabileceğine ilişkin bir yönerge sunması için bir kısıtlama olmadığı için FreeBSD geliştiricileri JAVA için bir port oluşturup burada derleme yönergelerini sundular. Böylece JAVA'ya gereksinim duyan bir FreeBSD kullanıcısı SUN sitesinden JAVA kodunu indirip kendisi derleyerek sistemine JAVA kurabiliyordu. Bugün ise FreeBSD Foundation- FreeBSD Vakfı, bu lisans sorununu aşarak kullanıcılara bütün bir JAVA paketi sunabiliyor. JAVA paketinin lisans anlaşmasını kabul ederek freeBSD Foundation sitesinden indireceğiniz JAVA kodunu kullanarak FreeBSD sisteminize JAVA kurup kullanabilirsiniz. Benzer bir durum bazı yazılımların ticari uygulamalarda kullanılmasını kısıtlamaktadır. Bundan dolayı söz konusu kodu kendi geliştireceğiniz FreeBSD çalıştıran ürünlerde kullanmanız söz konusu olmayacaktır. Benzer olarak kriptografi yazılımlarının derlenmiş olarak ABD dışına ihracı yasal olarak olanaklı değildir. Bunun yerine kaynak kodu port

ağacı ile indirip derleyerek kullanabilirsiniz. Kısıtlamalar tüm port ağacı üzerinde iki elin parmaklarını aşmayacak kadardır. Bu nedenle lisansların getirmiş olduğı kısıtlama sadece bir istisnadır.

MOVED, port ağacı üzerindeki bir bölümden bir başka bölüme taşınan yazılımları tanımlar. Böylece port yönetim araçları olan portmaster bu değişiklikleri izleyerek port ağacı üzerinde çalışabilir.

Makefile, tüm portağacı için tanımlı olan makefile dosyasıdır. Yüksek seviyeli yapılandırma yönergelerini barındırır.

Mk dizini ise tüm port apacı için düşük seviyeli yönergeleri barındırır. Port ağacı üzerinde yer alan uygulamaların sorunsuz olarak kurulup kullanılabilmesi için bu dizinde yer alan mk uzantılı dosyalar ait olduğu port üzerindeki yazılımın freeBSD sisteme sorunsuz olarak kurulmasını ve kullanılmasını sağlayan yönergeleri barındırır.

```
droideka# ls -l
total 762
-rw-r--r--  1 root  wheel   13851 20 Eyl  2007 bsd.apache.mk
-rw-r--r--  1 root  wheel    9904 19 Ağu 05:26
bsd.autotools.mk
-rw-r--r--  1 root  wheel    2382  2 Eyl 01:19 bsd.cmake.mk
-rw-r--r--  1 root  wheel    3312  3 Ağu 18:36 bsd.commands.mk
-rw-r--r--  1 root  wheel   14492  3 Eyl 18:23 bsd.database.mk
-rw-r--r--  1 root  wheel    7660 12 Mar  2008 bsd.destdir.mk
-rw-r--r--  1 root  wheel   13383 19 Nis  2008 bsd.efl.mk
-rw-r--r--  1 root  wheel    7742 23 Haz 18:05 bsd.emacs.mk
-rw-r--r--  1 root  wheel    5409 13 Tem 22:29 bsd.fpc.mk
-rw-r--r--  1 root  wheel    6381 26 Eyl 04:02 bsd.gcc.mk
-rw-r--r--  1 root  wheel   20827 18 Tem 14:10 bsd.gecko.mk
-rw-r--r--  1 root  wheel   34825 22 Ağu 20:45 bsd.gnome.mk
-rw-r--r--  1 root  wheel   20002 20 Mar  2009 bsd.gnustep.mk
-rw-r--r--  1 root  wheel    6777  6 Nis 13:46
bsd.gstreamer.mk
-rw-r--r--  1 root  wheel   20990 22 May 07:11 bsd.java.mk
-rw-r--r--  1 root  wheel    4953 11 Haz 12:09 bsd.kde.mk
-rw-r--r--  1 root  wheel    5196  2 Eyl 01:19 bsd.kde4.mk
-rw-r--r--  1 root  wheel    3126 20 Tem 11:13 bsd.ldap.mk
-rw-r--r--  1 root  wheel   23927 14 Eyl 13:09 bsd.linux-
apps.mk
-rw-r--r--  1 root  wheel    7898  1 Tem 23:35 bsd.linux-
rpm.mk
-rw-r--r--  1 root  wheel     771 10 Ara  2006 bsd.local.mk
-rw-r--r--  1 root  wheel   18951 25 Ağu  2008 bsd.lua.mk
-rw-r--r--  1 root  wheel    1956  4 Ağu  2007 bsd.mail.mk
-rw-r--r--  1 root  wheel    5514 19 May  2008 bsd.ocaml.mk
-rw-r--r--  1 root  wheel    2435 23 Ağu 19:49 bsd.octave.mk
-rw-r--r--  1 root  wheel    4987 21 Tem 20:51 bsd.openssl.mk
-rw-r--r--  1 root  wheel    9930  8 Haz 17:56 bsd.perl.mk
-rw-r--r--  1 root  wheel   12680 15 Haz 12:22 bsd.php.mk
-rw-r--r--  1 root  wheel  215056 21 Eyl 22:13 bsd.port.mk
-rw-r--r--  1 root  wheel     556 25 May  2007
bsd.port.options.mk
-rw-r--r--  1 root  wheel     139 25 Ağu  1999
bsd.port.post.mk
-rw-r--r--  1 root  wheel     140 25 Ağu  1999
bsd.port.pre.mk
-rw-r--r--  1 root  wheel   17022 19 Tem  2008
bsd.port.subdir.mk
-rw-r--r--  1 root  wheel   26816 22 Ağu 10:22 bsd.python.mk
-rw-r--r--  1 root  wheel    6356 29 Ağu 10:34 bsd.qt.mk
-rw-r--r--  1 root  wheel   18288 19 Tem 19:43 bsd.ruby.mk
-rw-r--r--  1 root  wheel    2492 14 Ağu 14:34 bsd.scons.mk
-rw-r--r--  1 root  wheel    4286  2 Ağu 11:35 bsd.sdl.mk
-rw-r--r--  1 root  wheel   68838  2 Eyl 18:01 bsd.sites.mk
-rw-r--r--  1 root  wheel   14635  7 Tem 10:34 bsd.tcl.mk
-rw-r--r--  1 root  wheel   17489 13 Haz 19:35 bsd.wx.mk
-rw-r--r--  1 root  wheel    2307  6 May 17:56 bsd.xfce.mk
-rw-r--r--  1 root  wheel   14457 25 Nis 22:49 bsd.xorg.mk
droideka#
```

README, Port ağacının tanıtımını barındırır. Bu yazı README dosyasından yararlanılarak yazılmıştır.

Templates dizini diğer port ağacı kullanılarak kurulacak olan yazılımlar için gereken temel dosyaları barındırır.

Tools dizini, port geliştiricileri tarafından kullanılan araçlar ve yazılımları barındırır.

UID, port ağacı tarafından kullanılan normal kullanıcı user ID değerlerini barındırır. Port ağacı üzerinde yer alan yazılımlar için gereken user ID değerleri bu dosyadan okunur.

UPDATING dosyası, port ağacı kullanılırken yazılımların güncellenmesi sırasında ortaya çıkabilecek olası sorunların önüne geçilmesi için gerekli olan işlemleri anlatan dosyadır. Bir port güncellenirken öncelike bu dosyaya bakılması gerekir.

distfiles dizini, kaynak kodların indirildiği dizindir. Port üzerinden kurulum yaptığınızda kaynak kodlar internetten indirilip bu dizine kayıt edilir. Eğer indirdiğiniz kaynak koda yeniden gereksiniminiz varsa /usr/ports/distfiles dizinine göz atın.

Diğer dizinler port ağacında tanımlı olan bölümlerdir. Bu bölümler dizin adından anlaşılacağı gibi özel uygulamalara aittir. Örneğin mail dizini e-posta uygulamalarını barındırır. Bir bölüm altında birçok uygulama yer alır.

Port Ağacını Kullanarak Yazılım Kurmak

Port ağacını kullanarak yazılım kurmak paketleri kullanarak yazılım kurmaya göre daha fazla zaman alır. Port ağacını kullanacak iseniz sadece zamana değil aynı zamanda da aktif bir internet erişimine gereksiniminiz vardır. Hazır derlenmiş paketlere göre port ağacını kullanarak derleme yapılarak yazılım kurmak ise daha iyi optimize edilmiş yazılım kullanımını sağlar. Bunu görebileceğiniz en uygun yazılım ise tüm port ağacı üzerindeki en büyük tekil yazılım olan openoffice.org 'tur. port ağacı üzerinde openoffice.org'a ait olan birden çok port bulunmaktadır. bunlar arasında openoffice.org'un 2.x serisi ile 3.x kararlı sürümleri üzerinden devam edeceğiz. Openoffice.org. 3.1.1 sürümünü kurabileceğiniz openoffice.org-3 portunun yer aldığı /usr/ports/editors/openoffic.org-3 dizininde aşağıdaki dizinleri ve dosyaları görebilirsiniz.

```
droideka# cd /usr/ports/editors/openoffice.org-3
droideka# pwd
/usr/ports/editors/openoffice.org-3
droideka# ls
Makefile                        distinfo
openoffice.org-3.1.1.tbz        pkg-plist
README.html                     files
pkg-descr                       work
droideka# ls -l
total 146972
-rw-r--r--  1 root  wheel      11559  2 Eyl 22:57 Makefile
-rw-r--r--  1 root  wheel       5116 18 Eyl 00:46 README.html
-rw-r--r--  1 root  wheel        981 31 Ağu 13:47 distinfo
drwxr-xr-x  2 root  wheel       1024 31 Ağu 20:52 files
-rw-r--r--  1 root  wheel  150361919  3 Eyl 05:37
openoffice.org-3.1.1.tbz
-rw-r--r--  1 root  wheel        830 19 Eki  2006 pkg-descr
-rw-r--r--  1 root  wheel         70 19 Eki  2006 pkg-plist
drwxr-xr-x  4 root  wheel       1024  3 Eyl 05:37 work
droideka#
```

Openoffice.org 3.1.1 sürümünü önceden derlediğim için bu dizinde openoffice.org-3.1.1.tbz paketini de görebilirsiniz. Bunu nasıl hazırladığımı daha sonra ele alacağım. Bu dizinde gördüğünüz Makefile derleme işlemi için gerekli olan yönergeleri barındırır. Makefile'a baktığınızda openoffice.org ile ilgili bir açıklama vb göremeyeceksiniz. Ancak makefile içinde openoffice.org-3 portunun derlenmesi için gereken yönergeler bulunmaktadır. Böylelikle openoffice.org-3.x kaynak kodlarını FreeBSD üzerinde nasıl derleneceği kesin olarak belirtilmiştir.

Distinfo dosyası portun indireceği kaynak kod için gerekli olan bütünlük kontrolü değerlerini barındırır. Böylelikle kuruluma başladığınızda kaynak kod dosyasının sorunsuzca indirilmiş

olduğundan ve daha da önemlisi dosyanın bütünlüğünden -birilerinin kurcalamadığından- emin olabilirsiniz. Files dizini altında openoffice.org-3.x derlenmesi için gereken yamalar ve diğer dosyalar yer almaktadır. Bu yamaların büyük bir kısmı openoffice.org'un FreeBSD paket yönetimi ile uyumlu olmasını sağlamaya yöneliktir. pkg-descr yazılımı tanımlamayan tanıtım bilgisini barındırır. pkg-plist dosyası paketleme listesidir ve kurulacak olan dosyaların listesini barındırır.

Bütün bu dosyalar openoffice.org'un kurulması için gereken tüm işlem basamaklarını tanımlar.

Bir Portu Kurmak

Port ağacı, bir çok dizin ile bunların altdizinlerinden oluşan bir yazpıdır. Port ağacı üzerinde yer alan dizinler içerisinde kaynak kod bulunmaz. Yukarıda ıopenoffice.org-3 portuna gözattığımızda kaynak kodun bulunmadığını hatta distfiles altında da kaynak kod olmadığını görebilirsiniz. Bu durumda FreeBSD'de port kullanarak nasıl kaynak koddan kurulum yapılıyor sorusu akla gelir.

Bir portu kurmak istediğinizde FreeBSD, o portun belirttiği kaynak kodu tanımlı olan internet sitesinden indirir. İndirilen kod bütünlük kontrolünden geçer ve ardından geçici olarak work dizinine açılır. Bu dizin yukarıda gördüğünüz work adlı dizindir. Bir portu kurmadıysanız work dizin ile içeriğini görmeniz sözkonusu değidir. Portun kurulumu tamamlandığında kuruluma ait olan bilgi /var/db/pkg dizininde saklanır. Eğer bir portun kurulabilmesi için ayrıca kurulması gereken bağımlılıkları var ise bu bağımlılıklar ayrıca diğer portlardan kurular ve aynı şekilde kayıt altına alınır. Bu bağımlıklıkların kurulması için asıl portun kurulması işlemine ara verilir, ilgili bağımlılıklardan eksik olanlar kurulur ve asıl porta geri dönülerek kurulum işlemine devam edilir.

Bir portu kurmak için kullanılabilecek ilk araç make'tir. make tek başına kurulumu gerçekleştirmez. Bunun için make ile birlikte başka seçeneklerin de kullanılması gereklidir.

Kurulumu başlatmak için kurmak istediğiniz portun bulunduğu dizine örneğin openoffice.org-3.1.1 için /usr/ports/editors/openoffice.org-3 dizinine geçip make install komutu verilmelidir.

```
# cd /usr/ports/editors/openoffice.org-3
# make install
```

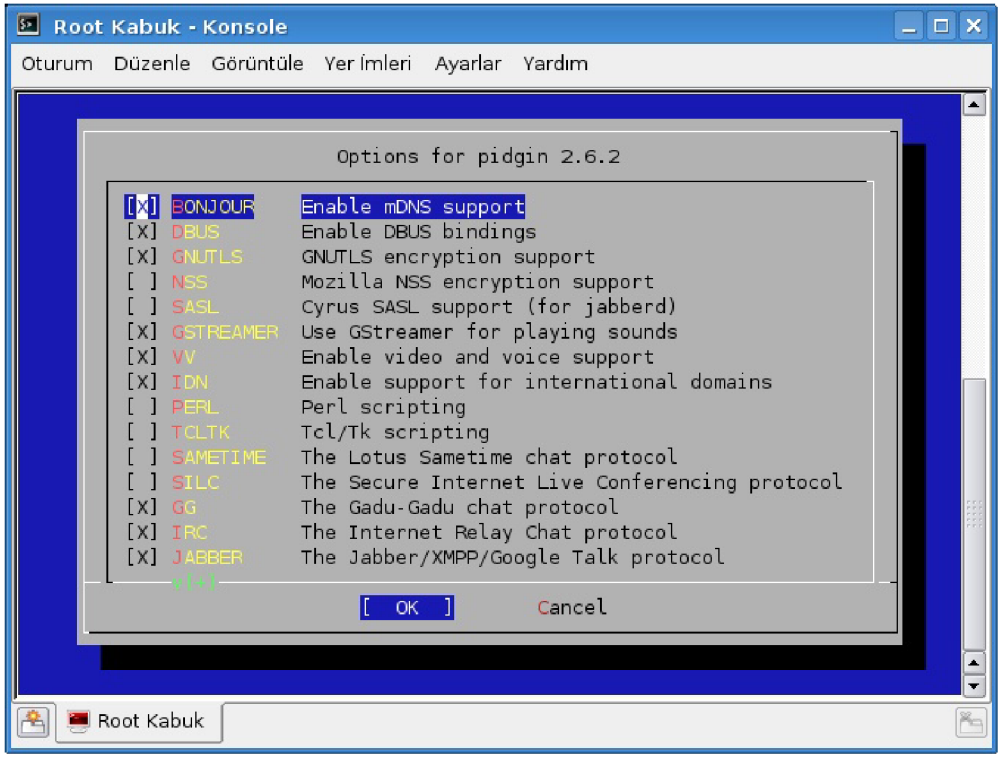
make install ile openoffice.org-3 portu kurulmaya başlanacaktır. Bu aşamada sizin yapacağınız tek işlem eğer varsa portun yapılandırma seçenekleri arasından seçim yapmak olacaktır. OpenOffice.org-3 portunun yapılandırılması gereken bir seçeneği olmadığı için size bir soru sormayacaktır. Kurulum işlemi bitinceye dek diğer işlerinizi yapmaya devam edebilirsiniz. make install gereken tüm işlemleri yapacaktır. Portun kurulması için gereken diğer bağımlılıkların kurulmasından API/ABI yapılandırmasına dek tümü otomatik olarak yapılacaktır.

Zaman içinde port ağacını kullanırken her zaman make install dışında diğer seçenekleri de kullanmak isteyebilirsiniz. Aslında make install tek bir komut gibi görünse de birden çok komutu da çalıştırmaktadır. Bu komutlardan birisini veya birkaçını kullansanız da make install ile kurulum yapmaktan pek farklı olmayacaktır. Örneğin make extract komutu, make config, make fetch, make checksum ve make depends komutlarını da çalıştırmaktadır.

make config

Bazı portlar isteğe balı olarak değişen yapılandırma seçeneklerine sahiptir. Örneğin pidgin gibi. Bu portların yapılandırılması için yaptığınız seçimler yazılımın gelecekteki sürümleri içinde esas alınacaktır. Bu esas alınan yapılandırma ancak ve ancak yazılımın

seçeneklerinde bir değişiklik olmadığı sürece geçerlidir. Aksi söz konusu ise, bu durumda yeni sürüme terfi ederken veya kurarken bu seçeneklerin yeniden belirlenmesi gerekecektir. Bu seçenekler make config ile görüntülebilir.



make fetch

Port yapılandırmasını tamamladıktan sonra önceden belirlenmiş olan internet sitelerinden kaynak kodun indirilmesi gerekir. Portun Makefile'ı kendi içerisinde kaynak kodun indirileceği internet sitelerini barındırır. İlk sıradaki internet aitesinde kaynak kod bulunmuyorsa sıradaki bir sonraki siteyi deneyecektir. Kaynak koda erişildiğinde distfiles dizinine indirecektir. Bu indirme işlemini elle yapmak için make fetch komutunu kullanırız.

make checksum

Bir sonraki aşamada gerçekleştirilen işlem bütünlük kontrolüdür. İndirilen kaynak kodun bütünlüğü kontrol edilir. İnsternet üzerinde yer alan sunucudaki kaynak kodun yetkisiz kişilerce erişilmiş olmasının önüne geçmek için bir önlem olarak kaynak kodun bütünlük kontrolü yapılır. Bu aşamayı başarıyla geçen kaynak kodlar bir sonraki aşamaya geçer

make extract

Kaynak kodun bütünlük kontrolünün ardından indirilen dosyanın geçici bir dizine açılması gereklidir. Bu geçici dizin port altındaki work dizinidir. Kaynak kod derleme için bu dizine açılır.

make patch

Eğer varsa, portta belirtilen yamaları uygular. Eğer sistemde yer alan patch yerine ayrı bir uygulama kullanılacak ise bu komut sözkonusu uygulamayı çalıştırıp yamalama işlemini gerçekleştirir.

make depends

Kodun derlenmesine başlanmadan önce mutlaka bağımlılıklar açısından kontrol edilmesi gerekir. Bu işlem derleme işleminin sorunsuzca gerçekleşmesi için gereklidir. gerekli olan bağımlılıklar ve eğer varsa ve sistem kurulmadıysa bu komut söz konusu eksik olan bağımlılıkları da sisteme kuracaktır.

make configure

Bu komut yazılıma ait olan bir yapılandırma betiğinin bulunup bulunmadığını kontrol edecektir. Eğer söz konusu kaynak koda ait

olan bir yapılandırma betiği bulunuyorsa bu çalıştırılacaktır.

make build

Bu komut indirilmiş bütünlük kontrolünden geçmiş, yapılandırılmış ve yamaları uygulanmaış olan kaynak kodu derleyecektir.

make install

Derlenen kaynak kod bu komut ile sisteme kurulacak ve ardından /var/db/pkg dizini altında kayıt edilecektir.

Port Seçeneklerini Yapılandırmak

Bugün kullandığımız birçok yazılım kendisine ait olan özel yapılandırma seçeneklerine sahiptir. Bu seçenekler bazen doğrudan makefile ile tanımlanır bazen de doğrudan derleyiciye belirtilen seçenekler ile tanımlanır. Bu seçenekleri tanımlamak zor olmamakla birlikte birden fazla yoldan yapılabilir. Bütün tanımlama şekillerini öğrenmek hem zaman alıcı hemde can sıkıcı olabilir. FreeBSD bu işi iki farklı yoldan yapmaktadır. Pidgin portunun yapılandırmasını gördüğünüz resimdeki gibi diyalog kullanılarak yapılandırma sırasında ok tuşları ile dolaşıp boşluk tuşu ile seçim yapabilirsiniz. Bu yapılandırma dosyası eğer portun özel yapılandırma seçenekleri varsa karşınıza gelecektir ve /usr/ports/port_adı/option dosyasına kayıt edilecektir. Bu arayüze make config ile ulaşabilirsiniz.

Bunun yanında bazı portlar halen makefile ile bu yapılandırma seçeneklerini tanımlamaktadır. örneğin openofficeçorg -2.x/3.x portunun grafik yapılandırması bulunmaz. make install komutunu verdiğinizde ekranda kullanılabilecek olan yapılandırma seçenekleri hızlıca geçer ve ardından kaynak kod indirme aşamasına gelinir. Bu durumda CTRL+C basarak işlemi durdurup seçenekleri değerlendirip ardından kuruluma yeniden başlamanız gerekecektir.

örneğin openoffice.org KDE arayüzü ile tümleşik olarak derlemek için

```
make -DWITH_KDE
```

komutunu vermeniz gereklidir. Eğer kurulumu kesip baştan başlamk istemiyorsanız makefile içeriğine bakarak kullanabileceğiniz seçenekleri belirleyebilirsiniz.

Port Makefile

Her port bir makefile ile gelir. Aşağıdaki örnek openoffice.org-3 portuna ait olan makefile içeriğini göstermektedir.

```
droideka# cd /usr/ports/editors/openoffice.org-3
droideka# cat Makefile
# New ports collection makefile for: OpenOffice.org
# Date created:         28 February 2002
# Whom:                 Martin Blapp
#
# $FreeBSD: ports/editors/openoffice.org-3/Makefile,v 1.324
2009/08/31 10:47:55 maho Exp $
#

PORTNAME?=      openoffice.org
PORTVERSION?=   ${OOOVERSION}
CATEGORIES+=    editors java
MASTER_SITES+=  http://ooopackages.good-
day.net/pub/OpenOffice.org/sources/ \
                ftp://ftp.cs.man.ac.uk/pub/toby/gpc/:gpc \
                ${MASTER_SITE_MOZILLA:S/$/:mozsrc/} \

http://tools.openoffice.org/unowinreg_prebuild/680/:unowinreg
MASTER_SITE_SUBDIR+=
mozilla/releases/mozilla${MOZILLA_VERSION}/source/:mozsrc
DISTFILES+=     ${OOOSRC} unowinreg.dll:unowinreg
.if defined(WITH_GPC)
DISTFILES+=     gpc231.tar.Z:gpc
.endif
```

```
EXTRACT_ONLY=   ${OOOSRC}

MAINTAINER=     openoffice@FreeBSD.org
COMMENT?=       Integrated
wordprocessor/dbase/spreadsheet/drawing/chart/browser(release
branch)

OOOVERSION=     3.1.1
NO_LATEST_LINK= yes
USE_GNOME=      desktopfileutils gtk20
MOZILLA_VERSION=        1.7.5
MOZILLA_SOURCE=         mozilla-source-
${MOZILLA_VERSION}.tar.gz
.if !defined(WITHOUT_MOZILLA)
DISTFILES+=     ${MOZILLA_SOURCE}:mozsrc
USE_GNOME+=     libidl
.endif

.if defined(WITH_KDE)
USE_KDELIBS_VER=        3
.endif
USE_XORG=       x11 ice xaw xau xext xrender xrandr \
                xi xt xcursor xdamage xcomposite xfixes
USE_GMAKE=      yes
USE_PERL5=      yes
USE_BZIP2=      yes
WITHOUT_CPU_CFLAGS=     true
MAKE_JOBS_SAFE= yes

.include <bsd.port.pre.mk>

USE_JAVA=       yes
JAVA_BUILD=     jdk
JAVA_VENDOR=    freebsd bsdjava openjdk
.if (${OSVERSION} >= 700000)
JAVA_VERSION=   1.5 1.6
.else
JAVA_VERSION=   1.4 1.5 1.6
.endif

.include <${FILESDIR}/Makefile.localized>
```

```
ONLY_FOR_ARCHS= i386 amd64

.if ${ARCH} == amd64
FREEBSD_ENV_SET=        FreeBSDAMDEnv.Set
.else
FREEBSD_ENV_SET=        FreeBSDX86Env.Set
.endif

MILESTONE?=             19
OOOTAG?=                OOO310_m${MILESTONE}
OOOSRC?=                OOo_${OOOTAG}_source${EXTRACT_SUFX}
OOODIR=                 openoffice.org3
INSTALLATION_BASEDIR?=  openoffice.org-${OOOVERSION}
EXECBASE?=              openoffice.org-${OOOVERSION}
DIST_SUBDIR=            openoffice.org3
SIMPLEOSVER=            ${OSREL:C/\.//g}
.if ${ARCH} == amd64
PACKAGE_BASENAME=
OOo_${OOOVERSION}_${OPSYS}${SIMPLEOSVER}X86-64
.else
PACKAGE_BASENAME=
OOo_${OOOVERSION}_${OPSYS}${SIMPLEOSVER}Intel
.endif

LOCALIZED_LANG?=        tr
# FIXME (Somehow INDEX build fails)
.if defined(LANG_PKGNAME)
PKGNAMEPREFIX=          ${LANG_PKGNAME}-
.endif
.if defined(LANG_SUFFIX)
PKGNAMESUFFIX=          -${LANG_SUFFIX}
.endif

RUN_DEPENDS+=
${LOCALBASE}/share/icons/hicolor/index.theme:${PORTSDIR}/misc
/hicolor-icon-theme

.if defined(WITH_CCACHE)
BUILD_DEPENDS+= ccache:${PORTSDIR}/devel/ccache
CCACHE_PREFIX=  ccache
```

```
OOOCC=          ${CCACHE_PREFIX} ${CC}
OOOCXX=         ${CCACHE_PREFIX} ${CXX}
.else
OOOCC=          ${CC}
OOOCXX=         ${CXX}
.endif
CONFIGURE_ENV+= CC="${OOOCC}" CXX="${OOOCXX}"           \
                CPPFLAGS="-I${LOCALBASE}/include"       \
                LDFLAGS="-L${LOCALBASE}/lib" \
                PATH=${WRKSRC}/solenv/bin:$$PATH

BUILD_DEPENDS+= zip:${PORTSDIR}/archivers/zip \
                unzip:${PORTSDIR}/archivers/unzip \
                gcp:${PORTSDIR}/sysutils/coreutils \
                gpatch:${PORTSDIR}/devel/patch \

${SITE_PERL}/Archive/Zip.pm:${PORTSDIR}/archivers/p5-Archive-
Zip \
                bash:${PORTSDIR}/shells/bash \
                imake:${PORTSDIR}/devel/imake \
                ${LOCALBASE}/bin/gperf:${PORTSDIR}/devel/gperf
\
                ant:${PORTSDIR}/devel/apache-ant

.if !defined(WITH_GPC)
LIB_DEPENDS+=   art_lgpl_2:${PORTSDIR}/graphics/libart_lgpl
.endif

USE_BISON=      build
GNU_CONFIGURE=  yes
WRKSRC?=                ${WRKDIR}/${OOOTAG}
TCSH?=          /bin/tcsh
PKGMESSAGE=     ${WRKDIR}/pkg-message

.if defined(DISABLE_MAKE_JOBS)
MAKE_JOBS_NUMBER=       1
.endif

CONFIGURE_ARGS+=        --with-gnu-cp=${LOCALBASE}/bin/gcp
\
                        --with-gnu-
```

```
patch=${LOCALBASE}/bin/gpatch       \
                        --enable-crashdump=yes
\
                        --with-system-zlib
\
                        --with-system-stdlibs
\
                        --with-epm=internal
\
                        --enable-minimizer
\
                        --enable-presenter-console
\
                        --enable-pdfimport
\
                        --enable-wiki-publisher
\
                        --enable-report-builder


OOOEXTENSIONS+=         minimizer/sun-presentation-
minimizer.oxt
OOOEXTENSIONS+=         presenter/presenter-screen.oxt
OOOEXTENSIONS+=         pdfimport/pdfimport.oxt
OOOEXTENSIONS+=         swext/wiki-publisher.oxt
OOOEXTENSIONS+=         sun-report-builder.oxt

.if ${ARCH} == amd64
WITHOUT_MOZILLA=        yes
.endif
.if (${OSVERSION} <= 602102)
EXTRA_PATCHES+= ${FILESDIR}/rtld-workaround-i66667
.endif

ICONS=  ${WRKSRC}/sysui/desktop/icons

.include <${FILESDIR}/Makefile.knobs>

pre-everything::
# really tweak, extremely useful when you build all localized
language versions
# needed after when you build with ALL_LOCALIZED_LANGS.
```

```
.if defined(TWEAK_L10N)
        @${RM} -f ${WRKDIR}/.PLIST*
        @${RM} -f ${WRKDIR}/.install_done.*
        @${RM} -f ${WRKDIR}/.package_done.*
        @${RM} -f ${WRKDIR}/.extract_done.*
        @${RM} -f ${WRKDIR}/.patch_done.*
        @${RM} -f ${WRKDIR}/.configure_done.*
        @${RM} -f ${WRKDIR}/.build_done.*
        @${MKDIR} ${WRKDIR}
        @${TOUCH} ${EXTRACT_COOKIE}
        @${TOUCH} ${PATCH_COOKIE}
        @${TOUCH} ${CONFIGURE_COOKIE}
        @${TOUCH} ${BUILD_COOKIE}
.endif

#issue XXXXXX not raised yet...
CRLFFILES=curl/curl*patch neon/neon*patch
jfreereport/patches/*.patch

post-extract:
        for i in ${CRLFFILES};  do \
##g' $$i ; \    cd ${WRKSRC} ; ${REINPLACE_CMD} -e 's#
        done
        @${CP} ${DISTDIR}/${DIST_SUBDIR}/unowinreg.dll
${WRKSRC}/external/unowinreg/
.if defined(WITH_GPC)
        @cd ${WRKDIR} ; ${CAT}
${DISTDIR}/${DIST_SUBDIR}/gpc231.tar.Z | ${TAR} xfz -
        @${CP} ${WRKDIR}/gpc231/gpc.c ${WRKSRC}/external/gpc/
        @${CP} ${WRKDIR}/gpc231/gpc.h ${WRKSRC}/external/gpc/
.endif
.if !defined(WITHOUT_MOZILLA)
        @${CP} ${DISTDIR}/${DIST_SUBDIR}/${MOZILLA_SOURCE}
${WRKSRC}/moz/download
.endif

pre-configure:
#Workaround for gperf. #i85469#
        @${LN} -sf ${LOCALBASE}/bin/gperf
${WRKSRC}/solenv/bin/gperf
```

```
do-build:
        @cd ${WRKSRC} ; ./bootstrap
        @cd ${WRKSRC} ; ${SETENV} "LANG=C" "LC_ALL=C"
"TMP=${WRKSRC}" ${TCSH} -c "source ${FREEBSD_ENV_SET} ; cd
instsetoo_native ; build.pl --checkmodules ; build.pl
-P${MAKE_JOBS_NUMBER} --all --html --dontgraboutput --
-P${MAKE_JOBS_NUMBER}"

.if ${LOCALIZED_LANG} == "alllangs"
        @${MAKE} languagepack
.endif

do-install:
.if ${LOCALIZED_LANG} == "alllangs"
        @cd
${WRKSRC}/instsetoo_native/unxfbsd?.pro/OpenOffice/bsd/instal
l/en-US/freebsd-*/ ; ${LS} *.t?z > ${WRKDIR}/INSTALLFILES
        @cd
${WRKSRC}/instsetoo_native/unxfbsd?.pro/OpenOffice_languagepa
ck/bsd/install/ ; ${LS} */freebsd*/*.t?z >
${WRKDIR}/LANGPACKFILES
        @${RM} -Rf ${WRKDIR}/tmp
        @${MKDIR} ${WRKDIR}/tmp
        @for i in `${CAT} ${WRKDIR}/INSTALLFILES`; do \
                ${ECHO_CMD} "extracting $$i" ; \
                cd ${WRKDIR}/tmp ; ${TAR} xfz
${WRKSRC}/instsetoo_native/unxfbsd?.pro/OpenOffice/bsd/instal
l/en-US/freebsd-*/$$i ; \
        done
        @for i in `${CAT} ${WRKDIR}/LANGPACKFILES`; do \
                ${ECHO_CMD} "extracting $$i" ; \
                cd ${WRKDIR}/tmp ; ${TAR} xfz
${WRKSRC}/instsetoo_native/unxfbsd?.pro/OpenOffice_languagepa
ck/bsd/install/$$i ;\
        done
        @${MKDIR} ${PREFIX}/${INSTALLATION_BASEDIR}
        @cd ${WRKDIR}/tmp/opt/ ; ${TAR} cf - -C . . | ${TAR}
xf - -C ${PREFIX}/${INSTALLATION_BASEDIR}
.else
        @cd
${WRKSRC}/instsetoo_native/unxfbsd?.pro/OpenOffice/bsd/instal
```

```
l/${LOCALIZED_LANG}/freebsd-*/ ; ${LS} *.t?z >
${WRKDIR}/INSTALLFILES
        @${RM} -Rf ${WRKDIR}/tmp
        @${MKDIR} ${WRKDIR}/tmp
        @for i in `${CAT} ${WRKDIR}/INSTALLFILES`; do \
                ${ECHO_CMD} "extracting $$i" ; \
                cd ${WRKDIR}/tmp ; ${TAR} xfz
${WRKSRC}/instsetoo_native/unxfbsd?.pro/OpenOffice/bsd/install
/${LOCALIZED_LANG}/freebsd-*/$$i ; \
        done
        @${MKDIR} ${PREFIX}/${INSTALLATION_BASEDIR}
        @cd ${WRKDIR}/tmp/opt/ ; ${TAR} cf - -C . . | ${TAR}
xf - -C ${PREFIX}/${INSTALLATION_BASEDIR}
.endif
        @${MKDIR}
${PREFIX}/${INSTALLATION_BASEDIR}/extensions/
        @cd ${WRKSRC}/solver/310/unxfbsd?.pro/bin/ ;
${INSTALL_DATA} ${OOOEXTENSIONS}
${PREFIX}/${INSTALLATION_BASEDIR}/extensions/

post-install:
        @${ECHO_MSG} "===>  Add wrapper scripts";
        @${CP} ${FILESDIR}/openoffice.org-wrapper ${WRKDIR}/
        @${REINPLACE_CMD} -e 's#%%PREFIX%%#${PREFIX}#g' \
                        -e 's#%%OOOTAG%%#${OOOTAG}#g' \
                        -e 's#%%OOODIR%%#${OOODIR}#g' \
                        -e
's#%%INSTALLATION_BASEDIR%%#${INSTALLATION_BASEDIR}#g' \
                        ${WRKDIR}/openoffice.org-wrapper
        @${INSTALL_SCRIPT} ${WRKDIR}/openoffice.org-wrapper \
                ${PREFIX}/bin/${EXECBASE}
        @${LN} -fs ${PREFIX}/bin/${EXECBASE}
${PREFIX}/bin/${EXECBASE}-sbase
        @${LN} -fs ${PREFIX}/bin/${EXECBASE}
${PREFIX}/bin/${EXECBASE}-scalc
        @${LN} -fs ${PREFIX}/bin/${EXECBASE}
${PREFIX}/bin/${EXECBASE}-sdraw
        @${LN} -fs ${PREFIX}/bin/${EXECBASE}
${PREFIX}/bin/${EXECBASE}-setofficelang
        @${LN} -fs ${PREFIX}/bin/${EXECBASE}
${PREFIX}/bin/${EXECBASE}-simpress
        @${LN} -fs ${PREFIX}/bin/${EXECBASE}
${PREFIX}/bin/${EXECBASE}-smath
        @${LN} -fs ${PREFIX}/bin/${EXECBASE}
${PREFIX}/bin/${EXECBASE}-spadmin
        @${LN} -fs ${PREFIX}/bin/${EXECBASE}
${PREFIX}/bin/${EXECBASE}-swriter
        @${ECHO_CMD} "" > ${TMPPLIST}
        @cd ${PREFIX} ; ${FIND} -s bin \( -type f -or -type l
\) -name "${EXECBASE}*" >> ${TMPPLIST}
        @cd ${PREFIX} ; ${FIND} -s ${INSTALLATION_BASEDIR} \(
-type f -or -type l \) >> ${TMPPLIST}
        @cd ${PREFIX} ; ${FIND} ${INSTALLATION_BASEDIR} -type
d | ${SORT} -r | \
                ${XARGS} -n 1 ${ECHO_CMD} @dirrm >>
${TMPPLIST}
        @for app in base calc draw impress math writer; do \
                ${REINPLACE_CMD} -e
"s/^Exec.*/Exec=${EXECBASE} -$${app} %U/" \
                        -e "s/^Icon.*/Icon=${EXECBASE}-
$${app}.png/" \

${PREFIX}/${INSTALLATION_BASEDIR}/${OOODIR}/share/xdg/$${app}
.desktop ; \
        done
        @${REINPLACE_CMD} -e "s/^Exec.*/Exec=${EXECBASE}-
spadmin %U/" \
                -e "s/^Icon.*/Icon=${EXECBASE}-
printeradmin.png/" \

${PREFIX}/${INSTALLATION_BASEDIR}/${OOODIR}/share/xdg/printer
admin.desktop
        @${REINPLACE_CMD} -e "s/^Exec.*/Exec=${EXECBASE}
-quickstart -nologo -nodefault/" \

${PREFIX}/${INSTALLATION_BASEDIR}/${OOODIR}/share/xdg/qstart.
desktop
        @${RM}
${PREFIX}/${INSTALLATION_BASEDIR}/${OOODIR}/share/xdg/*.deskt
op.bak
        @${RM} -f ${DESKTOPDIR}/${EXECBASE}
        @${MKDIR} ${DESKTOPDIR}
```

```
        @${LN} -sf
${PREFIX}/${INSTALLATION_BASEDIR}/${OOODIR}/share/xdg \
                        ${DESKTOPDIR}/${EXECBASE}
        @${ECHO_CMD} "share/applications/${EXECBASE}" >>
${TMPPLIST}
        @${ECHO_CMD} "@unexec ${RMDIR} %D/share/applications
2>/dev/null || ${TRUE}" >> ${TMPPLIST}
        @${PREFIX}/bin/update-desktop-database 2>/dev/null ||
${TRUE}
        @${ECHO_CMD} "@exec ${PREFIX}/bin/update-desktop-
database 2>/dev/null || ${TRUE}" >> ${TMPPLIST}
        @for dir in `ls ${ICONS}/hicolor | ${GREP} -v CVS`; do
\
                for app in base calc draw impress math
printeradmin writer; do \
                        if [ -r
${ICONS}/hicolor/$${dir}/apps/$${app}.png ]; then \
                                ${INSTALL_DATA}
${ICONS}/hicolor/$${dir}/apps/$${app}.png \

${PREFIX}/share/icons/hicolor/$${dir}/apps/${EXECBASE}-
$${app}.png ; \
                                ${ECHO_CMD}
"share/icons/hicolor/$${dir}/apps/${EXECBASE}-$${app}.png" >>
${TMPPLIST} ; \
                        fi \
                done ; \
                for file in `cd
${ICONS}/hicolor/$${dir}/mimetypes; ls *.png`; do \
                        ${INSTALL_DATA}
${ICONS}/hicolor/$${dir}/mimetypes/$${file} \

${PREFIX}/share/icons/hicolor/$${dir}/mimetypes/ ; \
                        ${ECHO_CMD}
"share/icons/hicolor/$${dir}/mimetypes/$${file}" >>
${TMPPLIST} ; \
                done ; \
        done
        @${PREFIX}/bin/gtk-update-icon-cache -q -f
${PREFIX}/share/icons/hicolor 2>/dev/null || ${TRUE}
        @${ECHO_CMD} "@unexec ${RM}
```

```
%D/share/icons/hicolor/icon-theme.cache 2>/dev/null ||
${TRUE}" >> ${TMPPLIST}
        @${ECHO_CMD} "@exec ${PREFIX}/bin/gtk-update-icon-
cache -q -f %D/share/icons/hicolor 2>/dev/null || ${TRUE}" >>
${TMPPLIST}
        @${ECHO_CMD} "@unexec ${PREFIX}/bin/gtk-update-icon-
cache -q -f %D/share/icons/hicolor 2>/dev/null || ${TRUE}" >>
${TMPPLIST}
        @${CP} ${FILESDIR}/pkg-message.in ${PKGMESSAGE}
        @${REINPLACE_CMD} -e 's#%%PREFIX%%#${PREFIX}#g' \
                          -e 's#%%EXECBASE%%#${EXECBASE}#g' \
                          -e 's#%%OOOTAG%%#${OOOTAG}#g' \
                          -e 's#%%OOODIR%%#${OOODIR}#g' \
                          ${PKGMESSAGE}
        @${ECHO_CMD}
        @${CAT} ${PKGMESSAGE}
        @${ECHO_CMD}

.include <${FILESDIR}/Makefile.others>
.include <bsd.port.post.mk>
droideka#
```

Openoffice.org-3 olarak tanımlı olan port Openoffice.org-3.1.1 sürümünün kurulumu için gereken yönergeleri barındırmaktadır. Dosyanın başında yer alan porta ait olan bilgiler bulunmaktadır. Porttan sorumlu olan geliştirici, kullanılacak olan kaynak kod ve indirileceği site, yazılımın yapılandırmasına ilişkin bilgiler bulunmaktadır. Tüm makefile yapısı üzerinde durmak olanaklı olmadığı için incelenmesini okuyuculara bırakıyoruz.

Port Kurulumu Kaldırmak ve Yeniden Kurmak

Port kullanımı aynı paket kullanımı ile eşdeğerdir. Kurduğunuz port ile bir paket arasında bir fark bulunmaz. Kurulu olan bir portu pkg_delete ile kaldırabilirsiniz. Port hakkında pkg_info ile bilgi edinebilirsiniz. Port kurulumları ile paket kurulumları /var/db/pkg altında kayıt edilir. Dolayısıyla port ile paket kullanımı arasında yöntem olarak bir fark bulunmaz.

Bir portu kaldırmak için pkg_delete kullanmak durumunda değiliz. Kaldırmak istediğiniz portun bulunduğu dizine geçip make deinstall komutunu vermeniz yeterli olur. Bu komut söz konusu portu sistemden kaldırır. Bu işlemin ardından derlenmiş dosyalar ile kaynak kod work dizininden silinmaz ve orada kalırlar. make reinstall komutu da kaldırdığınız portu yeniden sisteme kuracaktır.

Port Derleme/Kurulum Durumu

Port ile kurulan yazılımın var/db/pkg altında kaydı tutulmakla birlikte port ağacı üzerinde kurulu olan bir porta ilişkin bilgi work dizini altında yer alan gizli dosyalarda saklanır.

```
droideka# cd /usr/ports/editors/openoffice.org-3/work/
droideka# ls -l
total 89138
-rw-r--r--    1 root  wheel    417815  3 Eyl 05:35
.PLIST.flattened
-rw-r--r--    1 root  wheel    409136  3 Eyl 05:35
.PLIST.mktmp
-rw-r--r--    1 root  wheel  90312881  3 Eyl 05:35
.PLIST.objdump
-rw-r--r--    1 root  wheel         0  3 Eyl 05:35
.PLIST.setuid
-rw-r--r--    1 root  wheel         0  3 Eyl 05:35
.PLIST.writable
-rw-r--r--    1 root  wheel         0  3 Eyl 05:34
.build_done.openoffice.org._usr_local
-rw-r--r--    1 root  wheel         0  3 Eyl 02:18
.configure_done.openoffice.org._usr_local
-rw-r--r--    1 root  wheel         0  3 Eyl 02:18
.extract_done.openoffice.org._usr_local
-rw-r--r--    1 root  wheel         0  3 Eyl 05:35
.install_done.openoffice.org._usr_local
-rw-r--r--    1 root  wheel         0  3 Eyl 05:37
.package_done.openoffice.org._usr_local
-rw-r--r--    1 root  wheel         0  3 Eyl 02:18
.patch_done.openoffice.org._usr_local
-rw-r--r--    1 root  wheel      2384  3 Eyl 05:34
INSTALLFILES
drwxr-xr-x  196 root  wheel      4096  3 Eyl 05:34 OOO310_m19
-rw-r--r--    1 root  wheel       389  3 Eyl 05:35
openoffice.org-wrapper
-rw-r--r--    1 root  wheel       388  3 Eyl 05:35
openoffice.org-wrapper.bak
-rw-r--r--    1 root  wheel      1307  3 Eyl 05:35 pkg-
message
-rw-r--r--    1 root  wheel      1209  3 Eyl 05:35 pkg-
message.bak
drwxr-xr-x    3 root  wheel       512  3 Eyl 05:34 tmp
droideka#
```

Dizinin içeriğini ls ile görüntülediğinizde adının önünde nokta olan dosyalar görüntülenmeyecektir. Port ağacı bu dosyaları portun kurulum aşamalarını izlemek için kullanır. .install_done.openoffice.org._usr_local dosyası kurulum sürecinin tamamlandığını gösterir. Bazı durumlarda bir portun birden çok kurulum ve kaldırma işleminin ardından yeniden kurulumu gerçekleşmeyebilir. Bu durumda kurulumum tamalandığını belirten dosyanın silinmesi sorunun çözülmesi için yeterlidir.

Port Temizliği

Port kaynak kodları indirip işlem yaptığı için zamanla diski doldurmaya başlar. Bu nedenle port üzerinde işi biten dosyaları kaldırmak disk alanını geri kazanmak için uygun olacaktır.

Port üzerinden kurulum tamamlandığında kaynak kod ve diğer dosyalara gereksinim kalmayacağı için bu dosyaların güvenli bir şekilde kaldırılması uygun olur. Bunun için make install ardından make clean komutunu vererek artık dosyaları temizleyebilirsiniz. Bu iki komutu tümleşik olarak kullanabilirsiniz.

```
# make install clean
```

Bunun dışında distfiles dizininde olan kaynak kod dosyalarını da

temizlemek için make distclean komutu kullanılabilir. Bu şekilde tüm portlar ve distfiles için ayrı ayrı temizlik yapmak yerine birtek komut ile tüm /usr/ports dizininin ve alt dizinlerindeki artık dosyaları temizleyebiliriz.

```
# make clean -DNOCLEANDEPENDS
```

Port Kullanarak Paket Hazırlamak

Eğer birden fazla bilgisayarda FreeBSD çalıştırıyorsanız her bir makinada port kurulumu ile uğraşmak durumunda kalmaya gerek kalmadan kurulum yaptığınız portların paket oluşturmasını sağlayabilirsiniz. Böylece tüm bilgisayarlarda yapılandırması aynı olan yazılımları kullanabilirsiniz. Bir paket oluşturmak port kurulumu ile aynıdır. fark ise sonuçta kullanılan komuttur.

```
# make package
```

Yukarıdaki komut bulunduğu portun kurulumunu gerçekleştirir ve söz konusu port dizininde tbs uzantılı freeBSD paketini oluşturur. Diğer bilgisayarlarda pkg_add komutunu kullanarak kurulumunu yapabilirsiniz. Öte yandan eğer /usr/ports/packages dizinini oluşturursanız, her hazırlanan paket bu dizinde toplanacaktır. Bu dizin altında her paket ilgili dizin altında toplanır. Örneğin KDE paketleri KDE3 veya KDE4 sürümüne göre ayrı ayrı dizinlerde tutulacaktır.

Port ve Paket Güvenliği

Port ve paketlerin sayısının yirmibinden fazla olduğu bir sistemde kurulan her yazılım için yayınlanan güvenlik yamaları vb. için izlemek kolay değildir. Yazılım geliştiricileri yeni sürümler ve hatta yamalar yayınlar. Bu yani sürümler freeBSD port ağacında yerini alır. Portun yeni sürümünün güvenlik yaması veya yazılımın yeni sürümü olup olmadığını hatta o sırada yaması çıkmadığı halde güvenlik açığı bulunan bir portun belirlenmesi gereklidir. Port ağacını oluşturan yazılımları güvenlik açısından kontrol eden bir araç yine port ağacında bulunan portaudit uygulamasıdır.

Portaudit port ağacı üzerinde port yönetimi araçlarının bulunduğu ports-mgmt/portaudit dizininde bulunur. Eğer FreeBSD geliştiricileri bir güvenlik açığı bulurlarsa bunu portaudit veritabanına kayıt ederler. Sistem yöneticisine ise portaudit kurup portların güvenliğini kontrol etmesi ve gereken önlemleri alması kalır.

Portaudit kurulum CD/DVD setlerinde yer alır ve paket seçimi aşamasında seçilerek sisteme kurulabilir. Eğer kurulu değilse port üzerinden kurulabilir.

```
# cd /usr/ports/ports-mgmt/portaudit
# make all install clean
```

portaudit kurulduğunda periodic girdisini oluşturulup otomatik olarak çalışması sağlanabilir. Böylelikle belirlenen zamanda sunucudan güvenlik bilgilerini edinip kurulu olan tüm portları kontrol eder. Güvenlik açığı bulunan portlar raporlanır. rapor doğrudan /var/mail/root dosyasına kayıt edilir veya sistem yöneticisinin yapılandırmasına bağlı olarak bir e-posta adresine yönlendirilebilir. Portaudit ayrıca istenirse elle çalıştırılarak planlanan zamandan önce kontrol edilmesi için kullanılabilir.

portaudit sadece belirli zamanlarda veya elle çalıştılarak da kullanılmaz. Port kurulumlarında en son indirilen güvenlik kaydı dikkate alınarak kurulumlar kontrol edilir. Eğer bir porta ilişkin güvenlik açığı bulunmuş ise ve bunun için bir yama bulunmuyorsa, söz konusu port kurulmaz. Kurulum işlemi hata döndürerek sonlanır.

Port Ağacını Güncellemek

Tüm işletim sistemlerinin geliştirilmesi ve güncellenmesi gibi port ağacı da zaman içinde yeni eklenen yazılımlar ve kaldırılanlar ile

sürekli olarak güncellenir. BSD sisteminizi güncellediğiniz gibi aynı zamanda port ağacının da güncellenmesi gereklidir. port ağacının güncellenmesi iki boyutlu bir işlemdir. Birincisi port ağacının güncel port ağacına terfi edilmesi ikincisi de kurulu olan yazılımların güncel sürümlerine terfi edilmesidir.

Burada başlıktan anlaşıldığı gibi port ağacınızı güncelleme işleminden söz edeğiz. Yazının önceki bölümlerinde kısaca portsnap'a değindik. Şimdi portsnap'ın işleyişi üzerinde biraz duralım. portsnap freeBSD port ağacına ait olan düzenli aralıklar ile oluşturulan portağacı imajlarını/yamalarını sisteminize kurar. freeBSD merkezi port ağacı sunucusu yaklaşık olarak birer saat arayla port ağacındaki güncellemeleri kontrol ederek var olan değişiklikleri yeni port ağacı imajını oluşturmak için kullanır. Bu değişiklikler bir yama olarak hazırlanır ve merkezi portsnap sunucusunda barındırılır. Portsnap manuel olarak kullandığınızda merkezi portsnap sunucu-sundan bu yamaları/imajı indirip sisteme kurulumunu yapar. Port ağacını güncel tutmak için portsnap'tan başka cvsup'da kullanılabilir. Bu iki araçtan birisini kullanın ama her ikisini birden kullanmayın.

Portsnap'ı Yapılandırmak

Portsnap, /etc/portsnap.conf dosyası ile yapılandırılır. Bu dosyayı düzenlemeye pek gereksinim duyulmamakla birlikte bazı ayarları tercihlerinize göre düzenleyebilirsiniz. Yapılandırma dosyası aşa-ğıdaki gibidir:

```
droideka# cd /etc
droideka# cat portsnap.conf
# $FreeBSD: src/etc/portsnap.conf,v 1.3.2.1.4.1 2009/04/15
03:14:26 kensmith Exp $

# Default directory where compressed snapshots are stored.
# WORKDIR=/var/db/portsnap
# Default location of the ports tree (target for "update" and
"extract").
# PORTSDIR=/usr/ports
# Server or server pool from which to fetch updates.  You can
change
# this to point at a specific server if you want, but in most
cases
# using a "nearby" server won't provide a measurable
improvement in
# performance.
SERVERNAME=portsnap.FreeBSD.org
# Trusted keyprint.  Changing this is a Bad Idea unless
you've received
# a PGP-signed email from <security-officer@FreeBSD.org>
telling you to
# change it and explaining why.
KEYPRINT=9b5feee6d69f170e3dd0a2c8e469ddbd64f13f978f2f3aede40c
98633216c330
# Example of ignoring parts of the ports tree.  If you know
that you
# absolutely will not need certain parts of the tree, this
will save
# some bandwidth and disk space.  See the manual page for
more details.
# WARNING: Working with an incomplete ports tree is not
supported and
# can cause problems due to missing dependencies.  If you
have REFUSE
# directives and experience problems, remove them and update
your tree
# before asking for help on the mailing lists.
#
# List of INDEX files to build and the DESCRIBE file to use
for each
INDEX INDEX-5 DESCRIBE.5
INDEX INDEX-6 DESCRIBE.6
INDEX INDEX-7 DESCRIBE.7
droideka#
```

Bu dosyada yer alan bazı değişkenlere göz atalım:

```
SERVERNAME=portsnap.freebsd.org
```

Portsnap sunucuları aslında birçok sunucudan oluşur. Bu gün için merkezi bir sunucu dışında yansı bulunmadığı için zaman içerisinde kullanılabilecek olan yansılardan birisini tanımlamak için kullanılabilir.

```
KEYPRINT=9b5...
```

portsnap sunucusundaki imajların bütünlüğü ile güvenli olduklarının kontrolünü sağlayan değerdir. Bu değeri değiştirmeyin.

```
REFUSE arabic korean
```

Bu değişken portsnap'a atanılanan portların güncellenmeyeceğini belirtir. Bu değişkeni kullanırken dikkatli olmak gerekiyor. Port ağacının bir bütün olduğunu ve bazı portların doğru olarak kurulması ve çalışması için diğer portlara bağımlı olduğunu her zaman göz önünde bulundurmak gereklidir.

Portsnap Kullanımı

Portsnap elle sistem yöneticisi tarafından çalıştırılabilir. Yukarıdaki bölümde elle kullanımını gördük. Port ağacının elle güncellenmesi yerine bunun cron ile otomatik olarak yapılmasını sağlayabilirsiniz. Aşağıdaki cron girdisi sabah saat 09:00 'da bilgisayardaki port ağacının portsnap ile güncellenmesini sağlamaktadır.

```
0   5   *    *    *    /usr/sbin/portsnap cron fetch update
```

prosnap cron update komutu sabah saat 09:00 ile 10:00 arasında rastgele bir zamanda portsnap sunucusundan port ağacının güncel sürümünü indirecektir. Bu aşamadan sonra sizin yapmanız gereken, gerekiyorsa portları terfi etmek olacaktır.

Port Güncellemesi

Port ağacını güncellediğinizde ve port ağacının güncel sürümünü kullanarak yazılım kurduğunuzda bu yazılımların güncel sürümlerini kullanıyor olacaksınız. Önceden kurduğunuz yazılımlar ise güncel olabileceği gibi olmayabilir de. Bu ayrımı belirlemek ve sorunsuzca yazılımları güncel sürümlerine terfi etmek için FreeBSD'de port-upgrade veya portmaster ile bu işlemler gerçekleştirilir.

portupgrade, FreeBSD'nin ilk port yönetim araçlarındandır ve RUBY ile yazılmıştır. Port ağacındaki bilgileri kullanarak portların durumunu karşılaştırarak güncelleme işlemlerini gerçekleştirir.

portmaster, bir kabuk programıdır ve ayrıca bir veritabanına gerek duymadan birçok yazılım yönetimi işlevlerini yerine getirir. Dikkat edilmesi gereken nokta ise bu aracın bazı uç durumlarda yetersiz olabileceğidir. Portmaster veya portupgrade arasında bir karşılaştırma yapılması birinin diğerine üstün olduğu sonucunu vermeyecektir ama bazı özellikleri açısından portupgrade sonuca daha kolay götürecektir.

portmaster sistem kurulumu yapılırken kurulmaz. Sistem yöneticisi tarafından sonradan kurulması gerekir. /usr/ports/ports-mgmt/portmaster portu üzerinden kurulabilir. Kurulumun ardından portlarınızın kontrol edilmesini sağlamak için aşağıdaki komutu vermeniz gerekir:

```
# portmaster -L
```

Bu komutu kullandığınızda daha önceden kurulmuş olan ama sonradan kaldırdığınız portun kurulumu sırasında kurulan bağımlılıklar da kontrol edilir. Bu bağımlılıklar artık portlar olarak kalır. Başka bir port tarafından gereksinim duyulmuyorsa kaldırabilirsiniz. Bunu kontrol etmenin en kolay yolu portmaster kullanımıdır. Bu komut tüm portları kontrol edeceği için çıktısı da uzun olacaktır. Bu nedenle komutun çıktısını okumak üzere bir dosyaya yönlendirmeniz yerinde olur.

```
# portmaster -L > portmaster_L_sonuc.txt
```

portmaster port ağacındaki benzeşimi kullanarak kök portlar ile dallar olarak tanımlanan kök portlardan uzanan diğer bağımlı portları listeler. Böylece bir porta gereksinim duyulup duyulmadığını size tanımlar. Örneğin sudo, zip. ve bash gibi portlar root/kök port olarak tanımlanır.

Bunların dığında ağacın gövdesine benzetilen trunk port yani gövde port bulunur. Bu portların bir bağımlılığı bulunmaz ama diğer portlar tarafından gereksinim duyulur. Örneğin paylaşımlı kütüphaneler gibi. Buna en güzel örnek PERL olacaktır. PERL diğer portlara bağımlı değildir ama diğer birçok portun kurulumu için gereklidir.

Gövdenin ucunda yukarıda adı geçen dal port/brach ports bulunur. Bu portlar hem diğer portlara gereksinim duyarlar hem de diğer portlar bu portlara. Örneğin X11, JAVA vb. bu dal portlar olarak tanımlanır.

Dalların ucunda ise yapraklar bulunur. Bir diğer deyişle leaf ports. Bu portlar diğer portlara gereksinim duyar ama diğer portların bu portlara gereksinimi veya bağımlılığı sözkonusu değildir. Örneğin metin editörleri, ofis paketleri vb. gibi.

Portmaster ile portları güncellerken bir dal veya gövde portun değişmesi yaprak portlara dek birçok portu etkileyecektir. Örneğin grafik kütüphanelerindeki bir terfi eski sürüme bağımlı olan veya eski sürümdeki dosyaları kullanan birçok yazılımın çalışmasını önleyecektir. Bu durumda portmaster sözkonusu olan diğer portları da güncelleyecektir.

Eski Port Kurulumlarını Kaldırmak

Bazı durumlarda önceden kurulmuş olan portlara bugün gerek duyulmayabilir. Bu portları kaldırabilirsiniz. Ancak bu portlara ait olan bağımlılıklar sistemde kalacaktır. Port ağacını güncellediğinizde bu gerek duyulmayan portların da güncellenmesi sözkonusu olacaktır. Bu noktada portmaster'in kullandığı kök, gövde, dal ve yaprak yaklaşımı işleri son derece kolaylaştıracaktır.

Eğer güncellenecek olan yazılımın bulunduğu portu bilemiyorsanız wehereis <isim> ile portunu öğrenebilir ve sözkonusu porta gözatıp bir karar verebilirsiniz. Eğer sözkonusu port gerekli ise sistemde kalabilir ama gereksiz olduğunu düşünüyorsanız ve hatta daha önemlisi emin iseniz sistemden kaldırabilirsiniz. Bu size hem zaman hem de sistem kaynaklarından tasarruf sağlayacaktır.

Openoffice.org-3.1.1 örneğini alacak olursak, openoffice.org-3 portuna gerek kalmadığında pkg_delete ile sadece openoffice.org-3 portunu kaldırmış oluruz ama bağımlılıkları sistemde kalacaktır. Bu durumda tek bir portu kaldırıp geride artık bırakmak yerine portmaster kullanarak gerek duyulmayan diğer portların da sistemden kaldırılmasını sağlayabiliriz.

```
# portmaster openoffice.org-3.1.1
```

Hem OpenOffice.org 3.1.1'i hem de gereksinim duyulmayan diğer portları da sistemden kaldıracaktır.

Portmaster ile Port Güncelleme

portmaster tüm portları inceleyerek ayrıntılı bir rapor sunacaktır. Bu raporun incelenmesi ile güncellenecek olan portlara ilişkin kesin bir fikir edinebilirsiniz. Bir portu güncellemek için portmaster'a özel bir değişken tanımlamak gerekmez. Doğrudan güncellenecek olan portun adını vermek yeterlidir. Örneğin kendi sistemimde kurulu olan etcdf paketini portmaster ile terfi etmek için

```
# portmaster netcdf
```

komutunu vermem gereklidir. Portmaster ile port güncellemesi yapılırken hem bağımlılıkları hem de asıl porta ait olan kaynak kodları

indirecektir. Derleme işlemine başlanmadan önce sistem yöneticisinin yapması gereken düzenlemeleri de soracaktır. Bu bazı durumlarda sözkonusu porta bağlı olan diğer portların da yapılandırmasının en başından yapılmasını sağlayacaktır. Böylece gereken yapılandırma seçenekleri başından belirlenmiş olduğu için güncelleme işlemi sırasında bilgisayar başında oturmanıza gerek kalmayacaktır.

Her zaman için bir portun güncellenmesi sırasında işlemlerin başarısılıkla sonuçlanması sözkonusu olabilir. Bir portun güncellenebilmesi için öncelikle eski sürümün sistemden kaldırılması ve ardından yeni sürümün kurulması gereklidir. portmaster önce eski sürümün bir yedeğini alır ve ardından yeni sürümü kurar. Güncelleme işleminin başarılı olmasının ardından eski sürümün yedeğini siler. Ancak yedeğin sistemde tutulmasını sağlamak için -e parametresinin portmaster ile kullanılması gereklidir. Olası bir sorun durumunda /usr/ports/packages/All dizininden eski sürümü kurabilirsiniz. Eğer birden çok eski paketi güncelliyorsanız bu durumda bağımlılıklar ve ilgili tüm portlara ait olan yedeklenen paketler saklanır. Böylelikle yarısı güncellenmiş veya güncellenmemiş karma bir sistem ile karşı karşıya kalmazsınız.

Portu Değiştirmek

freeBSD port ağacında bazen bir portun adı ya geliştiricilerin tercihleri veya isimlendirmede kolaylık sağlamak amacı ile değiştirilir. Örneğin ghostscript-gpl olarak geçen port daha sonra ghostscript8 olarak anıldı. Bu durumda port ağacını güncellediğinizde eski portun ortadan kaldırılıp yerine yeni portun konulması gerekecektir. Bu durumda düzenleme portmaster ile

```
# portmaster -o print/ghostscript-gpl print/ghostscript8
```

komutu ile sözkonusu değişiklik sistem genelinde uygulanabilir.

Port Ağacı Üzerinde Temizlik Yapmak

Port ağacı üzerinde kaynak kodların nasıl tutulduğuna yukarıda yer vermiştik. Kaynak kodların temzlenmesine gereksinim duyuyorsanız bu durumda genel bir temzlik yapabilirsiniz veya sadece artık gereksinim duyulmayan portlara ait olan kaynak kodları kaldırıp gerisini bırakabilirsiniz. Gereksinim duyulmayan kaynak kodların port ağacı genelinde temizlenmesi için

```
# portmaster --clean-distfiles
```

Komutunu kullanabilirsiniz. Portmaster bu konutu aldığında tüm gereksinim duyulmayan kaynak kodların silinmesi için onayınızı alacaktır. Her soruya yanıt vermek yerine otomatik olarak silinmesini sağlamak için aşağıdaki komutu kullanın:

```
# portmaster --clean-distfiles-all
```

Portupgrade, portinstall, portversion ve portsclean

Portmaster gibi diğer port yönetim araçları benzer seçenekleri kullanarak port yönetimini sağlamaktadır. portları güncellemek için portupgrade kullanabilirsiniz. portupgrade veri tabanını kullanarak bir portu gerek tekil gerek bağımlılıkları ile birlikte güncelleyebilir. Ancak portmaster'in aksine olarak ağaç benzeşimini kullanmaz. Bu nedenle portupgrade kullanırken UPDATING dosyasının içeriğine başvurmak veya gerekiyorsa bağımlılıklardan başlanarak güncellemeler yapılmalıdır. Güncellenecek olan portlar üzerinde portupgrade işlem yaparken bağımlılıkları dikkate alır.

Güncellenecek olan portların bir listesini almak portversion ile daha kolaydır. portversion -l "<" güncellenecek olan portların bir listesini döndürür.

```
droideka# portversion -l "<"
```

```
cdf3                            <
gcc                             <
libxml2                         <
netcdf                          <
png                             <
py26-libxml2                    <
tiff                            <
droideka#
```

Bu portları güncellemek için portupgrade kullanılır. Bir portun güncellenmesini yapmak için portupgrade -vr kullanabiliriz.

```
droideka# portupgrade -vr gcc
```

Bu komut gcc portunu güncellenmesi gereken bağımlılıkları ile birlikte güncelleyecektir. Ancak öte yandan başka portlar bu portun güncellenmesine bağlı olarak güncellenmek durumunda kalınabilir. portmaster'in yapmış olduğu işlem burada portupgrade -vRr seçeneği ile yapılabilir. R seçeneği portupgrade varsa diğer portların da bu güncellemeden sonra yeniden inşa edilmesini sağlar.

Portupgrade portların güncellenmesini tek tek veya toplu olarak yapabilir. Güvenli tarafta kalmak için öncelikle UPDATING dosyasına göz atıp ardından portları teker teker güncellemek gereklidir. Ancak portların güncellenmesi ile diğer portların da yeniden derlenmesi veya güncellenmesi gerekli değilse bu durumda

```
# portupgrade -varRp
```

komutu tüm portları güncelleyecektir.

Portinstall adından da anlaşılacağı üzere make install ile aynı işleve sahiptir. make ile aynı işleve sahip olmakla birlikte make ile kullanılan seçenekleri kullanmak sözkonusu değildir. Makefile ile tanımlanan yönergelere sadık kalarak standart bir kurulum sürecini gerçekleştirir, bir portu bağımlılıkları ile birlikte kurar. Kurulum işlemi bir portun kurulması için gereken tüm bağımlı portların kurulmasının ardından asıl portun kurulması ile sona erer. Olası sorun ise kurulum sırasında bağımlılıklardan birisinin kurulumunun başarısızlıkla sonuçlanıp kurulumun yarıda kalmasıdır. Portinstall kurulan veya kurulmayan tüm portların bir listesini sunacağı için eğer kurulumda sorun ile karşılaşıyorsanız bu durumda diğer kurulan portları kaldırarak sistemi temizleyebilirsiniz.

Portsclean, kaynak kodları veya sadece work dizinini ve ayrıca kullanılmayan ya da birden fazla kurulmış olan paylaşımlı kütüphaneleri temizlemek için kullanılan bir araçtır. Kurulum yaptığınız portlara ait olan kaynak kodların sadece work dizinindekileri silebilirsiniz.

```
# portsclean -C
```

distfiles altında yer alan kaynak kodları da temizleyebiliriz.

```
# portsclean -D
```

Paylaşımlı kütüphanelerden kullanılmayanlar var ise bunları da kaldırmak olanaklıdır. Özellikle /var/db/pkg dizininde yer alan kayıtların düzenlenmesinde çok işe yarayacaktır.

```
# portsclean -L
```

Ports konusunda çok şey yazılabilir. Ancak zaman ve yer kısıtı dolayısıyla bazı temel özelliklerine yer verilen ports, en iyi ilgili araçların kılavuz sayfalarını okuyarak ve uygulayarak öğrenilebilir.

Gelecek ay pkgsrc ile devam edeceğiz.

---

[1]http://en.wikipedia.org/wiki/Jordan_Hubbard

pkgsrc - NetBSD paket yönetim aracı

Gökşin Akdeniz
goksin@enixma.org

PKGSRC -NetBSD Paket Yönetim Aracı

BSD sistemlerinde bir tek paket yönetim aracını kullanmak durumunda değilsiniz. FreeBSD'nin PORTS'u yanında NetBSD ekibinin geliştirdiği pkgsrc'da yaygın olarak kullanılmaktadır. pkgsrc, ports'da olduğu gibi kaynak koddan yapılacak olan kurulumlar yanında hazır derlenemiş ve paketlenmiş olan yazılımları da desteklemektedir. Ports ve pkgsrc ayrıca kaynak kodlardan yararlanarak birden çok bilgisayarda kullanılmak üzere paket hazırlanmakta da kullanılabilir.

pkgsrc, NetBSD geliştiricileri tarafından geliştirildiği için sadece özelde NetBSD ve diğer BSD işletim sistemi ailesinin üyeleri yanında birçok farklı platformda da kullanılabilmektedir. Aşağıdaki tabloda pkgsrc kullanabileceğiniz platformalar ile bu platformlara aktarıldığı tarihleri görebilirsiniz. pkgsrc birçok platforma aktarılmasının ve kullanılmasının temelinde geliştiricilerin pkgsrc için belirledikleri ilkeler bulunur.

"Sadece doğru yazılmış kod çalışır. - Bir diğer deyişle bir paket hatalar barındırıyorsa, öncelike bunları belirleyip gidermek, derleyip çalışmasını ummaktan daha iyidir. Bu hatarları belirlemek için çeşitli araçlar kullanılır. Örneğin pkglint ve derleme ile sonrasında gerçekleştirilen kontroller gibi."

"Eğer çalışıyorsa her yerde çalışmalıdır. - NetBSD nasıl birçok platforma aktarılabiliyorsa, pkgsrc'da birçok platdforma aktarılabilmelidir. Dolayısıyla tüm paketlerin doğru çalıştığından emin olmalıyız."

pksrc'nin Aktırıldığı Platformlar

Platform	Aktarıldığı Tarih
NetBSD	Ağustos 1997
Solaris	Mart 1999
Linux	Ocak 1999
Darwin (Mac OS X)	Ekim 2001
FreeBSD	Kasım 2002
OpenBSD	Kasım 2002
IRIX	Aralık 2002
BSD/OS	Aralık 2003
AIX	Aralık 2003

```
Interix (Microsoft Windows Services for Unix)   Mart 2004
DragonFlyBSD                            Ekim 2004
OSF/1                                   Kasım 2004
HP-UX                                   Nisan 2007
-------------------------------------------------------------
```

pkgsrc, listeden gördüğünüz gibi birçok farklı platforma aktarılmış olsa da pkgsrc işleyiş ve yapı açısından netBSD'de olduğu gibi çalışacaktır. Geçekleştireceğiniz kurulum işlemi sırasında derlenecek olan kaynak kod ile yamalar asıl kaynaktan veya öntanımlı yansılardan indirilip kontrol edildikten sonra derlenecektir. Derlenmiş dosyalar, kılavuz sayfaları, belgelendirme ile yapılandırma dosyaları her platformda aynı dizine kopyalanır. Kurduğunuz bir yazılımın güncellenmesi işlemi sırasında güncellenmesi gereken diğer bağımlılıklar varsa pkgsrc bu durumu da dikkate alarak gereken güncellemeleri gerçekleştirir. pkgsrc birçok açıdan ports ile aynı şekilde çalışır. Farklı olduğu nokta ise pkgsrc her bir yazılım için merkezi bir yapılandırma dosyası kullanır. pkgsrc ile ports'un ayrıldığı bir diğer önemli nokta da ports ve paket terimlerinin anlamlarıdır. NetBSD ekibi için "ports" terimi netBSD'nin aktarıldığı diğer platformları ifade eder. freeBSD geliştiricileri için ports ise önceki yazıda anlatıldığı gibi port ağacını ifade eder. freeBSD ile NetBSD geliştiricileri pkgsrc ile kurduğunuz bir yazılımı paket olarak adlandırırken freeBSD geliştiricileri de yanı terimi aynı anlam ile kullanır. NetBSD geliştiricileri paket terimini sık kullanmaz ve paket yerine pkgsrc demeyi tercih eder. pkgsrc kullanarak derlenmiş ve tgz olarak tek bir dosya olarak sunulan yazılımlar da paket olarak anılır. Bu dosyaları ilgili BSD sistemin ftp sunucularında bulabilirsiniz. Eğer kendiniz pkgsrc kullanarak tgz dosyaları oluşturursanız bu paketler de /usr/pkgsrc/packages dizini altında yer alacaktır. Paket terimi pkgsrc söz konusu olduğunda bu dosyalara atıfta bulunmaktadır.

pkgsrc Kurulumu ve Güncellemesi

pkgsrc, NetBSD ftp sunucularından edinilebilir. Eğer ilk defa bir sisteme pkgsrc kurmak ve kullanmak istiyorsanız, öncelikle hangi dizin altına kuracağınıza karar vermeniz gerekir. netBSD üzerinde kullanırken yazılım yönetimini root kullanıcısının yetkisinde tutuyoruz. Bu nedenle root olarak pkgsrc ile çalışırken /usr/pkgsrc dizinini kullanıyoruz. Ancak pkgsrc, NetBSD dışında başka bir platformda kullanmak isterseniz bu durumda pkgsrc'yi istediğiniz bir dizine kurabilirsniz. Kuracağınız dizinin adında $SHELL tarafından yorumlanacak olan özel karakterlerin örneğin boşluk ve tırnak işaretleri gibi olmamasına dikkate etmeniz gerekiyor. Yazıda pkgsrc kurulumu ve güncellenmesi işlemlerini NetBSD 5.0.1 üzerinde uygulamalı olarak göstereceğiz.

pkgsrc ilk Kurulumu

pkgsrc geliştirme ekibi iki ayrı sürüm sunarlar. Bunlar "current" ve "stable" olarak adlandırılırlar. "current" geliştirme sürümüdür. Her gün var olan cvs ağacı tar.gz ve tar.bz2 dosyası olarak hazırlanır ve ftp sunucusuna aktarılır. Current ağacındaki geliştirme çalışmalarının tamamlanması ile kararlı duruma gelmiş olan cvs ağacı "stable" olarak duyurulur.

pkgsrc'un stable/kararlı dizininde kararlı sürümü yer alır. Bu dizinde göreceğiniz tar.gz ve tar.bz2 dosayaları arasından seçim yapmanız gerekecektir. Kararlı olan pkgsrc sürümleri hazırlandığı yıl ile dönemin birleşiminden oluşturulan bir isme sahiptir. Bu yazının hazırlandığı sırada güncel sürüm 2009Q2 iken ertesi gün 2009Q3 yayınlandı. Bu isimlendirmede 2009 yılı Q3 ise yılın üçüncü çeyreğinde hazırlanmış olan pkgsrc arğacını belirtmektedir. Bu isimlendirme ile en son hazırlanmış olan pkgsrc tar.gz veta tar.bz2 dosyasını indirmeniz uygun olur. Zira son sürüm güvenlik yamalarını ve güncel sürümleri barındırmaktadır.

İlk defa pkgsrc kuracaksanız yukarıda anlatıldığı gibi güncel sürümü bilgisayarınıza indirin. Önceki sürüm olan 2009Q2 sürümünü

indirmiştik. Bütünlük kontrolünü yaptıktan sonra /usr/pkgsrc altına açıyoruz.

```
netbsd501 # tar -jxvf pkgsrc-2009Q2.tzr.bz2 -C /usr
```

Bu işlem ile /usr/pkgsrc dizini ile diğer alt dizinler oluşturulacaktır. Bir sonraki pkgsrc sürümü olan 2009Q3 yine aynı yöntemle kurup kullanabilirsiniz. Ancak yeni dosyayı indirip açmadan önce dist files ile packages dizinindeki dosyalarını yedeklemeniz ve ardından var olan /usr/pkgsrc ile alt dizinleri silmeniz gereklidir. 2009Q2 ile 2009Q3 arasında farklar bulunmaktadır. Bazı dizinler kaldırılmış bazı yeni dizinler eklenmiş olabilir. Bu farkların dikkate alınabilmesi için öncelikle eski dizinlerin silinmesi ve ardından yeni dosyanın yine aynı dizine açılması gereklidir. Bunu yapmadığınız durumda pkgsrc dizinin hatalar barındıracaktır. Bu nedenle ya eski dizini silmelisiniz ya da bir sonraki sürüme terfi etmek için cvs kullanmanız gerekir.

CVS ile pkgsrc Güncellenmesi

Güncel pkgsrc sürümünü edinmek için cvs en uygun yoldur. Cvs ile pkgsrc'yi güncellemeden önce bazı ortam değişkenlerini ve cvs'yi yapılandırmanız gerekli. Kurulumda root hesabı için C kabuğunu seçtiğimiz için aşağıdaki yapılandırma C kabuğuna özgüdür. Kullandığınız kabuğa göre gerektiği gibi uyarlamanız gereklidir.

```
netbsd501# setenv CVSROOT anoncvs@anoncvs.NetBSD.org:/cvsroot
netbsd501# setenv CVS_RSH ssh
```

cvs ile birçok kullanıcı ilk defa kullandıklarında hata mesajları alır veya bekledikleri işlemlerin gerçekleşmediğini görürler. Bu "beklenmedik durumların" önüne geçmek için ".cvsrc" dosyasını oluşturup içeriğine aşağıdaki komutları eklemek yeterlidir. Bu yapılandırma ile ilgili ayrıntılı bilgi pkgsrc kılavuzlarında yer alır. Aşağıdaki yapılandırma pkgsrc kılavuzunda önerilen yapılandırmadır.

```
#.cvsrc ayarlar
checkout -P
update -dP
release -d
diff -upN
cvs -q -z3
rdiff -u
```

Bu aşamadan sonra cvs kullanarak kararlı sürüm olan pkgsrc-200xQy'ye terfi edebiliriz. Aşağıdaki örnekte 2009Q2'den 2009Q3'e güncelleme yapıyoruz

```
netbsd501# cd /usr/pksrc
netbsd501#  cvs update -dP
```

Bu güncelleme işleminden sonra CVS sadece /usr/pkgsrc dizininde bulunan, var olan pkgsrc sürümünü kararlı sürüm ile karşılaştıracak, eski dosyaları güncelleyecek ve varsa kaldırılmış olan dosyaları da silecektir. Üzerinde değişiklik yapılmamış olan dosyalar ile dizinler olduğu gibi kalacaktır. Eğer cvs tarafından kontrol edilen bazı pkgsrc dosyaları üzerinde işlem yaptıysanız bu durumda cvs yaptığınız düzenlemleri güncel sürüme ait değişiklikler ile birleştirmeye çalışacaktır. Ortaya çıkabilecek olan problemleri çözmek için cvs kılavuzundaki update bölümüne göz atmanızda yarar var.

pkgsrc ile Paket Yönetimi

pkgsrc, ports'da olduğu gibi kaynak koddan derleme yanında hazır derlenmiş paketleri de kullanmanızı sağlar. pkgsrc indirdiğiniz sunucuda kullanmakta olduğunuz pkhsrc sürümüne ait olan paketler -üzerinde çalıştığımız sistem NetBSD olduğu için-

ftp.netbsd.org/pub/pkgsrc/packages/NetBSD/platform/pkgsrc_sürüm/

adresinde yer alır. Bu paketleri doğrudan yazılım kurmak ve kaldırmak için kullanabilirsiniz. Bu paketleri kurmadan önce yapılandırmanızı

hazırlamalınız gereklidir. pkgsrc bir tar dosyasıdır ve NetBSD kurulum CD içinde yer almadığı için sonradan tanımlanması gereklidir.

Hazır Paketleri Kullanarak Yazılım Kurulumu

Paketleri uzak bir sunucudan alabileceğiniz gibi yerel ağ üzerinde bulunan bir sunucudan da edinebilirsiniz. Sunucunun uzak veya yerel ağ üzerinde olmasının bir önemi yoktur. Sadece adresinin ortam değişkenlerinde tanımlanmış olması yeterlidir. Bu işlemi root kullanıcısının kabuk yapılandırma dosyasında örneğin csh.login dosyasına ekleyerek giriş yaptığında aktif olmasını sağlayabilirsiniz. Eğer bunu yapmak istemiyorsanız gerekli değişken tanımlamalarını gerektiğinde aşağıdaki gibi yapabilirsiniz. Paketlerin bulunduğu sunucuların adreslerini netbsd ftp sunucularından öğrenebilirsiniz. Standart olarak

ftp://ftp.netbsd.org/pub/pkgsrc/packages/NetBSD/platform/pkgsrc_sürüm/

adresinde yer alır.

```
netbsd501# PATH="/usr/pkg/sbin:$PATH"
netbsd501#
PKG_PATH="http://nyftp.netbsd.org/pub/pkgsrc/packages/NetBSD/i
386/5.0.1_2009Q3/All/"
netbsd501# export PATH PKG_PATH
```

Bu aşamadan sonra kurmak istediğiniz paketi adını yazarak geçen bölümde anlattığımız pkg_add kullanarak kurabilirsiniz. Sözkonusu paketleri NetBSD ftp sunucusunda bulamıyorsanız paketlerin bulunduğu adresi belirten bir metin dosyası tutulmaktadır. Yazının yazıldığı tarihte netBSD 5.0.1 sürümü için PKGSRC-2009Q3 sürümüne ait olan paketler

http://nyftp.netbsd.org/pub/pkgsrc/packages/NetBSD/i386/5.0.1_2009Q3/All/

adresinde yer alıyordu.

```
netbsd501# pkg_add -vr openoffice2
```

Söz konusu kurmak istediğiniz paket için gereken diğer bağımlı olduğu paketler de bu işlem sırasında kurulacaktır. Bunun gerçekleşmesi için sadece paketin değil aynı zamanda bağımlılıklarının da tanımladığınız sunucuda bulunması gereklidir. Aksi halde kurulum işlemi başarısızlıkla sonuçlanacaktır.

Kurduğunuz paketleri, kaldırmak için pkg_delete kullanabilirsiniz. pkg_add, pkg_delete ve pkg_info kullanımı önceki bölümde ports yazısında anlattığımız gibi işlemektedir.

Kaynak Koddan Derleyerek Yazılım Kurulumu

pkgsrc'yi sisteminize kurduğunuzda yazılımların FreeBSD'deki port ağacına benzer bir yapıda olduğunu göreceksiniz. Port ağacında olduğu gibi pkgsrc üzerindeki paketlere ilişkin bağımlılıklar vs. bilginin barındırıldığı README.html dosyalarını hazırlayabilirsiniz. Bunun için /usr/pkgsrc dizinine geçip

```
netbsd501# make readme
```

komutunu vermeniz yeterlidir. README.html dosyaları her bir paket için ilgili dizinde yer alacaktır. Daha sonra bu dosyaları lynkx veya firefox gibi ibr tarayıcı ile görüntüleyerek inceleyebilirsiniz. Bu dosyalar ileride ortaya çıkabilecek olan sorunları aşmanızda işiniz kolaylaştıracaktır.

Kaynak koddan derleme işlemi yaptığınızda kurduğunuz paketler /usr/pkg dizini altında yer alacaktır. Bu dizini değiştirmek isterseniz mk.conf dosyasını düzenlemeniz gereklidir.
Kaynak koddan derleme yaparak paket kurulumu yapabilmeniz için -buradaki örnekler NetBSD 5.0.1 kullanılmaktadır- kurulum sırasında

dağıtım seçenekleri içinden derleyici ve ilgili araçları da seçmeniz gereklidir. X11 kullanacak iseniz X11 dağıtımlarını da kurmanız zorunludur.

Gerekli olan tüm dağıtımların kurulmuş olması durumunda pkgsrc'yi NetBSD sisteminizde kolaylıkla kullanabilirsiniz. pkgsrc içinde göreceğiniz üzere meta-pkgs adlı bir dizin bulunmaktadır. Bu dizinde yer alan paketleri kurmak istediğinizde -örneğin meta-pkgs/kde3 olduğu gibi- kaynak koddan derlenerek gerçekleşecek olan kurulum süreci indirilecek olan kodun büyüklüğü, bant genişliği, ile sunucudaki yoğunluğa bağlı olarak değişkenlik gösterecektir. Bu nedenle meta paketlerde ve diğerlerinde öncelikle kaynak kodların indirilip ardından gerekli olan kuruluma geçilmesini sağlayabilirsiniz.

```
netbsd501# make fetch-list | sh
```

Bu komut gerekecek olan tüm kaynak kadları indirip ardından derleme, kurulum işlemine başlayacaktır. İsterseniz kaynak kodları kendiniz elle indirip ardından bu işlemi gerçekleştirebilirsiniz.

PKGSRC ile kurulum işlemi önceki bölümde ele aldığımız port ağacı kurulumundaki make kullanılması ile aynıdır. Kurulum işleminin ardından work dizininde yer alan dosyaların kaldırılması için

```
netbsd501# make clean
```

komutu yeterli olur. Öte yandan kurulum sırasında gerekli olan bağımlılıklar da kurulacağı için bu bağımlılıklara ait olan dosyalara da gerek olmayacağı için silinebilirler.

```
netbsd501# make clean-depends
```

Kaynak koddan derleme işlemleri port ağacındakinden farklı değildir. NetBSD'nin kullandığı pkgsrc tüm derleme işlemleri için kullanılan pkgsrc/mk/defaults/mk.conf dosyasını kullanır. Bu dosyanın içeriği gerekli olan düzenlemelerin nasıl yapılacağını anlatmaktadır. Bu dosyayı doğrudan kullanmak yerine /etc altına /etc/mk.conf olarak kopyalayarak kullanabilirsiniz. Bu dosya üzerinde yapılacak değişiklikler öncelikle make tarafından dikkate alınacaktır. Eğer bu dosya bulunmuyorsa mk/defaults/mk.conf dosyası esas alınır. Ancak gereken düzenlemelerin öncelikle /etc/mk.conf üzerinde yapılması ve asıl kaynak dosyanın korunması yerinde olacaktır.

Kurulu Paketlerin Güvenlik Açıklarına Karşı Kontrol Edilmesi

Ports'da olduğu gibi pkgsrc da belirlenen güvenlik açıklarının söz konusu sistemdeki paketlerde bulunup bulunmadığını kontrol etmemizi sağlayan security/audit-packages uygulamasına sahiptir. Çalıştırdığınız zaman NetBSD ftp sunucularında barındırılan

ftp://ftp.netbsd.org/pub/pkgsrc/distfiles/vulnerabilities

dosyasını indirip paketleri kontrol eder ve sorunlu paketleri raporlar. Bu işlem sistem yöneticisi tarafından elle yapılmak durumunda olduğu için cron ile düzenli aralıklarla güncellenmesi elle yapılan kontrollere göre daha etkin olacaktır.

Root kullanıcısı olarak aşağıdaki cron girdisini kullanarak düzenli olarak güncelleyebilirsiniz.

```
# download vulnerabilities file
0 3 * * * /usr/sbin/pkg_admin fetch-pkg-vulnerabilities
>/dev/null 2>&1
```

Security/audit-packages iki aşamalı olarak çalışır. Birinci aşama download-vulnerability-list olarak adlandırılır ve adından anlaşılacağı gibi pkgsrc yer alan paketlere ilişkin açıkların bir listesini NetBSD FTP sunucularından indirmektedir. İkinci aşama sistemde kurulu olan paketlerin sözkonusu listede yer alan güvenlik açıkları ile karşılaştırarak sorunlu paketleri listeler. Paketlerden herhangi birisinde

bir açık bulunuyorsa durumu belirten bir uyarı mesajı döndürecektir.

Kurulu Paketleri Terfi Etmek

pkgsrc ile kurduğunuz paketleri terfi etmek, eski bir pkgsrc sürümünden güncel olanına terfi ettiğinizde söz konusudur. Bu durumda güncel pkgsrc'da yer alan paketler ile sistemde kurulu olanlar karşılaştırılarak terfi edilecek olan paketleri belirleyebilirsiniz. Paketleri güncelleme için kontrol etmek için lintpkgsrc uygulaması kullanılır. lintpkgsrc kurulu olmadığı için öncelikle kurmanız gerekecektir. Kurulumun ardından çalıştırıp güncellenecek olan paketleri aşağıdaki gösterildiği gibi belirleyebilirsiniz:

```
netbsd501# lintpkgsrc -i
```

Komutun çıktısı aşağıdakine benzer olacaktır

```
Version mismatch: 'tcsh' 6.15.00nb2 vs 6.16.00
```

Bu aşamada kurulu olan paketleri bir üst sürüme terfi etmek için aşağıdaki yöntemlerden birisini kullanabilirsiniz.

Hazır Derlenmiş Paketleri Kullanarak Güncelleme

pkgsrc her yeni sürümünde NetBSD ve pkgsrc geliştiricileri yeni sürüm pkgsrc için en son NetBSD sürümü için paketleme yaparlar. Paketleme işlemi tüm pkgsrc paketlerinin hazırlanması işlemidir. Kendi sisteminizde kaynak koddan pkgsrc ile paket derlemek yerine hazır derlenmiş paketleri kullanabilirsiniz. Öncelikle söz konusu paketlerin bulunduğu adresin ve ilgili diğer dosyaların bulunduğu dizinleri ve adresleri $PATH, $YOL değişkenine atayıp sonra işleme devam etmeniz gerekir.

Gereken düzenlemelerin ardından pkg_add ile sunucudaki paketleri kullanarak terfi edebilirsiniz. Yukarıdaki örnekte tcsh paketinin güncel sürümüne terfi etmek istiyoruz.

```
netbsd501# pkg_add -uu tcsh
```

pkg_add kılavuzunu okuduğunuzda pkg_add -u sadece söz konusu paketin güncel sürümünü kuracaktır. Bu pakete gereksinim duyan diğer paketler (+REQUIRED_BY) ise olduğu gibi korunur. Eğer diğer paketlerde de güncellenecek olanlar var ise onları da güncellemek için pkg_add -uu kullanılmalıdır.

Burada bir noktaya dikkat etmek gerekiyor. Paylaşımlı kütüphaneler pkg_add -uu ile güncellenmeyebiliyor. Bu konuda girilmiş bir hata bildirimi bulunmaktadır. Söz konusu hatanın neden olabileceği sorunları ortadan kaldırmak ve güncelleme sırasında gerekli olan paylaşımlı kütüphanelerin de güncellenmesi için kontrol edilmesinde yarar var. Bu işlem için pkg_chk kullanabiliriz.

```
netbsd501# pkg_chk -b -P
http://nyftp.netbsd.org/pub/pkgsrc/packages/NetBSD/i386/5.0.1_
2009Q3/All/ -a -C pkg_chk.conf
```

Kaynak Koddan Derleme Yaparak Güncelleme

kaynak koddan derleme yaparak güncelleme işlemleri için pkgsrc kararlı güncel sürümünü kullanmanız gereklidir. pkgsrc'nin kısmen güncellenmiş olması her türlü probleme davetiye çıkarmış olmak demektir.

Kurulu paketleri terfi etmek için öncelikle lintpkgsrc ile güncellenecek paketleri belirleyerek başlayın. Herhangi bir paketin güncelenmesi için, örneğin tcsh paketini güncellemek için, /usr/pkgsrc üzerindeki ilgili paketin yer aldığı dizine geçip ardından aşağıdaki komutu vermeniz gereklidir:

```
netbsd501# make update
```

Yukarıdaki komut öncelikle sözkonusu paketi ve bu pakete gereksinim duyan paketleri kaldıracaktır. Bu işlemin ardından ise sırayla gereken tüm paketleri derlemeye başlayıp kuracaktır. Bu işlemin sorunsuz bir şekilde sonuçlanmasını beklememek gereklidir. Çok sayıda paketin derlenmesi sırasında bir hata olması nedeni ile make update süreci yarım kalacaktır. Bu durumda aynı komutu verip işlemi yinelerseniz süreç yeniden başlayacak ve kaldığı yerden devam ederek tamamlanacaktır. Bu çözümün olumsuz yanı ise güncelleme sırasında sistemin kullanımı açısından sıkıntılar yaratabilmekte olduğu için kullanıcılar tarafından pek tercih edilmeyen bir yöntem olmasıdır.

make update ile aynı zamanda derlenmiş paketleri de kullanabilirsiniz. Bunun için ise "UPDATE_TARGET=bin-install" tanımlamak gerekir. Bu işlemin gerçekleşmesi için ise ${PACKAGES} kullandığınız pkgsrc sürümüne ait olan derlenmiş paketleri barındıran dizini göstermelidir. Aksi halde make update doğası gereği kaynak koddan güncelleme yapacaktır.

make update sürecinde eski paketlere geri dönmek olanaklı olmadığı için eski paketlerin bir yedeğinin alınması uygun olur. Bunun için pkg_tarup kullanılmalıdır. pkg_tarup kurulmuş olan bir paketin daha sonradan yeniden kullanılmak üzere tgz paketini hazırlar. make update ile olası bir sorunda açıkta kalmamak için yedekleri oluşturmakta kullanabilirsiniz.

pkg_tarup ile bir programın sonradan paketini hazırlamak için

```
pkg_tarup -a -d <paketlerin_barındırılacağı_dizin>
<paketlenecek_program/pksrc>
```

Buradaki -a seçeneği gerekli olan bağımlılıkları da paketler. make update yapmadan önce yedekleri almakta yarar var.

make ile yapabileceğiniz bir diğer güncelleme işlemi "make replace"dir. "make replace" özellikle kurduğunuz veya terfi ettiğiniz paketin ABI-Application programming BInary- uyumsuzluğu ortaya çıkması durumunda bu sorunun aşılması için kullanılır. Bu durum kurulan paketin kullandığı ABİ var olan diğer paketlerdeki ABI ile uyumsuzdur. Bunun için uyumsuzluğa neden olan paketlerin kaldırılıp yeniden kurulum yapılması gerekir. make replace bu işlemi teorik olarak gerçekleştirecek olan uygulamadır. "make replace"in gerektiği gibi çalışması için sorunlu olan pkgsrc ağacı üzerindeki paket üzerinde yani dizinde çalıştırılması gerekir. "make update"te olduğu gibi sorunlu olan paketler için işlemin birden çok tekrarlanması gerekebilir. İşlemin sürekli yinelenmesi yerine pkg_rolling-replace kullanılması daha uygundur pkg-rolling-replace zaten make replace komutunu kendisi çalıştırmaktadır.

pkg_rolling-replace "make update"e göre süreci otomasyon ile gerçekleştirmektedir. Her güncelleme işleminden sonra durumu kontrol edip gerekli olan paketleri yeniden kontrol eder ve gerekli olan paketleri sıralayarak yeniden derleme işlemini başlatır. Bu nedenle her paket için "make replace" çalıştırmaya göre daha etkin bir çözümdür.

ABI sorununu ortadan kaldırmak ve gereken tüm paketlerin yeniden derlenerek kurulması için aşağıdaki komut yeterli olacaktır:

```
pkg_rollin-replace -rsuv
```

Bu komut gereken tüm güncellemeleri yapacak ve her programın bir paketini hazırlayacaktır. Derleme işlemi bağımlılıkları da dikkate alacağı için bu işlem uzun sürebilir. Bu durumda derleme sürecini kısaltmak için

```
pkg_rolling-replace -uv
```

komutu kullanılabilir. GTK+ yaygın kullanılan bir kütüphane olduğu için birçok uygulamada kullanılmaktadır. Örneğin openoffice paketinin yeniden derlenmesi çok uzun bir zaman alacağı için bunu atlayabiliriz.

```
pkg_rolling-replace -rsv -X openoffice2
```

"make replace" ile Karşılaşabileceğiniz Sorunlar

ABI değişmesi durumunda pkg_rolling-replace kullanılması en iyi çözüm iken bazı durumlarda işlem başarısızlıkla sonuçlanacaktır. Bu durum bir paketin adının değişmesi veya farklı iki veya daha çok sayıda pakete ayrılması durumunda ortaya çıkar. Kurulum işlemi için eski paketin yeniden derlenmesi gereklidir. Eski paketin yeni sürümde data ve libs gibi iki bileşene ayrılması durumunda "make replace" eski paketi kurmaya çalışacak ama başarılı olmayacaktır. Bu durumda make update ile söz konusu paketi güncelleyebilirsiniz. make update paketi ve bağımlılığı olan tüm paketleri kaldıracak ve yeniden kurulumu gerçekleştirecektir.

Güncellemelerde, ABI uyumsuzluklarında karşılaşabileceğiniz sorunlar bir paketin kurulması sırasında karşılaşbileceğiniz sorunlar ile aynıdır. Dolayısıyla da kurulum sırasında uyguladığınız çözümler pkg_rolling-replace ile de kullanılabilir. pkg_rolling-replace pkgsrc içinde pkgtools dizininde yer alır.

pkg_rolling-replace ,diğer pkgtools uygulamaları gibi bir kabuk programıdır. Çalıştırdığınızda tüm paketleri bağımlılıklarına göre listeler ve ilk olarak güncellenecek olan paketten başlayarak sırayla hepsini günceller.

pkg_chk ile Kaynak Koddan Güncelleme

pkg_chk hem kaynak kodları hem de hazır paketleri kullanır. Kaynak kodu kullanacak iseniz öncelikle güncellenecek olan paketleri belirlemeniz gereklidir.

```
netbsd501# pkg_chk -u -q
```

Bu komutun çıktısı yukarıdaki örneklerdeki ile aynı olacaktır. Sistemdeki tüm paketleri bir tek komutla ve bağımlıklarını da dikkate alarak kaynak kod kullanarak güncellemek için aşağıdaki komut yeterli olacaktır.

```
netbsd501# pkg_chk -u -s
```

pkg-chk kullanacak iseniz diğer örneklerde olduğu gibi güncellenecek olan paketlerin öncellikle sitemden kaldırılıp ardından güncelleme işlemi gerçekleştirecektir. pkg-chk'nın iyi tarafı ise kaldırdığı paketleri pkg_chk<pkat_adı> ile pkgsrc dizini altında saklamasıdır. Eğer eski sürüm paketlere gereksinim olursa bu dizinlerden paketlerinizi geri alabilirsiniz.

BSD - XII

BSD yazıcı yapılandırması

BSD sistemlerde iki farklı yazıcı sistemi kullanılır. Bunlar LPD ve CUPS'tır. LPD ve CUPS işlev olarak sistemdeki yazıcının kullanıcılar tarafından kullanılmasını sağlamayı amaçlar. Yapılandırma açısından bakıldığında benzer yanları olmakla birlikte birbirlerinden ayrıldıkları noktalar da bulunmaktadır.

BSD sistemlerde hangisinin kullanılacağı ise sistem yöneticisinin tercihine bağlıdır. Eğer ek bir yazılım kurmadan yazıcıyı kullanmayı tercih ediyorsanız bu durumda LPD -Line Printer Daemon- kullanılmalıdır. LPD, ilk bakışta yapılandırması zor gibi görünse de aslında UNIX ve türevi sistemlerde kullanılan ve gelecek bölümde üzerinde duracağımız CUPS'a göre daha kolay ve basittir. CUPS veya LPD kullanılsa da işin zor olan kısmı kullanacak olan yazıcının bilgisayar ile olan bağlantısına ve yazıcınızın özelliklerini ne kadar iyi bildiğinize bağlıdır. Bunlar içinde GDI adı verilen hiç bir UNIX sitem ile çalışmayan -soft modemler gibi- yazıcılar da bulunmaktadır. GDI yazıcıların işleyişinde eksik olan ve UNIX ve türevleri ile çalışmalarını engelleyen özellikleri bazı işlemlerin yazıcı yerine sistem tarafından -CPU- yapılmasını sağlayan sürücüler ile sunulmasından kaynaklanmaktadır. Bu tür yazıcılar sadece Windows ile çalışmaktadır.

Yazıcı satın alacaksanız veya bir yazıcınız var ise Postscript uyumlu olması artı değerdir. Günümüzde satılan yazıcıların büyük bir bölümü postscript uyumlu yazıcılar değildir. Dolayısıyla da alırken Postscript desteği bulunan bir yazıcı alırsanız işiniz kolaylaşacaktır. Eğer postscript uyumlu bir yazıcı bulamadıysanız veya elinizdeki postscript uyumlu olmasa da, BSD sisteminizde kullanabilirsiniz. Bu yazıda LPD yapılandırması ve kullanımınına yer veriyorum.

PostScript Nedir?

PostScript, grafik ve metin dosyalarının biçimlendirilmesi ve yazdırılması için geliştirilmiş bir programlama dilidir. Donanımdan bağımsız olması sayesinde Postscript uyumlu -PostScript dilinden anlayan- tüm yazıcılarda çalışır. PostScript, 1985 yılında Adobe tarafından geliştirilmiştir. Bugün piyasada bulabileceğiniz yazıcıların büyük bir çoğunluğu tarafından desteklenmektedir. Donanımdan bağımsız olması nedeni ile dosyalardaki metinlerin ve grafiklerin yazdırılmasını kolaylaştırmaktadır. PostScript, aktarılabilir-portable- ve platform bağımsız olması nedeniyle kısa zamanda benimsenmiştir. Hatta NeXTSTEP işletim sisteminin grafik motoru olarak da kullanılmıştır.

Gökşin Akdeniz
goksin@enixma.org

PostScript desteği sunmayan yazıcılar PostScript belgelerini yazdıramazlar. Bu durumda getirilen çözüm ise GhostScript'tir. GhostScript PostScript dosya formatını PostScript destekli olmayan yazıcıların yazdırabileceği formata dönüştürmektedir. Böylece PostScript destekli olmayan yazıcılar, GhostScript ile PostScript dosyalarını yazdırabilirler.

LPD Kullanımı

LPD arka planda çalışır. İletilen yazdırma isteklerini kabul eder ve gerekiyorsa bunları filtreden geçirip kuyruğa ekleyerek sırası geleni yazıcıya yollar. Bu işleyiş tüm işletim sistemlerinde aynıdır. Kuyruk dediğimiz vaya bildiğiniz adı ile yazdırma kuyruğu aslında sabit disk üzerindeki bir alandır. Sisteme bağlı birden fazla yazıcı varsa her yazıcı için ayrı bir kuyruk bulunur. Yazdırma kuyruğunda bulunan yazdırma işleri -jobs- spool[1] adı verilen bir alanda barındırılır. Yazdırma kuyruğu spool adı verilen disk üzerindeki alanı kullanır. Bu alanda saklanan işler yazdırma kuyruğu ile yazıcıya yollanır. Dolayısıyla yazıcının sisteme bağlı olması veya olmaması, ağ yazıcısı ile ağa bağlı olmanız veya olmamanız, yazıcının açık veya kapalı olmasının bir önem yoktur. Yazıcıya erişilebildiği anda bekleyen tüm yazdırma işleri sırası ile yazıcıya yollanır. Sistemde tanımlı olan tüm yazıcılar için bu yazıcılara özel yazdırma kuyrukları bulunur.

Sistemdeki bir kullanıcı bir yazdırma işi yolladığında bu işlem alınır ve spooler'da tutulur. Yazıcı erişilebilir olduğunda yazdırılır. Yazdırma işleri arasında öncelikler tanımlanabilir. Bu durumda ilgili yazdırma işlemleri, önceliklerine bakılarak yazıcıya yollanır. Spooler ve yazdırma kuyruğu işleyişleri nedeni ile "canlı" çalışmazlar. Yazıcı erişilebilir olduğunda yazdırma işlemleri tamamlanır. Bu çalışma şeklinin yararları şunlardır:

* Yazıcının birden çok kullanıcı tarafından paylaşılmasına olanak verir.

* Arka planda çalıştığı için kullanıcıların yolladıkları yazdırma işleri sıraya konulup saklanır. Böylece yazdırma işlemleri kaybolmaz. Yazdır komutunu verdiğinizde veya düğmeye bastığınızda işlemin sonucunu beklemek ve çıktı alamadıysanız yeniden yazdırmak için işlemi yinelemenize gerek kalmaz. İşiniz kayıt altına alınmıştır ve yazdırılacaktır.

* Yazıcıda sorun olması durumunda tüm yazdırma işlerini kaybetmiş olmazsınız. Yazdırılmamış olan tüm işler halen saklıdır. Sadece yazdırma işlemi gerçekleşirken oluşan hatalardan dolayı yeniden çıktı almanız gerekir. Yazıcı çalışır duruma geldiğinde bekleyen tüm yazdırma işleri tamamlanır.

Yazdırma kuyruğunda yer alan yazdırma işleri spool dizininde barındırılır. Bu dizin varsayılan olarak /var/spool dizinidir. Sistemde bir tane yazıcı bulunuyorsa paralel, USB veya ağ yazıcısı ayrımı yapılmadan /var/spool/lpd dizinini kullanır. LPD yapılandırmasını düzenleyerek bunu /var/spool/lpd/<inkjet895cxi> şekilnde kullanabilirsiniz.

Yazıcınızı BSD Sisteme Tanıtmak

Yazıcılar dört farklı bağlantı ile bilgisayara bağlanabilirler. Bu bağlantılardan bazıları güncel yazıcılarda kullanılmamaktadır. Eğer bir gün elinize bu tür bir yazıcı geçecek olursa bunu eski bir bilgisayarda BSD kurup kullanabilirsiniz. Bu eski yazıcılardan halen çalışır durumda olan ve kullanılmakta olanlar var ve kullanıcılar bu yazıcılara bir yazdırma sunucusu üzerinden erişebiliyor.

* RS-232/COM Port Bağlantısı: Bugünün bilgisayarlarında pek kullanılmayan ancak eski bilgisayarlarda bulunan bir seri port bağlantı noktasıdır. Bu bağlantılardaki veri iletim hızının üst değeri 115200 bps'tir. Bu nedenle büyük dosyaların yazdırılmasında kullanılacak

yazıcılar için uygun bir seçenek değildir. Özel bir kabloya gereksinim duyarlar.

* Paralel Port Bağlantısı: Bu bağlantılar RS-232 göre daha hızlıdır. Piyasada kablosunu bulmak RS-232'e göre daha kolaydır. paralel port üzerinden gerçekleşen bir iletişim olmadığı için bu arayüzlerin yapılandırılması diğerlerine göre daha kolaydır. Bazı kaynaklarda paralel bağlantı için "Centronics" adı da kullanılır. Bu isim kullanılan bağlantı kablosunun üreticisinden esinlenilerek bulunmuştur. Paralel port üzerinden iki yönlü iletişim iki şekilde gerçekleşmektedir. Birincisi, yazıcı ile iletişimin sağlanması için özel olarak geliştirilmiş sürücülerin kullanılmasıdır. Bu genellikle mürekkep püskürtmeli yazıcılarda sık olarak kullanılır.Böylece mürekkep seviyesi vb. bilgiler yazıcıdan öğrenilebilir. Diğeri ise yazıcının PostScript desteğine sahip olması durumunda gerçekleşir. PostScript.destekli yazcılarda yazdırma işleri doğrudan yazıcıya yollanır. yazıcıdan gelen bilgi -örneğin hatalar, kağıt sıkışması vb.- bilgisayara döndürülür. Bu iki yönlü iletişim özellikle kullanıcı tarafında olumlu karşılanmaktadır. Bu nedenle PostScript destekli yazıcılar için iki yönlü iletişim zorunludur.

* USB bağlantı: USB-Universel Serial Bus- pararlel port ve RS-232 bağlantılarına göre daha hızlı veri iletişimi yapabilir. Kablolarını bulmak ise RS-232 ve pararle port kablolarına göre daha kolaydır. UNIX sistemlerde USB yazıcılar için destek kısıtlıdır. Hem USB hem de paralel port bağlantısına sahip bir yazıcıda paralel port bağlantısının kullanılması durumunda yazdırma işlemleri USB'ye göre daha kısa zamanda bitmektedir. Paralel port'un veri iletim hızı USB göre daha az olsa da. Genel olarak bakıldığında USB çift taraflı iletişim kurulmasını sağlarken paralel port bağlantıları bilgisayardan yazıcıya tek yönlü iletişim sağlamaktadır. Yeni nesil paralel port olan EPP/ECP, BSD sistemlerde iki yönlü iletişim desteği sunmaktadır. Bu iletişim ise USB bağlantısı üzerinden yazıcı kullanmaya göre yine daha hızlı olmaktadır. Bu hız IEEE-1284 uyumlu bir kablo kullanılması durumunda en yüksek değerlerine ulaşmaktadır.

* Ağ Destekli Yazıcılar: Bu tür yazıcılar doğrudan ağa bağlanabilir. Bu yazıcıların bazı modellerinde spool dizini yer alır. Dolayısıyla da yazdırma işlemleri doğrudan yazıcıya yollanır. Bazı modellerde ise yazıcının kendi spool dizini bulunmaz ve yazdırma işlemleri bilgisayardan yazıcıya yollanır.

Donanım ve İletişim Yapılandırması

BSD sisteminiz ile gelen çekirdekte yazıcıyı kullanabilmeniz için gereken bileşenler çekirdek yapılandırmasında USB, pararlel ve seri port bağlantılı yazıcılar için çekirdek modülleri halinde bulunur. Bu nedenle çekirdek düzeyinde -yeniden derlemek gibi- ek bir işleme gereksinim duyulmaz. Yazıcyı kurmadan önce sisteminizdeki pararlel ve seri portları kontrol etmeniz gereklidir. Bu bilgileri /var/run/dmesg.boot dosyasından edinebilirsiniz. Paralel port için ppc* ve seri portlar için sio* girdilerini görebilirsiniz. Eğer bilgisayarınıza takılı bir USB yazıcı bulunuyorsa bu yazıcı ulpt* olarak bulunabilir.

Burada *, tanımlanan çekirdek tarafından adreslenen donanımı göstermektedir. PC'lerde genel olarak dört adet seri I/O portu bulunur. Bu portlara ilişkin ppc* değerleri ppc[0-3] olarak dmesg.boot dosyasında görülebilir. Bir bilgisayarda bir tane paralel port bulunacağı için paralel port takılı olan yazıcınız ppc0 üzerinde yer alacaktır. USB yazıcılarda ise sistem çalışırken takılabilen donanımlar olduğu için donanıma ilişkin bilgiler doğrurdan çekirdek mesajları içinde görülebilir. BSD sisteminize takılı olan USB yazıcıya ilişkin bilgileri paralel porta takılı bir yazıcıya göre daha çok bilgi verecektir. Elinizde yazıcıya ait kılavuz bulunuyorsa işiniz çok daha kolay olacaktır.

Paralel, seri ve USB yazıcılar için aşağıdaki komutlar kullanılarak bilgi edinilebilir.

Paralel port yazıcılar için

```
# grep ppc /var/run/dmesg.boot
```

Seri port yazıcılar için

```
# grep sio /var/run/dmesg.boot
```

USB yazıcılar için

```
# grep ulpt /var/run/dmesg.boot
```

Eğer bir kerede tüm yazıcı bilgilerini döndürmek isterseniz aşağıdaki komut işinizi görecektir.

```
# dmesg | grep '^ppc|^sio|^ulpt'
```

Eğer komutun çıktısı herhangi bir bilgi döndürmüyorsa bu durumda ilgili çekirdek modüllerini kldload ile elle yüklemeniz gerekecektir. Paralel port modülü için

```
# kldload lpt.ko
```

Seri port modülü içinde

```
# kldload ppbus.ko
```

Usb modülü için

```
# kldload usb.ko
# kldload ulpt.ko
```

Paralel Port Yazıcı Yapılandırması

Paralel port üzerinden çalıştırdığınız yazıcı ppc0 tarafından kontrol edilen lpt0 donanımıdır. Paralel port yazıcılarını iki farklı şekilde kullanabilirsiniz. Birincisi interrupt-driven mode yani iş kesmeyle başlatılma kipi, diğeri de polled mode-sorgulama kipi'dir. Öncelikle paralel port yazıcıyı hangi kipte çalıştırmak istediğinize karar vermeniz gerekir. Bu kararda etkili olan ise donanımınızdır. Bazı yazıcılar sorguluma kipinde daha hızlı çalışırken bazıları da iş kesme kipinde daha hızlıdır. Teknik olarak iş kesme kipinde çalışma daha hızlıdır. Bu kipin olumsuz yanı ise IRQ-interrupt request- kesme isteği kullanmasıdır. Bazı sistemlerde yeterli IRQ olmaması bu durumda performans kayıplarına neden olabilmektedir. Bazı yazıcılar da IRQ kipinde çalışırken kararsız olarak çalıştığı için sorun yaratabilmektedir.

Paralel porta takılı yazıcınızı yapılandırmak için root olarak işlem yapmanız gerekir. Yapılandırma için paralel porta takılı bir yazıcıya ve lptcontrol aracına gerek vardır. Öncelikle yazıcı ile olan iletişimi yapılandırarak başlayalım:

İş kesme kipinde çalıştırmak için

```
# lptcontrol -i -d /dev/lpt0
```

Sorgulama kipinde çalıştırmak için de

```
# lptcontrol -p -d /dev/lpt0
```

Eğer BIOS ve yazıcınız ECP/EPP paralel port iletişimini destekliyorsa bu durumda "-e" parametresini de kullanabilirsiniz. Yukarıdaki komutlarda gördüğünüz "-d" parametresinin kullanımı zorunlu değildir. Doğrudan lptcıontrol aracına /dev/lpt0 aracını kullanmasnını söyleyebilirisiniz. Ancak elinizde birden fazla parelel port bulunuyorsa bu durumda "-d" parametresi ile birlikte kullanılacak olan portu tanımlamanız gerekir ve bu da zorunludur! Aşağıdaki komutlar bu durumda kullanılacaktır.

İş kesme kipinde çalıştırmak için

```
# lptcontrol -i -e -d /dev/lpt0
```

Sorgulama kipinde çalıştırmak için de

```
# lptcontrol -p -e -d /dev/lpt0
```

"-e" parametresi ile aktif hale getirilen uzatılmış kip -extended mode- "-s" kapatılabilir.

Sistemi yeniden başlattığınızda yukarıda verdiğimiz yapılandırma komutlarının otomatik olarak çalıştırılmasını istiyorsanız bu komutları barındıran bir kabuk programı hazırlayıp /etc/rc.local dosyasının içerisine kayıt etmeniz yeterli olur.

Çekirdek tarafından desteklenen bağlantı noktasını belirledikten sonra söz konusu bağlantının test edilmesi gerekir. Bu aşamada ayrıca yazıcının PostScript desteğini de deneme ve görme olanağınız söz konusu olacaktır. Yazıcınızı bilgisayara bağlayıp root olarak basit bir PostScript dosyası yazdırarak kontrol işlemini gerçekleştirin. İlk olarak bağlantınızın çalışıp çalışmadını kontrol edin.

Paralel port üzerinde bulunan yazıcı için

```
# lptest > /dev/lpt0
```

USB bağlantısı ile kullanacağınız yazıcı için

```
# lptest > /dev/ulpt0
```

Eğer bağlantınız çalışıyorsa yazıcıdan alacağınız çıktıda standart ASCII karekterlerini göreceksiniz. Bu bağlantınızın çalıştığını gösterecektir. İkinci adım olarak PostScript desteğini kontrol edebiliriz. Bu işlem için internetten bir postscript dosyası arayıp yazdırmanıza gerek yok. Bu dosyayı doğrudan kendiniz bir metin editörü ile hazırlayabilirsiniz. Aşağıda gördüğünüz metni test.ps olarak kayıt edin.

```
%!PS
100 100 moveto 300 300 lineto stroke
310 310 moveto /Helvetica findfont 12 scalefont setfont
(Merhaba!) show
showpage
```

Bu dosyayı yazıcıya yollayarak PostScript desteğini kontrol edebilirsiniz.

Paralel port için

```
# cat test.ps > /dev/lpt0
```

USB için

```
# cat test.ps > /dev/ulpt0
```

Yazıcıdan aldığınız çıktıda "Merhaba!" yazısını okuyabiliyorsanız yazıcınız PostScript desteklidir. Kutlarız, doğru seçim!

/etc/printcap Dosyasının Düzenlenmesi

LPD'nin yazıcınıza gönderilen dosyaları yönetebilmesi için /etc/printcap dosyasının düzenlenmesi gerekir. LPD'nin kullandığı spool sistemi bu dosyayı her yazdırma kuyruğuna konan dosya için kontrol eder. Dolayısıyla da bu dosyada yapılacak olan her düzenleme anında uygulamaya konur.

/etc/printcap dosyasını düzenlemek için /usr/share/misc/termcap ve /etc/remote dosyalarını esas alabilirsiniz. Dosyayı düzenlerken şu noktalara dikkat etmek gereklidir:

1- Yazıcıyı tanımlayacak bir isim belirleyin. Bu isim /etc/printcap dosyasında kullanılacağı için bazı noktalara dikkat etmek gerekli. Yazıcıya istediğiniz adı verebilirisiniz. Yazıcı yapılandırmasını elle ve bir metin editörü ile yapmakta olduğunuz için yazıcınızın marka ve modelini belirten bir isim kullanmakta yarar var. Zira super_duper

veya klingon_blaster gibi isimler eğer sistemde birden fazla yazıcı bulunuyorsa bunları ayırt etmekte bir yarar sağlamayacaktır. Yine aynı şekilde eğer aynı sistemden erişebildiğiniz birden çok yazıcı bulunuyorsa bu durumda varsayılan yazıcının adında "lp" yer almalıdır. Diğer yazıcılar varsayılan yazıcı olmadığı için bu ismi kullanmalarına gerek olmayacaktır. Dolayısıyla kendi gereksinmelerinizi düşünerek uygun olan bir yazıcıyı varsayılan yazıcı yapmanızda yarar var. Aşağıdaki örnekte kullanmakta olduğum sistemde yer alan yazıcıya ait olan girdiyi görebilirsiniz.

```
#
#  droideka icin /etc/printcap
#
inkjet895Cxi|lp|HP Deskjet 895Cxi:\
```

LPD varsayılan olarak her yazdırma işinden önce bir başlık sayfası yazdırır. Bu özellikle çok sayıda kullanıcı tarafından yazıcıya yollanan her yazdırma işini birbirinden fiziksel olarak ayırmakta son derece yararlıdır. Bir başlık sayfası söz konusu yazdırma işine ait olan bilgilerden oluşur ve kullanıcıların kendilerine ait olmayan çıktıları almalarını önler. Aşağıda örnek bir başlık sayfasının çıktısını görebilirsiniz [2]

```
k                   ll       ll
k                    l        l
k                    l        l
k   k     eeee       l        l     y    y
k  k     e    e      l        l     y    y
k k      eeeeee      l        l     y    y
kk k     e           l        l     y    y
k   k    e    e      l        l     y   yy
k    k    eeee      lll      lll     yyy y
                                        y
                                  y    y
                                   yyyy

                              ll
                    t          l        i
                     t         l
 oooo    u    u   ttttt       l       ii     n nnn     eeee
o    o   u    u     t         l        i     nn   n   e    e
o    o   u    u     t         l        i     n    n   eeeeee
o    o   u    u     t         l        i     n    n   e
o    o   u   uu     t  t      l        i     n    n   e    e
 oooo     uuu u      tt      lll      iii    n    n    eeee


r rrr     oooo     ssss     eeee
rr   r   o    o   s    s   e    e
r        o    o    ss      eeeeee
r        o    o      ss    e
r        o    o   s    s   e    e
r         oooo     ssss     eeee
```

Bu başlık sayfalarını kullanmak istemiyorsanız iptal etmek için aşağıdaki satırı ekleyin.

```
#
#  droideka icin /etc/printcap
#
inkjet895Cxi|lp|HP Deskjet 895Cxi:\
      :sh:
```

Eğer başlık sayfalarını yazdıracak iseniz :sh: satırını kullanmayın.

3- Bunun ardından yazdırma kuyruğunun barındırılacağı spool dizinini tanımlamak gerekiyor. spool dizini sadece yazdırma işlemlerini barındırmaz aynı zamanda başka dosyaları da barındırır. spool dizinindeki dosyaların sürekli değişen dosyalar olmaları nedeni ile /var altında bulundurulması gereklidir. Dolayısıyla da spool dizini /var/spool olacaktır. Yedekleme işlemlerinde bu dizini yedeklemeye gerek yoktur. Eğer spool dizini bulunmuyorsa bunu kendimiz oluşturabiliriz. Spool dizininde birden çok yazıcıya atanan farklı spool dizinleri kullanılabilir. Bu nedenle aralarındaki ayrımın yapılabilmesi için iki farklı yol izlenebilir. Birincisi, yazıcıya atadığınız ismi burada kullanarak dizin oluşturmanız.

```
# mkdir /var/spool/inkjet895cxi
```

Diğer uygulama ise aynı bilgisayara bağlı birçok yazıcının olması durumunda her bir yazıcıya ayrı bir spool dizini açmak yerine tek bir dizin altında farklı alt dizinlerde toplanmasıdır

```
# mkdir /var/spool/lpd
# mkdir /var/spool/lpd/yazıcı_bir
# mkdir /var/spool/lpd/yazıcı_iki
```

Bu dizindeki yazdırma işlerinin diğer kullanıcılar tarafından görülebilir olmasını istemiyorsanız bu durumda var/spool ve ilgili dizinlerin daemon ve daemon grubunca okunabilir, yazılabilir ve aranabilir kılabilirsiniz.

```
# chown daemon:daemon /var/spool/lpd/yazıcı_bir
# chown daemon:daemon /var/spool/lpd/yazıcı_iki
# chmod 770 /var/spool/lpd/yazıcı_bir
# chmod 770 /var/spool/lpd/yazıcı_iki
```

Bu dizinleri oluşturduktan sonra bu dizinleri /etc/printcap dosyasına ekleyebiliriz.

```
#
#  deroideka icin /etc/printcap
#
inkjet895Cxi|lp|HP Deskjet 895Cxi:\
      :sh:sd=/var/spool/lpd/inkjet895cxi:
```

4- Kullanacağınız /dev girdinizi /etc/printcap dosyasına ekleyin. Bunun için gerekli olan bilgileri topladığımız için yeni bir işlem yapmamıza gerek yoktur. LPD yazdırma kuyruğuna eklenen bir yazdırma işini işleyebilmek için gerekli olan donanımı çalıştıracaktır. Benim kullandığım sistemde yazıcı paralel port üzerinde bulunduğu için /dev/lpt0 olarak girdiyi tanımlıyorum.

```
#
#  droideka icin /etc/printcap
#
inkjet895Cxi|lp|HP Deskjet 895Cxi:\
      :sh:sd=/var/spool/lpd/inkjet895cxi:
        :lp=/dev/lpt0:
```

Eğer /etc/printcap dosyasında "lp=...." biçiminde bir tanımlama yapmaz iseniz bu durumda /dev/lp yazdırma için kullanılacaktır Aslında /dev/lp dizini bulunmadığı için de yazdırmanız söz konusu olmayacaktır.

5- Bu işlemin ardından da filtre yapılandırması gereklidir. Bu özellikle de PostScript destekli yazılarda zorunlu bir işlemdir. PostScript yazıcılar doğrudan metni yazdıramazlar, bunun için öncelikle metin dosyasının Postscript'e dönüştürülmesi gereklidir. Bunun için hazırlayacağınız basit bir kabuk programı ile bir filtre hazırlayabilir ve kullanabilirsiniz. Hazır bir metin filtresi /usr/share/examples/printing dizininde yer almaktadır. Söz konusu örnek dosya if-simple adı ile yer almaktadır. Bu dosyayı doğrudan /usr/local/libexec/if-simple olarak kopyalamanız yeterlidir. Bu dosayayı ardından çalıştırılabilir kılmak gerekiyor

```
# chmod 555 /usr/local/libexec/if-simple
```

Bu işlemin ardından /etc/printcap dosyasına hazırladığımız filtreyi ekleyebiliriz

```
#
#  droideka icin /etc/printcap
#
inkjet895Cxi|lp|HP Deskjet 895Cxi:\
      :sh:sd=/var/spool/lpd/inkjet895cxi:
        :lp=/dev/lpt0:\
        :if=/usr/local/libexec/if-simple:
```

Yazıcının PostScript desteği bulunuyorsa yukarıdaki yapılandırma

işinizi görecektir. Ancak yazıcının PostScript desteği bulunmuyorsa -örneğin benim Hp Deskjet 895cXi yazıcımdaki gibi- yukarıdaki filtre işinizi görmeyecektir. Bu durum yukarıda anlattığımız durumun tam tersidir. Yazıcıya yollayacağınız veriyi PostScript olarak değil yazıcının anlayacağı biçimde göndermeniz gerekir.

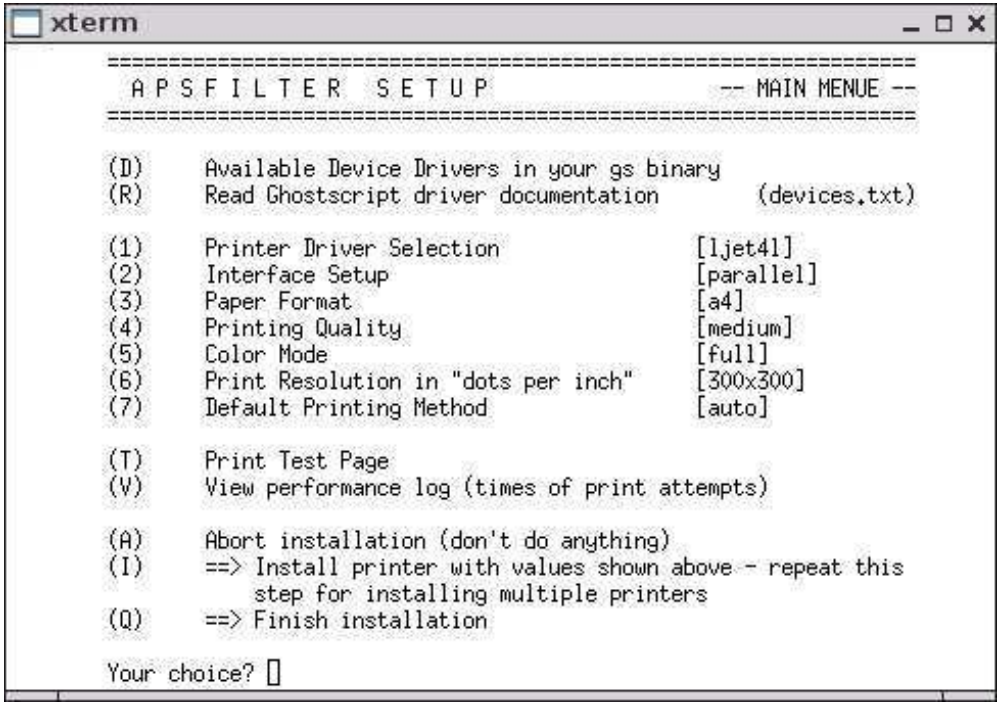
BSD ve UNIX sistemlerde ise PostScript, düz metin dosyası dışında yazdıracağınız her dosya için temel standarttır denilebilir. Yazdıracağınız dosyalar, grafikler vb. dosya formatlar hep PostScript olarak yazıcıyı gönderilir. Bazı uygulamaların bu tür dosyalar için özel araçları olabilir ama UNIX için hazırlanmış bir uygulama kullanıyorsanız PostScript standarttır. Bu nedenle PostScript destekli olamayan uygulamalar veya donanımlar ile çalışacak isenizi işiniz son derece zordur. Bu sorunu aşmak için GhostScript kullanılarak PostScript destekli bir yazıcı kullanılıyormuş gibi yazıcıyı kullanabilirsiniz. GhostScript, PostScript dosyalarını yazıcının anlayabileceği dile çevirir. Yukarıdaki örnekte adı geçen yazıcı PostScript destekli bir yazıcı olmadığı için yukarıdaki filtre işe yaramayacaktır.Çözüm ports üzerinden GhostScript kurumak ve filtre olarak yine /usr/share/examples dizininde yer alan ifhp filtresini kullanmaktır. GhostScript'in desteklediği yazıcıların bir listesini internetten [3] bulabilirsiniz. Bu durumda yukarıdaki yapılandırma dosyası aşağıdaki gibi olacaktır:

```
#
#  droideka icin /etc/printcap
#
inkjet895Cxi|lp|HP Deskjet 895Cxi:\
      :sh:sd=/var/spool/lpd/inkjet895cxi:
        :lp=/dev/lpt0:\
        :if=/usr/local/libexec/ifhp:
```

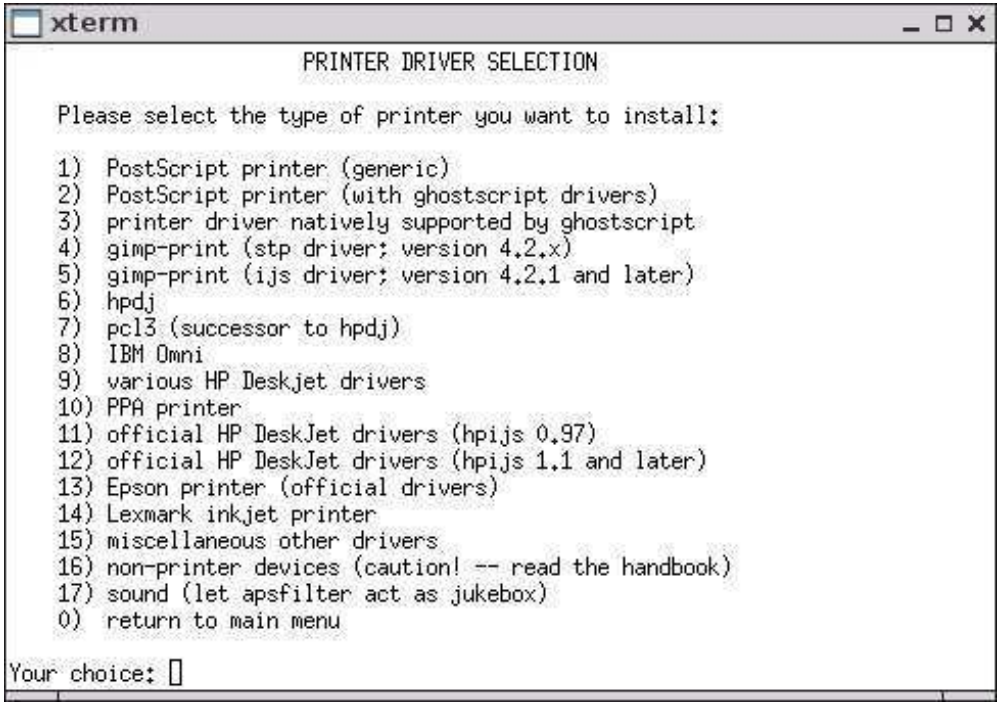
ifhp dosyasını yine yukarıdaki gibi çalıştırılabilir yaparak /usr/local/libexec altına kopyalayarak kullanabilirisiz. ifhp kabuk programına bakacak olursanız karmaşık gibi gelebilir ama işleyişi son derece basittir. Yazdırılacak olan dosya PostScript olarak işlenecek ise GhostScript'e aktarılıp gerekli işlemlerin yapılmasını sağlamaktadır. Aksi durumda yani bir metin dosyası ile doğrudan yazıcıya aktarmaktadır. GhostScript'in desteklediği yazıcıları ve temel yapılandırma seçeneklerini görmek için aşağıdaki komutu kullanabilirsiniz.

```
# gs -h
```

Yukarıdaki filtre örnekleri metin dosyaları için uygundur. Diğer dosya formatları örneğin DVI, grafik vb. gibi için uygun olmayacaktır. Bunun yaratabileceği sorunları aşmak için farklı dosya formatları için hazırlanmış olan filtrelerin kullanılması gerekir. Birden çok filtreyi /etc/printcap dosyasına eklemek zahmetli olacağı için print/apsfilter portunun kurulması işleri kolaylaştıracaktır. Sözkonusu port farklı



Resim 1



Resim 2

dosyalar için gerekli olan filtreleri otomatik olarak seçecek ve kullanacaktır. apsfilter ayrıca ek olarak /etc/printcap dosyasına bir girdi eklemenizi gerektirmez. Tersine kurulum tamamlandığında yapılandırmayı çalıştırıp gereken işlemleri sizin için yapmasını sağlayabilirsiniz. (Resim 1 ve 2)

Artık /etc/printcap dosyasının düzenlenmesi işlemi sona erdiği için LPD aktif kılınarak kullanılmaya başlanabilir.

6- LPD'yi Çalıştırmak

lpd, doğrudan /etc/rc.conf dosyası içerisinde tanımlı olan lpd_enable parametresi ile kontrol edilir. Varsayılan değeri "NO" olduğu için aşağıdaki gibi düzenlenmesi ve /etc/rc.conf dosyasının kayıt edilmesi gereklidir.

```
lpd_enable="YES"
```

Bu aşamadan sonra bilgisayarı yeniden başlattığınızda lpd açılışta çalışmaya başlayacaktır. yeniden başlatmadan çalıştırmak için ise doğrudan komutu vermeniz yeterli olur.

```
# lpd
```

Yazıcının Denenmesi

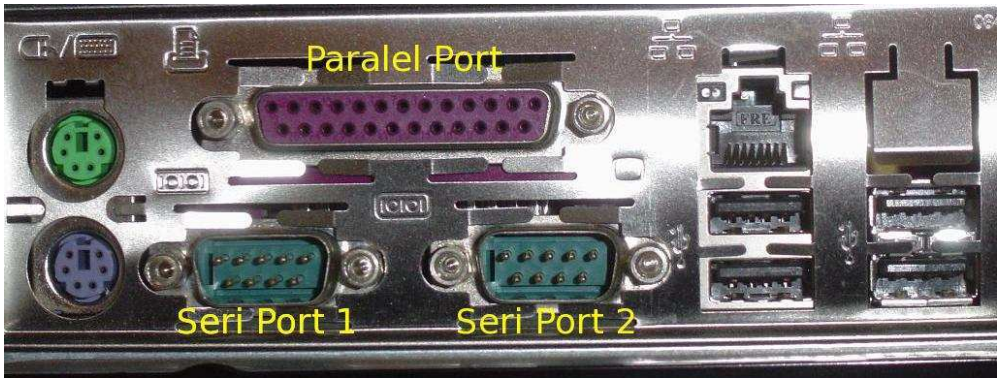
LPD yapılandırması tamamlandığı için bir deneme sayfası yazdırarak kontrol edebilirsiniz. Bu basamağın gerçekleştirilmesi kullanmakta olduğunuz yazıcının yapılandırmasında bir hata olup olmadığını görmek için gereklidir. Denemek için lptest kullanabiliriz. Aşağıdaki komut /etc/printcap dosyasında tanımlı olan yazıcılardan tanımladığınıza <yazıcı_adı> bir deneme sayfası yollayacaktır. Eğer doğrudan varsayılan yazıcıyı denemek isterseniz "-P" parametresini kullanmadan komutu verin.

```
# lptest 20 5 | lpr -P yazıcı_adı
```

Komutun çıktısı aşağıdakine benzer olacaktır. "20 5" değeri yerine farklı değerler koyarak farklı genişlikte ve uzunlukta deneme metinleri yazdırabilirsiniz.

```
!"#$%&'()*+,-./01234
"#$%&'()*+,-./012345
#$%&'()*+,-./0123456
$%&'()*+,-./01234567
%&'()*+,-./012345678
```

Yazıcı çıktısında yukarıdakine benzer bir çıktı görebiliyorsanız bu durumda yapılandırmanız sorunsuz olarak çalışıyor demektir.



Resim 3 - Bilgisayarda paralel ve seri portlar

Notlar:

---------- ----------

[1] SPOOL - Simultaneous Peripheral Operations OnLine. teriminin baş harflerinden oluşur. IBM'de geliştirilmiştir. Dosyaların bir programa veya donanıma gönderilip sıraya konularak işlenmesi sürecini tanımlar. Ganellikle yazıcı kuyruğu anlamında kullanılır.

[2]Başlık sayfası aşağıdaki adresten alınmıştır:
http://www.gta.ufrj.br/~dronsz/freebsd-handbook/handbook95.html

[3] http://pages.cs.wisc.edu/~ghost/doc/printer.htm

BSD - XIII

BSD Sistemlerinde Güvenlik

Gökşin Akdeniz
goksin@enixma.org

Doğru Sandığımız Yanlışlar

Bilgisayarların güvenliği söz konusu olduğunda akla ilk gelen işletim sistemlerinin güvenliğidir. İşletim sistemleri kendi aralarında karşılaştırılır, güvenlik testleri internet sitelerinden indirilir, çalıştırılır ve sistemin ne kadar güvenli olduğunu görmek için test sonuçlarına bakılır. Bu da yeterli gelmez birçok internet sitesi gezilir ve en iyi firewall hangisi sorusunun yanıtı aranır ve sonra en iyisi indirilip bilgisayara kurulur ve ardında yeniden güvenlik testi çalıştırılıp sistem kontrol edilir. Sonuç olumlu ise sorun yok demektir. Bir diğer durum ise internette bulunan bir güvenli sistem yapılandırmasının aynen alınıp kullanıcının kendi sistemine uygulamasıdır. İnternet sitesinde yer alan nasıl yapılır belgesi harfiyen uygulanır ve sistem güvenceye alınır. Bazı kullanıcılar açık kaynak kodlu işletim sistemi kullanmaktadır bu nedenle sistem güvenli olduğu için herhangi bir önleme veya kontrole gereksinim duymazlar, gönül rahatlığı ile bilgisayarlarını kullanmaya devam ederler. Bazı kullanıcılar da konu ile ilgili olarak feşmekan güvenlik yazılımının aslında güncellemeleri yüklerken sisteme zararlı kod yüklediğini belirtir ve bu nedenle feşmekanı kullanmadıklarını belirtirler, hatta bu işi uzman bir arkadaşlarından öğrenmişlerdir. Uzman arkadaşları yıllardır herhangi bir güvenlik yazılımı kullanmadan veya önlem almadan bilgisayarını rahat rahat kullanmaktadır. Bu saydığım olaylar ile bizim kadar sizler de karşılaşmışsınızdır. Hatta bazı topluluk dergilerinde kendi sistemlerinin en güvenli olduğu Windows'un yeni çıkan sürümünün zararlı kodlar için yeni bir mecra olduğu fikrini işleyen karikatürler, yazılara yer verirler. İşletim sisteminin güvenliğini kanıtlamak adına yapılan reklam amaçlı buyurun-saldırın-işe-yaramaz testleri de aslında güvenlik konusundaki bilgi eksikliğini ve cehaleti körüklemek dışında bir yararı da olmamaktadır.

Güvenlik söz konusu ise aslında işletim sistemlerinin güvenliğinden daha önemli olan birçok nokta bilgisizlikten veya konu hakkında yetersiz ya da eksik bilgi sahibi olmaktan dolayı gözden kaçar. Bir sistemin ele geçirildiği veya çalışamaz duruma getirildiği olaylarda işletim sisteminden kaynaklı olan bir zayıflık kullanılmamıştır. Aslında bu olayın gerçek nedeni işletim sistemi üzerinde çalıştırılan uygulamalardır. Uygulamalardaki zayıflık, yazılımı geliştiren ekibin hatası veya güvenlik konusundaki eksilikleri sonucunda yetkisiz girişlere, sistemin zarar görmesi gibi olaylara neden olur. Günümüzde birçok işletim sistemi geçmişte sık rastlanan ve suistimal edilen zayıflıklara karşı önlemler alınarak eskiye göre daha güvenli hale getirilmiştir. Ancak bugünkü durum sistemlerin genel olarak her türlü olumsuzluğa karşı koyabileceği anlamına gelmemektedir. Bugün işletim sistemleri üzerinde halen birçok yeni donanım ve yazılım birbiri ile etkileşim halindedir. Bu etkileşimin olumsuz olaylara neden olması da

olasıdır. Birçok kullanıcı için açık kaynak kodlu işletim sistemlerinin, hatta daha da ileri gidersek UNIX sistemlerin, en güvenli sistem olduğu fikri ile her hangi bir önleme gerek olmadığı düşünülür. Solaris 10 üzerinde yer alan telnetd'deki bir açıklık SOLARIS sistemlerin yetkisiz kişiler tarafından kontrol edilebilmesine olanak veriyordu. Bir başka ciddi olay yine Apache.org'un sunucularına dışarıdan yetkisiz kişilerce erişilmiş olması idi. Olay Apache.org'un kendisine ait olan web sunucuları üzerinde çalıştırılan CentOS Linux dağıtımında bulunan bir ssh anahtarlarına ait olan güvenlik açığının kullanılması ile gerçekleşmişti. CentOS sunucularında yer alan cgi programcıklarına erişilip üzerinde değişiklik yapılmış daha sonra ise CentOS üzerinde cron ile çalıştırılan bazı uygulamalar bu programcıkları Apache.org'un geliştirme sunucularına taşımıştı. Geliştirme sunucuları ZFS dosya sistemi kullanan FreeBSD sistemlerde ise izinsiz giriş sonucunda CentOS'tan kopyalanan kodlar çalıştırılmadan ZFS dosya sistemi olaydan önceki duruma getirilip sorunun daha ciddi boyutlara ulaşması önlenmişti. CentOS sistemine girişte kullanılan SSH güvenlik açığı ve yaması olaydan yaklaşık bir hafta önce duyurulmuştu.

Bu olayda gerçekleşen izinsiz erişime neden olan şletim sistemleri saldırının ana hedefi olmamış, saldırganlar uygulamadaki bir zayıflığı kullanarak web sunucusundan başlayarak diğer sistemlere de ulaşabilmişlerdi. Bunun dışında farklı Linux dağıtımları kullanarak yapılandırılmış olan çeşitli sunucu sistemlerinin zaman zaman bazı botnetlerin parçası oldukları da rastlanan durumdur. Sistem yöneticilerinin bu gelişmeleri çabuk fark etmeleri ile sorunun üstesinden gelinmektedir.

Apache.org ve SOLARIS sistemlerinde olan zayıflıklar dışında diğer UNIX türevi işletim sistemlerinde de sorunlar olabilmektedir. FreeBSD 8.0 ve 7.x'i etkileyen ve yetkisiz kişilerin root olarak kod çalıştırmalarına olanak tanıyan bir kod 8.0 duyurulduğu gün açıklanmış ve kod internete salınmıştı. Söz konusu açığı kapatan yama ise olayın üzerinden bir gün geçmeden yayınlanmıştı.

Güvenlik denildiğinde sadece internet siteleri ile kurumların sunucularının ele geçirilip üzerilerindeki internet sitelerinin ana sayfalarını bu işi yapan kişinin poposunun 10 MP çözünürlükteki fotoğrafını koyması ya da sunuculara yapılan DDos saldırıları ile çökertilmesi olayları akla gelir. Bu olaylar günümüzdeki internet kullanımının yaygınlaşması da dikkate alındığında karşı karşıya kalınabilecek tehlikeler içerisinde en tehlikeli 10 tehdit listesinde ilk sıralarda yer alacaktır ama kesinlikle birinci olmayacaktır.

Sistem yapılandırmasındaki hatalar, sadece birkaç sunucunun geçici olarak kontrol dışı kalmasından daha ciddi sorunlara davetiye çıkarabilir. Kurumsal sistemlerde bulunan kaynaklar sabit disk, bellek, işlemci ve bant genişliği vb sunucuların devre dışı kalmasını istenmeyen bir duruma dönüştürmektedir. Sistemlerin çalışır durumda olması ve kaynakların uzaktan sistem yöneticisi veya yöneticileri dışında da yetkisiz kişilerce kendi amaçları için kullanılması çökmüş bir sistemden daha iyidir. Bu durum geniş bant internet erişimi ile ev kullanıcılarını da olası hedefler haline getirmektedir. Birçok kullanıcı açısından kişisel bilgisayarının bu tür saldırıların hedefi olamayacağı düşünülür. Halbuki bu sistemlerde işlemci, bellek ve sabit disk alanı ile botnet oluşturmak için mükemmel bir hedef duruma gelmektedir. Bu botnetler içerisinde sadece ve sadece Windows sistemler bulunmamaktadır. UNIX ve türevi sistemler için yazılmış olan çeşitli botnet yazılımları hatta kiralanan ve satışa çıkarılan Linux tabanlı botnetler de bulunmaktadır. Arama motorlarında bir kaç arama ile bu tür sitelere ve yazılımlara kolaylıkla ulaşabilirsiniz.

Sorumluluk Kimin?

Güvenlik söz konusu olduğunda işletim sistemlerini geliştiren kişiler kadar sistemleri çalışır durumda tutan ve onlardan birinci derecede sorumlu olan sistem yöneticileri de sorumludur. Kişisel bilgisayarınızdaki sistemin root şifresini biliyorsanız bu durumda sistem

yöneticisi olan kişi de sizsiniz demektir. Dolayısıyla da sorumluluk birinci derecede size aittir.

Tehditlerin Kaynağı

Bugün için karşı karşıya olduğumuz tehditleri sınıflandırmak yerine tehditlerin kaynağına bakmak gerekli yapılandırma ve alınacak önlemleri belirlemek için daha kolay bir çözüm sunacaktır. Uygulamalar ve sistem konusunda bilgili olan saldırganlar zayıflıkları arayıp bulacak ve bunlardan yararlanmak için kod yazacaktır. Bu kişiler tarafından hazırlanan bu araçları kullanarak kendilerine uzman süsü verecek "script kiddie" ilk olarak karşı karşıya kalınabilecek olan tehdittir. Bu tehdidi bertaraf etmek için sisteminizi güncel tutmanız, yamalar ve ilgili güvenlik güncellemeleri ile duyurularını izleyip gerekeni yapmanız yeterli olacaktır.

Diğer tehdidin kaynağı ise yine botnetler olacaktır. Botnetlerin kiralanması ve satışı için hazırlanmış olan sitelerin sayısı bu işin bir ticari faaliyete dönüşmüş olduğunu göstermektedir. Botnetler sonuçta uzman bir saldırgan kadar tehlikeli değildir. Sadece üzerinde çalıştırılan programlar ne kadar etkili ise o kadar etkili olacaktır. Bu tehditleri de savuşturmak için yamaları ve güvenlik güncellemelerini, duyurularını izleyip gereğini yapmak yeterli olacaktır.

Kullanıcılar bir sistemdeki sorun yaratabilecek olan diğer etkendir. Kullanıcılarınızın bazılarının kendilerine verilen şifreleri diğer kullanıcılar ile paylaşmaları ciddi sıkıntılara neden olabilir. Sistemdeki kullanıcılar sisteme doğrudan erişen kişiler olacağı için sistemdeki sorunların ve zayıflıkların farkına varabilir ve bunların nerede olduğunu bilir. Bu durumda bu zayıflıkları kendi çıkarları için kullanabilir veya başka nedenler ile sisteme zarar vermek için bu zayıflıklardan yararlanabilirler. Bu kullanıcılar kendilerine tanınan yetkiler ile tatmin olmayan prens veya prenses kullanıcılar olabilir. Kuralların sadece diğer insanlar için geçerli olduğunu ve kendilerinin bu kurallardan bağımsız olduğunu düşünürler. Herhangi bir hata, zayıflık ve benzerini bu kuralları aşmak için kullanabilirler. İşten ayrılmaları durumunda bile sisteme ilişkin bilgilere sahip oldukları için sorun yaratabilir. Bu durumda işten ayrılan bu kişilere ait olan hesapların kapatılması, kullanıcı hesaplarına ilişkin şifrelerin değiştirilmesi ve gereken duyuruların yapılarak kullanıcıların konu hakkında bilgilenmesi sağlanmalıdır. Bu kullanıcıların sisteme zarar vermeleri veya yetkilerini aşan işlere kalkışmalarının önüne geçebilmek için ayrıca kurum içinde yaptırım uygulanması yoluna da gidilmelidir. Yamaları ve güvenlik güncellemelerini zamanında yapmak da bu tehditlere karşı bir derece güvenlik sağlar. Yukarıdakilere ek olarak başka önlemlerin de alınması gerekir.

Bütün bunlardan daha tehlikeli olanı ise "script kiddie"lerin kullandığı uygulamaları yazan asıl uzmanlardır. Bu kişiler, sadece sistem hakkında değil aynı zamanda ağ ve bu ağ üzerindeki bilgisayarlar hakkında da bilgi toplayabilir ve sistematik olarak çalışırlar. Bu tehdidin önüne geçebilmek için ciddi önlemler alınmalıdır. Öyle ki bu kullanıcının sistem ve ağ hakkında bilgi edinmek için Hollywood casus filmlerine taş çıkaracak sızma ve bilgi toplama yöntemleri geliştirmelerini ve uygulamalarını gerektirmelidir.

Bu son paragrafta adı geçen uzman kullanıcılara dergilerde, gazetelerde, internette "hacker" adı yakıştırılır ve hatta güvenlik konusunda uzmanlık derecesini belgeleyen sertifikalarda dahi "hacker" adı kullanılır. Hacker kavramı bu yazının kapsamı dışında kalmakla birlikte uzmanlığını kötü niyetli olarak kullanan kişilere verilen bir isim değildir. Bu kişilere "carcker/kırıcı" adı verilir. Hacker kavramının özgün anlamı ile kaynağını başka bir sayıda başka bir yazıda okuyabilirsiniz.

Güvenli Bir Sistem Oluşturmak

Güvenli bir sistem oluşturmak için ilk adım güvenlik duyurularını

yakından izlemek ve belirtilen zayıflıkları gideren yamaları gecikmeden uygulamak olmalıdır. Güvenlik duyurularını sizden başka tamamen zayıflıkları kendi amaçları doğrultusunda kullanmak isteyen kişiler de izlemektedir. Dolayısıyla duyurularda adı geçen zayıflıkları arayan birçok script kiddie için ilk kaynak bu duyurulardır. Duyurular, BSD sistemlerde e-posta listeleri ile yapılır ve konuya ilişkin kısa bir açıklama ile yamanın nasıl uygulanacağını belirten bir açıklama içerir. Bu duyuruların yapıldığı e-posta listelerinin mesaj trafiği düşük yoğunlukludur ve diğer yardım vb amaçlı listelere göre posta kutunuza daha az mesaj gelir. Kullandığını BSD sistemin güvenlik duyuruları listesine üye olarak izlemeye başlayın ve gereken önlemleri vakit geçirmeden alın. Zayıflıkları giderilmiş bir sistem birçok tehdidin neden olduğu sorunlardan bağışıktır.

Kullanıcı Hesaplarının Yönetimi

Kullanıcıların neden olabilecekleri tehditlerin önüne geçmek için BSD sistemlerde kullanıcıların yetkilerini ve yapabileceklerini kesin olarak belirleyebilirsiniz. Kullanıcı hesaplarının düzenlenmesi için yapılacak işlemler için başlangıç noktası kullanıcı hesaplarının oluşturulma aşamasıdır. BSD sistemler UNIX Sistemlerin standart kullanıcı yönetim programlarını kullanırlar. adduser(8), passwd (1), pw(8), vipw(8) bunlardan bazılarıdır. adduser bir kullanıcı hesabı tanımlamak için sadece root kullanıcısı tarafından kullanılabilir. Sistem yöneticisinin tercihlerine göre belirlediği yapılandırmaya dayanarak sisteme bir kullanıcı hesabı ekler.

```
Password:
tardis# adduser
Username: fahrettin
Full name: Fahrettin Cureklibatur
Uid (Leave empty for default):
Login group [fahrettin]:
Login group is fahrettin. Invite fahrettin into other groups?
[]:
Login class [default]:
Shell (sh csh tcsh bash rbash nologin) [sh]:
Home directory [/home/fahrettin]:
Home directory permissions (Leave empty for default):
Use password-based authentication? [yes]: yes
Use an empty password? (yes/no) [no]: no
Use a random password? (yes/no) [no]: yes
Lock out the account after creation? [no]: no
Username   : fahrettin
Password   : <random>
Full Name  : Fahrettin Cureklibatur
Uid        : 1003
Class      :
Groups     : fahrettin
Home       : /home/fahrettin
Home Mode  :
Shell      : /bin/sh
Locked     : no
OK? (yes/no): yes
adduser: INFO: Successfully added (fahrettin) to the user
database.
adduser: INFO: Password for (fahrettin) is: h4wGXbwrKwpckFs
Add another user? (yes/no):
```

Kullanıcı adı aynı zamanda kullanıcı hesabının da adıdır. Kullanıcı adı ataması yaparken belli bir sistematik izlemek gereklidir. Sorumlu olduğunuz sistemdeki hesaplar içinde batman, d0rk, piqlet gibi isimler kullanabilirsiniz ama sonuçlarına da katlanmanız gerekir.

Kullanıcının tam adı “Full name:” satırında girilir. Uid ise kullanıcı hesaplarına atanacak olan sayıdır. Bu değeri BSD sisteminiz otomatik olarak tanımlar veya sistem yöneticisi olarak siz belirlersiniz. BSD sistemlerde kullanıcı hesaplarına ait Uid değerleri 1000'den başlar. Küçük değerler sistem kullanıcılarına aittir. Uid değerini kendiniz atamak yerine otomatik olarak atanması için bu işlemi sisteme bırakın. Sıradaki ilk değer otomatik olarak kullanıcıya atanacaktır.

Class satırı sisteme girişte kullanılan login class bilgisidir. Bu bilgide kullanıcıya atamak istediğiniz yetkileri düşünerek hazırlanmış olan

login class -giriş sınıfı- bir tanesi belirtilmelidir. Bu yapılandırma ayrıca üzerinde durulması gereken bir durum olduğu için daha sonra yeniden ele alacağız.

Shell, kullanıcıya atanacak olan kabuk ortamıdır. Sistemde tanımlı olan tüm kabuk ortamları kullanıcı hesabı oluşturulurken listenir. İlgili satırda parantez içerisinde bu kabuklar listelenir.

Home, satırı ise kullanıcının ev dizininin oluşturulacağı dizini tanımlar. Bu dizinde yer alan dosyaların tamamının sahibi kullanıcı ve grubudur.

```
Use password-based authentication? [yes]: yes
Use an empty password? (yes/no) [no]: no
Use a random password? (yes/no) [no]: yes
Lock out the account after creation? [no]: no
```

Bu basamaklarında hesap ile ilişkili şifre oluşturulmaktadır. Kullanıcının kendi şifresini oluşturmasına izin vermek için 'Use an empty password? (yes/no) [no]:' sorusuna “y” veya “yes” olarak yanıt verebilirsiniz. Bu durumda kullanıcı hesabına giriş yapınca kendisi bir şifre atayabilir. Burada ise bir risk ortaya çıkmaktadır. Hesaba şifre atanmadığı için kullanıcı adını yazarak sistem giriş yapan ilk kişi bu hesaba şifre atayabilir. Bu işlemi gerçekleştiren başka birisi ise hesabın asıl sahibi sisteme giriş yapamaz. Öte yandan şifre atama işlemini kullanıcının inisiyatifine bırakmak ise kullanıcının boş şifre ataması olasılığını ortaya koyar. Bu durumda kullanıcı adını bilen herhangi biri bu hesaba giriş yapabilir. Rastgele şifre atama seçeneği -random password- yeni oluşturulan bir hesap için uygun bir çözümdür. Bu şifrelerin anımsanması çıktıda görüldüğü gibi zordur. Bu nedenle kullanıcılar sisteme ilk giriş yaptıkları anda bu şifreyi değiştirmek isteyeceklerdir.

```
Lock out the account after creation? [no]: no
```

Hesabın dondurulmasını isterseniz hesap oluşturulmuş olsa da herhangi bir kullanıcı bu hesaba giriş yapamaz. Bu işlemlerin ardından yaptığınız düzenleme onayınıza sunulur. Onaylayıp hesabı oluşturabilir veya yeniden düzenlemek için başa dönebilirsiniz. Onaylama işleminin ardından yeni bir hesap oluşturmak isteyip istemediğiniz sorulacaktır.

adduser Yapılandırması: /etc/adduser.conf

Yeni bir kullanıcı hesabı oluşturmak için kullanılan adduser, tüm süreci otomatik olarak gerçekleştirir. Kullanıcı ev dizini ve izinleri, kullanıcı adı ve şifresinin /etc/passwd dosyasına eklenmesi, kullanıcının giriş sınıfı vs. ayarlar otomatik olarak yapılır. Bu süreçte kullanılan standart yapılandırma genel olarak yeterlidir. Bu yapılandırmayı kendi tercihlerinize göre düzenlemek isterseniz adduser yapılandırma dosyasının yeniden oluşturulmasını sağlayabilirsiniz. Bunun için ise

```
# adduser -C
```

komutunun verilmesi gereklidir. Bu süreçte hesap oluşturulmasına ilişkin tercihler /etc/adduser.conf dosyasına tanımlanacaktır. Olağan yapılandırmada bu dosya bulunmamaktadır. Ancak sistem yöneticisinin yukarıdaki komutu verip süreci yapılandırması ile dosya oluşturulur. İşlem basamakları adduser komutunun çalıştırılması ile aynıdır ancak bu sefer varsayılan değerler yeniden tanımlanır.

```
# adduser -C
Login group []:
Enter additional groups []
Login class [default]:
Shell (sh csh tcsh bash rbash nologin) [sh]:
Home directory [/home/]:
Use an empty password? (yes/no) [no]:
Use a random password? (yes/no) [no]: yes
Lock out the account after creation? [no]: no
```

Login group değeri eğer tanımlanmaz ise varsayılan grup olarak kalır. Bu da kullanıcı hesabına verilen isim ile aynı isime sahip bir grup oluşturur. Burada hesabın ait olacağı yeni gruplar tanımlayabilirsiniz. Ayrıca -giriş sınıfı- login class benzer olarak tanımlanabilir. Kullanıcı hesapları için atanabilecek olan bir kabuk varsayılan olarak atanabilir. Kullanıcı hesabının yer alacağı ev dizini belirtilebilir. En son aşamada ise kullanıcı hesabı oluşturulurken parola atamasına ilişkin seçim yer almaktadır. Yapılandırmanın tamamlanmasının ardından yeni ayarlarınızı bir sonraki kullanıcı hesabını oluştururken kullanabilirsiniz.

Kullanıcı Hesaplarının Yeniden Düzenlenmesi

Kullanıcı hesaplarının yönetimi denildiğinde akla ilk gelen hesabın oluşturulması ve silinmesidir. Halbuki durum bundan daha farklıdır. Zaman zaman kullanıcı hesaplarının düzenlenmesi ve bazı ayarlarının değiştirilmesi gerekir. BSD sistemlerde kullanıcı hesaplarının düzenlenmesi için birçok araç bulunur ama bunlar içinde yaygın olarak kullanılanlar passwd(1), chpass(1), vipw(8), ve pw(8)'dir. Bu araçlar /etc/master.passwd, /etc/passwd, /etc/spwd.db, ve /etc/pwd.db. dosyaları üzerinde işlem yapar. Bu dosyalar kullanıcı hesaplarına ilişkin bilgileri barındırır. Bu dosyaların yapısı ve amaçları da farklıdır. /etc/master.passwd dosyasında kullanıcı hesaplarına ilişkin parolalar şifrelenmiş olarak tutulur. Bu dosyaya normal kullanıcıların erişim yetkisi yoktur. Öte yandan kullanıcı hesaplarına ilişkin temel bilgiler ise /etc/password dosyasında saklanır. Bu dosyada şifrelenmiş bilgiler yoktur. Bu dosya doğrudan kullanıcı hesapları ile ilişkili tüm uygulamalar tarafından erişebilir ve kullanıcı hesaplarına ilişkin temel bilgileri barındırır. Bu dosyalar metin dosyalarıdır.

Metin dosyaları olması ve kabuk ortamında uygulamaların bu dosyalara erişip okuması ve içeriğindeki bilgiyi ayrıştırması sistemi yavaşlatan bir işlem olduğu için – ilk BSD ve UNIX sistemlerin ortaya çıktığı tarih dikkat alınırsa cep telefonunuzdaki işlemci gücü ve belleğin o zamanki bilgisayarlardan kat kat fazla olduğu görülür - metin dosyasından veriyi ayrıştırmak yerine gerek duyulan bilgilerin bir veri tabanında saklanması yoluna gidilmiştir. Bu veri tabanı da /etc/master.passwd ve /etc/passwd dosyalarından oluşturulan /etc/spwd.db dosyasıdır. Bu dosya sadece root tarafından okunabilir. /etc/pwd.db tüm kullanıcılar tarafından okunabilir ancak içeriği /etc/passwd dosyasında yer alan bilgilerin bir kısmıdır.

Herhangi bir kullanıcı hesabı yapılandırma uygulaması bir kullanıcı hesabı üzerinde bir düzenleme yapacak olsa /etc/master.passwd dosyasındaki bilgiler güncellenmektedir. Bu işlemin ardından pwd_mkdb(8) çalıştırılıp diğer dosyalar güncellenir. passwd(1), chpass(1),ve vipw(8) araçları /etc/passwd dosyasını düzenlemenize olanak sağlar ve işlemin tamamlanmasının ardından pwd_mkdb çalıştırıp diğer dosyaların güncellenmesini sağlarlar.

Kullanıcı Hesabının Parolasını Değiştirmek

passwd(1) ile bir kullanıcı hesabının parolası değiştirilebilir. Kullanıcı kendi parolasını root ise tüm parolaları değiştirebilir. Kullanıcı komutu doğrudan çalıştırıp kendi parolasını değiştirebilir. Bu işlem için öncelikle kullanıcının eski parolasını yazması ardından yeni parolasını iki defa yazması gereklidir. Yeni parola ancak ve ancak bu iki defa düst üste doğru yazılırsa değişebilir. Bir kullanıcıoın hesabının parolasını değiştirmek için root olarak aşağıdaki komutun çalıştırılması gereklidir.

```
# passwd fahrettin
```

Bu aşamadan sonra kullanıcı hesabına atanacak yeni parola üst üste ardışık olarak iki defa yazılarak değiştirilir. Root kullanıcısının herhangi bir kullanıcının şifresini bilmesine gerek yoktur.

```
chpass(1) ile Kullanıcı Hesabının Düzenlenmesi
```

Kullanıcıların kendi hesaplarına ilişkin düzenleme yapmalarına olanak veren bir diğer araç ise chpass(8)'tır. Bu araç kullanıcının yetkisi kapsamında değiştirebileceği tüm bilgiyi değiştirebilir. Bunlar kabuk, ad ve soyad, adres ve telefon bilgileridir.

```
[goksin@tardis /usr/home/goksin]$ chpass goksin
#Changing user information for goksin.
Shell: /usr/local/bin/bash
Full Name: Goksin Akdeniz
Office Location:
Office Phone:
Home Phone:
Other information:
~
~
~
/etc/pw.dgbyDF: unmodified: line 1
```

Bu işlemi root hesabı ile yaparsanız kullanıcı hesabına ilişkin daha çok seçeneği düzenleyebilirsiniz.

```
tardis # chpass goksin
Changing user information for goksin.
Login: goksin
Password: $1$4d.nOPmc$uuBQy6ZL6hPQNTQef1jty
Uid [#]: 1001
Gid [# or name]: 1001
Change [month day year]:
Expire [month day year]:
Class: turkce
Home directory: /home/goksin
Shell: /usr/local/bin/bash
Full Name: Goksin Akdeniz
Office Location:
Office Phone:
Home Phone:
Other information:
~
~
~
/etc/pw.UOoZE4: unmodified: line 1
```

Burada göreceğiniz gibi kullanıcı hesabına ilişkin kullanıcı parolası da dahil olmak üzere birçok seçeneği değiştirebilirsiniz. Kullanıcının ev dizinini değiştirmeniz durumunda kullanıcı ve dizinindeki dosyalar yeni ev dizinine taşınmaz, bunları kendinizin taşıması gereklidir. Kullanıcı hesabının parolasının geçerlik süresini de değiştirebilirsiniz. Bu özellikle de kullanıcıyı parolasını sisteme ilk defa giriş yaptıktan sonra değiştirmeye zorlayacaksanız işe yarayacaktır. Bu özellikle de geçici bir süre için bir kullanıcı hesabı isteyen kullanıcılar için uygundur. Zira sistem yöneticisi olarak bu hesabın bir süre sonra kaldırılması gerektiğini unutup açık halde bırakabilirsiniz. BSD sistemler ise bunu asla unutmaz. Hesabın geçerlik süresi -expire- ile belirtilen süre sonunda otomatik olarak sonlandırılacaktır. Bu tarihin tanımlanması Ay Gün Yıl olarak yapılır. Ama tanımlamada İngilizce ay adlarının kısaltmaları kullanılır.

vipw(8)

Bir sistemde birçok kullanıcının olması durumunda tüm kullanıcı hesaplarının chpass ile teker teker düzenlenmesi zaman alıcıdır. Bunun yerine doğrudan /etc/master.passwd dosyası üzerinde vipw ile düzenleme yaparak tüm kullanıcı hesapları bir kerede düzenlenebilir. Bu özellikle de kullanıcı ev dizinlerini yeni bir diske taşımanızın söz konusu olduğu durumlarda işinize fazlasıyla yarayacaktır.

vipw işlemlerinizi gerçekleştirirken herhangi bir imla hatasına karşı yazdıklarınızı kontrol eder. Hata bulunmaz ise dosyayı kayıt eder ve ardından pwd_mkdb(8) komutunu çalıştırır. İlk bakışta vipw'nin dosyanızı kontrol etmesi ve ardından işlemi gerçekleştirmesi işlemin sorunsuz olarak tamamlanacağı anlamına gelmez. Dikkatli davranmadığınızda ciddi hatalar yapabilirsiniz. Örneğin /etc/master.passwd dosyasındaki bilgiler ile diğer dosyalardaki bilgiler arasında bir uyumsuzluk söz konusu ise /etc/master.passwd dosyasının doğru olduğu kabul edilir. Bir kullanıcı hesabının grubunun /etc/groups

içinde kullanıcı hesabının birincil grubunun tanımlanmamış olmasına karşın /etc/master.passwd dosyasında tanımlı olması durumunda uygulamalar bu bilgiyi esas alırlar. Bu durum sorunlara yol açabilir. Bu nedenle /etc/master.passwd dosyasını vipw ile düzenliyorsanız bu dosyanın yapısını da anlamak durumundasınız demektir. /etc/master.passwd dosyasındaki her satırda on adet alan bulunur. Bu alanlar şu sırayla yer alır:

```
KullanıcıAdı:ŞifrelenmişParola:KullanıcıID:KullanıcıSınıfı:Par
olaZamanAşımı:HesapZamanAşımı:KişiselBilgiler:KullanıcıEvDizin
i:KullanıcıKabuğu
```

Kullanıcı Adı: Kullanıcı hesabının adıdır. Bu kurulum sırasında sistem tarafından atanmış olan hesapları, örneğin root, daemon, www gibi ve sonradan kurulmuş olan çeşitli uygulamalara ait olan kullanıcı hesaplarını, örneğin cups, polkit, haldaemon gibi, barındırır.

Şifrelenmiş Parola: Normal kullanıcı hesaplarının parolaları şifrelenmiş olarak saklanır. Sistem kullanıcılarının ise şifresi bulunmaz.

Kullanıcı - UserID – ID: Her kullanıcıya atanan ve sade kendisine özel olan değerdir.

Grup ID: Bu kullanıcıların üyesi oldukları birincil grubu tanımlar. Kullanıcı adı ve ID değeri aynıdır.

Kullanıcı sınıfı: Bu alan /etc/login.conf dosyasında tanımlanan sınıflardan birisidir.

Parola Zaman Aşımı: Parolanın geçerli olduğu son kullanma tarihidir. Burada ise AyGünYIl olarak değil saniye bazında tanımlıdır. Bu saniye bazlı tarih ataması için son kullanma tarihi kullanılarak değeri aşağıdaki gibi belirlenebilir.

```
# date -j 201106w00000 '+%s'
```

Bu komutun çıktısı hesabın 20 Haziran 2010 tarihinde gece yarısı sona ereceğini belirlemek için süreyi saniye olarak döndürür. Değer ilgili alana olduğu gibi yazılmalıdır.

Kişisel Bilgiler (Personal Data) Kullanıcının gerçek adı, telefon numarası vb bilgiler burada yer alır. UNIX'in geçmişi ile ilgilidir ve tarihsel nedenlerden ötürü bugüne dek gelmiştir. Bu alandaki bilgiler virgül ile ayrılır, iki nokta üst üste kullanmayın!

Kullanıcı Ev Dizini: Kullanıcının ev dizinin yolunu belirtir. Bu değer sistem yapılandırmasında tanımlanan varsayılan değerlerdir. Bu alan boş olduğunda kullanıcılar sisteme giriş yapabilirler ama bir ev dizinleri olmaz. Bunun yaratabileceği sakıncalar /etc/login.conf dosyasında düzenlenerek önlenebilir.

Kullanıcı kabuğu: Kullanıcı hesabına atanmış olan kabuk ortamıdır. Eğer bu değer boş ise varsayılan kabuk /bin/sh'dir.

Bu alanlardaki bilgilerdeki bir hatanın tek nedeni sistem yöneticisinin veya root hesabını kullanan kişinin bilgisizliği veya dikkatsizliğidir. Bir hata ciddi sorunlara yol açabilir.

Kullanıcı Hesaplarını Kaldırmak rmuser(8)

Kullanıcı hesabını kaldırmak için kullanılan rmuser kullanıcı ev dizinlerindeki kişisel dosyalar da dahil olmak üzere bir kullanıcı hesabını sistemden kaldırır.

Tek Komutla Hesap Yönetimi: pw(8)

pw, diğer kullanıcı hesapları yönetimi araçlarına göre güçlüdür. Diğerlerinin aksine pw ile kullanılan seçenekler ile bir tek satırlık komut ile kullanıcı hesabını istediğiniz gibi oluşturabilirsiniz. Bu araç günlük

kullanımda yaygın olarak kullanılmaz ama bazı hesapların geçici olarak dondurulması vb. işlemler için idealdir. Kullanıcı hesabı geçerli olmakla birlikte pw ile dondurulan bir hesaba herhangi bir kullanıcı giriş yapamaz. Hesabın ancak sistem yöneticisi tarafından çözülmesi ile yeniden kullanılabilir. Bu süre içerisinde kullanıcı hesabındaki dosyalar vs. sistemde kalır.

```
# pw lock fahrettin
```

Hesap dondurulmuş oldu, şimdi yeniden çözelim

```
# pw unlock fahrettin
```

fahrettin bana çay getir, bana müteşekkir ol!

Kabuk Yapılandırması /etc/shells

Kullanıcıların kullanabilecekleri birçok kabuk ortamı bulunmaktadır. Bu kabukların hepsi farklı özelliklere sahiptir. Birçok kullanıcı alıştığı kabuğu kullanmak isteyecektir. BSD sistemlerde bazı kabuklar kurulum CD/DVD/FTP kaynaklarında hazır sunulurken bazılarının ise ports/pkgsrc üzerinden kurulması gereklidir.

Eğer pkgsrc/ports üzerinden bir kabuk kurduysanız gerekli olan kayıt /etc/shells dosyasına eklenerek kabuğun tüm kullanıcılar tarafından kullanılabilir olması sağlanır. Sistem yöneticisinin bu konuda herhangi bir işlem yapmasına gerek yoktur. Öte yandan ports/pkgsrc kullanmadan bir kabuğun kaynak koddan derleyerek kurulması söz konusu ise bu durumda kabuğun tam yolunun /etc/shells dosyasında sistem yöneticisi tarafından tanımlanması gereklidir.

BSD sisteminizde ftp daemon çalışıyorsa bir kullanıcının /etc/shells dosyasında tanımlanmayan bir kabuk ortamı ile sisteme giriş yapmasına izin vermeyecektir. FTP kullanıcılarının sadece ftp erişimine izin veriyorsanız bu durumda '/sbin/nologin'in bu kullanıcılar için geçerli olan kabuk ortamı olarak tanımlanması gereklidir. Kullanıcılara atanacak olan login.class-giriş sınıfları bu zorunluluğu ortadan kaldıracaktır.

Gelecek bölümde kullanıcı hesaplarının yapılandırılmasında kullanılan diğer yöntemler, grupların yönetimi ile diğer temel güvenlik yapılandırma araçlarını ele alacağız.

BSD - XIV

Gruplar

Gökşin Akdeniz
goksin@enixma.org

Gruplar

UNIX sistemlerin güvenlik yapısı diğer işletim sistemleri ile karşılaştırıldığında baskıcı ve geri olarak algılanır. Bunun nedeni olarak sadece ve sadece bir tane sistem yöneticisi -root- hesabının bulunması gösterilir. Diğer kullanıcıların yetkisinin olmadığı ve sistem yöneticisinin sınırlaması ile çalışmak durumunda oldukları fikri güvenlik konusunda yöneltilen eleştirilerin temel dayanaklarından ve tezlerinden birisidir. Aslında bir UNIX sistemde sistem yöneticisinin -root- diğer kullanıcılara gereksinmelerine göre verebileceği yetkiler yoktur. Bu durum bu eleştirilerin doğruluğunu kanıtlamakta gibi düşünülse de aslında durum farklıdır. Sistem yöneticisi kullanıcıları yetkilendirmek için kullanıcı hesaplarını düzenleyebilir ve bunlara ek olarak da grupları yöneterek kullanıcının gereksinim duyacağı yetkilendirmeyi sağlayabilir. Böylelikle sistemin güvenliğini tehlikeye düşürecek veya olumsuz bir takım olaylara neden olabilecek yetkilendirme eylemlerinin önüne geçilmiş olur.

Root Şifresi

Bir UNIX sistemde bazı işlemler sistem üzerinde mutlak egemenliği gerektirir. Örneğin, çekirdek, donanım sürücüleri, temel sisteme ait dosyalar, yetkilendirme vb. işlemler gibi. Bu işlemleri gerçekleştirilebilecek olan kullanıcı root'tur. Bu işlemleri gerçekleştirebilmek için sisteme normal kullanıcı hesabınız ile giriş yapıp ardından root yetkilerini alabilirsiniz veya doğrudan root olarak giriş yaparak işlemleri gerçekleştirebilirsiniz.

İkinci seçenek olan root girişi yapılamaması gereken bir işlemdir ve doğru yapılandırılmış olan bir sistemde yapılması olanaklı değildir. İlk seçenek olan root yetkilerini almak için ise kullanıcı hesabınızın özel bir grup olan "wheel" grubunun üyesi olması gereklidir. Root yetkilerini kullanmanız gereken durumlarda su(1) kullanmanız en uygun seçenektir. su ile root yetkilerini alabilirsiniz. Bu aracın her çağrılması işlemi kayıt altına alınır. Bu özelliğin nedeni ile sisteme uzaktan erişilmek durumunda su kullanılması tercih edilebilir. Kullanılması basittir.

```
[goksin@tardis /usr/home/goksin]$ su
Password:
tardis# id
uid=0(root) gid=0(wheel) groups=0(wheel),5(operator)
tardis#
```

Son olarak kullanılan id(1) komutu, su işleminin gerçekleşmesi durumunda root kullanıcı

hesabına geçtiğinizi göstermektedir. Root yetkilerine sahip olduğuna göre aslında diğer bir deyişle root olduğunuza göre bir hatanın ciddi bir yıkıma neden olacağını anımsatmaya gerek yok. Yapacağınız bir hata sisteminizi kullanılmaz duruma getirebilir. Bu durumda sabit diskinizi formatlayıp yeniden bir sistem kurmak, hatanızı düzeltmeye çalışmaktan daha kolay olabilir. Hatayı yapan siz de olmayabilirsiniz. Eğer root şifresini diğer kullanıcılardan biri veya daha fazlası ile paylaşıyorsanız sistemin kısa zamanda kullanılamaz duruma geleceğinden emin olabilirsiniz.

Bu nedenle root şifresini sadece ve sadece sistemi kuran ve yapılandıran kişi olarak sizin bilmeniz ve diğer kullanıcılarla şifreyi paylaşmamanız zorunludur. Bunun yanında sisteme doğrudan erişebilen bir kullanıcı BSD terminolojisi ile söylersek sistem konsoluna erişebilen bir kişi root şifresini biliyorsa istediğini yapabilir. Öta yandan fiziksel erişimi olmayan bir kişi bu şifreyi bilse de kullanıcı wheel grubunun üyesi değilse sadece şifreyi bilmekten öteye gidemez. İstediği kadar su deneyebilir ama işe yaramaz.

Root şifresini sadece sistem yöneticisi bilmelidir. Diğer kullanıcıların ise bilmesine gerek yoktur. O halde root hesabının üzerinde neden duruyoruz sorusu akla gelebilir. Sistemin temel yapılandırmasını yapacak olan kullanıcı root kullanıcısıdır. Doğru yapılandırılmış bir sistemde root şifresinin kullanılmasına pek gerek duyulmaz. Sistemde tanımlı olan grupları yapılandırıp ve gerekiyorsa ek gruplar ve kullanıcılar ekleyerek root yetkilerine gerek duyulmadan işleri yürütebilirsiniz. Böylelikle root şifresinin kullanımı azaltılır. Bunlara ek olarak da sudo kullanılarak (/usr/ports/security/sudo) gerekli olan sistem yönetimi ve bakım işlemlerini gerçekleştirebilirsiniz

Gruplar

UNIX ve benzeri sistemlerde kullanıcılar gruplara ayrılır. Her bir grup benzer işlemleri yapan kullanıcıları kapsar. Bir kullanıcı sadece bir tek grubun değil aynı zamanda birden çok grubun üyesi olabilir. Sistemi yapılandıran sistem yöneticisi grupları kullanarak bazı yönetimsel işleri bu grupların üyesi olan kullanıcıların gerçekleştirmesine izin verebilir. Örneğin database grubu sadece veritabanları ile ilgili işlemleri yürütecek olan kullanıcılar için oluşturulabilir ve veritabanı yöneticilerinin bu grubun üyesi olan kullanıcı hesaplarını kullanarak sistem üzerindeki veri tabanlarını yönetmesi ve yapılandırması sağlanabilir. Benzer olarak eposta, webmasters vb. gruplar ile ilgili hizmetlerin yapılandırılması ve yönetimi gerektiği gibi yapılabilir. Bu yapılandırma sistem yönetimi açısından kullanışlı bir yöntemdir ama bir sistem yöneticisi tarafından uygulanmaz ve yetki tek elde toplanır sistem yöneticisi de aslında işi olmayan işleri yapmak durumunda kalır. Ya da tersine olarak root şifresini herkese verir ve düzenli olarak iki haftada bir "bakım" yaparak durumu idare etmek durumunda kalır. Bu "bakım" işlemi aslında sabit diski formatlayıp yeniden sistem kurmaktan öte bir iş değildir.

Kendi kullanıcı hesabıma id ile göz atacak olursak aşağıdaki yapılandırmayı görebilirsiniz.

```
[goksin@tardis /usr/home/goksin]$ id
uid=1001(goksin) gid=1001(goksin)
groups=1001(goksin),0(wheel),5(operator)
[goksin@tardis /usr/home/goksin]$
```

Kullanıcı adım goksin, kullanıcı id 1001, Birincil grubum -gid- goksin. Bu standart kullanıcı hesaplarının yapılandırılmasında kullanılan grup üyelikleridir. Bunların dışında ayrıca wheel, operator gruplarına da üyeyim. wheel grubu üyeleri su kullanarak root şifresine sahip iseler root yetkilerini kullanabilirler. Kullanıcı olarak üyesi olduğum gruplara tanınan okuma, yazma ve çalıştırma yetkilerini ben de kullanabilirim. Gruplara ilişkin bilgiler /etc/group dosyasında bulunur.

/etc/group dosyası sistemde tanımlı olan tüm gruplara ilişkin bilgileri barındırır. Bu bilgi içinde ise kullanıcıların birincil gruplarına ilişkin bilgi

yer almaz ancak dosyanın içeriğine baktığınızda kullanıcıların üyesi oldukları grupları görebilirsiniz. Kullanıcının üyesi olduğu birinci gruba ilişkin bilgi ise /etc/master.passwd dosyasında yer alır. /etc/groups dosyasının içeriğine bakarsanız aşağıdakine benzer satırlar göreceksiniz.

```
wheel:*:0:root,goksin
```

Satırdaki ilk girdi grubun adıdır. İzleyen girdiler ise ":" ile ayrılmıştır. Dosyanın içeriğinde * ile görülen alan ise eski programların kullandığı grup şifresidir. Bu gün için grup şifresi kullanılmadığı için bu alana gereksinim yoktur ancak geriye doğru uyumluluk sağlanması için bu alana "*" yazılmıştır. Üçüncü alan ise group id yani gid değeridir. wheel grubu 0 gid değerine sahiptir. UNIX ve türevi sistemlerde grubun alabileceği en büyük GID değeri 65535'dir. Son olarak gelen kısım ise grubun üyesi olan kullanıcıların isimlerinin yer aldığı alandır ve bu alanda kullanıcı hesapları virgül ile ayrılarak yazılır.

Grup Üyeliklerini Düzenlemek

Bir kullanıcıyı bir grubun üyesi yapmak için kullanıcı adını ilgili grubun üyeleri arasına eklemek yeterlidir. örneğin sistemdeki diğer kullanıcı olan sema hesabını wheel gruna eklemk için kullanıcı hesabını ilgili satırın sonuna eklemem yeterlidir.

```
wheel:*:0:root,goksin, sema
```

Grupların Yönetimi

Yeni bir grup eklemek isterseniz, grup adı ile grup ID değerine gereksinim duyacaksınız. Bu değerleri atadığınızda grup sisteme eklenmiş olur. Gruba bir kullanıcıyı üye olarak atamanıza da gerek yoktur. Bazı programlar bir grubun üyesi olarak çalışır ve BSD sistemler bu programları kontrol etmek için nasıl kullanıcıları kontrol etmek için grupları kullanıyorsa aynı şekilde bu programlar için de grupları kullanırlar. ;)

Grup numaraları 0 ile 65535 arasında bir değere sahiptir. 0 ile 1000 arasındaki değerler sisteme ayrılmıştır. Bu aralıktaki değerlere sahip olan gruplar sistem programlarına aittir. Kullanıcı hesapları ise 1001'den başlayarak numaralandırılır. Bazı gruplar ise 65535'den başlayarak azalarak numaralandırılır.

Root Hesabının Kullanımını Gruplar İle Azaltmak

Root şifresinin birden çok kullanıcı tarafından bilinmesi ciddi güvenlik sorunlarına neden olabilir. Bu nedenle birçok sistem yöneticisi root şifresini sistemin diğer servislerinden sorumlu olan diğer yöneticiler ile paylaşmayı istemez. Var olan grup yapılandırması ile gereken işlerin yapılması söz konusu olamayacağı için yeni gruplar sisteme eklenerek gereken kullanıcılar bu gruplara üye yapılabilir. Böylelikle de root yetkileri gerekmeden gerekli olan işlemler yapılabilir. Root şifresinin sadece sorumlu sistem yöneticisinde olması olası bir problem durumunda root şifresini bilmekten kaynaklanan potansiyel sorumlu durumuna düşmenizi de önler.

BSD sistemlerde bazı kullanıcı hesapları çeşitli programlar için ayrılır ve bu programlar tarafından kullanılırlar. Örneğin DNS sunucusu olarak kullanılan bind kendi adını taşıyan bir kullanıcı hesabını kullanır. Dolayısıyla da kullanıcı adı bind yazarak bu hesabı kullanmaya kalkışmayın. öte yandan bir saldırganın bind'i ele geçirmesi durumunda ise sadece bind hesabına tanınmış olan yetkiler dahilinde sisteme erişebilir.

Bunun dışında kullanıcıların sadece bind'ın çalışması için gerekli olan dosyaların diğer kullanıcı hesaplarınca sahiplenilmesinin de önlenmesi gereklidir. Bu ve benzeri özel uygulamalar için ayrı bir kullanıcı hesabı ile grubu oluşturmak olası bir saldırıda saldırganın DNS sunucusu tarafından kullanılan dosyaların değiştirmesini de

önleyecektir. Öte yandan program sık sık dosyalar üzerinde işlem yaparak güncellemeler yapıyorsa bu durumda uygulamaya erişim yetkisinin de verilmesi gerekir. Diğer dosyalara erişmesi gerekmiyorsa bu yetkiyi vermemek gerekir. Herhangi bir uygulamanın kendi yapılandırma dosyasını değiştirmesi söz konusu değildir ve bu dosyaya erişim yetkisi vermeye de gerek yoktur.

Yönetici Grupları Oluşturmak

Bir dosyanın sahibinin sadece ve sadece ilgili kullanıcı olmasını sağlamanın yolu adduser ile dosyaların sahibi olan bir kullanıcı oluşturmak ve hesabın birincil grubunun da dosyalar ile aynı grupta olmasını sağlamaktır. Bu yöntem ile ilgili servisi yönetecek olan yönetici uzaktan sisteme erişerek sadece gerekli olan dosyaları ve uygulamaları yönetebilir. Root şifresini kullanmasına gerek yoktur. Bu uygulama için söz konusu servisi yönetecek olan kullanıcı adının ne olduğu önemli değildir ama sistem yöneticisi olarak kolay anımsayabileceğiniz bir isim seçmenizde yarar var.

İlgili servisi yönetecek olan kullanıcı hesabının sisteme giriş yapmasını önlemek için kabuk olarak /sbin/nologin atayın. Böylelikle herhangi birisi bu hesabı kullanarak sistem giriş yapamaz. Eğer istenirse bu yönetici hesapları için özel UID ve GID değerleri belirlenerek kullanılabilir. Örneğin bind için UID ve GID değerleri 53'tür. DNS sunucusundan sorumlu sistem yöneticisi için yaratacağımız hesap bind'ı anımsatacak şekilde GID ve UID değeri olarak 10053 kullanılabilir. Bu yaklaşım sözkonusu hesabın ilişkili olduğu servise ilişkin bir ipucudur. Bunun gibi veya sizin tercih ettiğini bir yapılandırma kullanmak işlerinizi kolaylaştıracaktır. Özellikle de bu yapılandırma tutarlı bir şekilde sadece ve sadece root şifresini bilen kişi yani siz tarafından yapıldığı sürece.

Servisi yönetecek olan yönetici hesabınızın herhangi bir diğer grubun üyesi olmaması zorunludur. Bu yeni oluşturacağınız hesabı asla wheel grubuna üye yapmayın. Kullanıcı hesaplarının tamamı bir ev dizinine gerek duyar ama bu hesabın ev dizini /nonexistent olarak atanabilir. Bunun ardından hesabın adduser tarafından kapatılmasını sağlayın. Bunu yapmanızın gereği sisteme kullanıcının uzaktan erişecek olmasıdır. Dolayısıyla kabuk vermedik, hesabın ev dizini yok ama ek bir önlem olarak da hesabın doğrudan fiziksel olarak erişilmesini engellemek yerinde olur. Hesabı ve grubu oluşturduktan sonra gerekli dosyaları bu gruba ve kullanıcıya atayabilirsiniz. Dosyaların gruplarını ve sahiplerini chown ve chgrp ile değiştirebilirsiniz. Bu işlemin ardından ls ile sahip ve gruplar ile diğer üyelerin erişim izinlerini kontrol edin. Birçok sistem yöneticisi dosyanın sahibi ile grubunun izinlerine bakar ama diğerlerini ihmal eder. Bir dosyaya sadece ve sadece erişmesi gereken kullanıcılar ve gruplar erişebilmelidir. Bu erişim de gerektiği kadar olmalı, ne eksik ne de fazla.

```
# chown kullanıcı:grup dosya
```

DNS servisinden sorumlu yöneticiler genelde sunucuyu yönetmek için mutlaka root parolasının gerektiğini iddia eder halbuki bu rndc(8) ile yapılabilir. Diğer işlemler için cron veya sudo(8) kullanılarak tüm işlemler yapılabilir.

Standart grup Yapılandırması

BSD sistemlerde standart olarak birçok grup bulunur. Bu grupların büyük kısmı sistem tarafından kullanılır ve sistem yöneticisi için günlük çalışmasının bir parçası olsalar da çok da önemli değildirler. Sistem yöneticisi olarak kendi gruplarınızı ekleyebilirsiniz ve bu sistemin yönetimini kolaylaştırır. Aşağıdaki gruplar ise kullandığım FreeBSD sistemde bulunan gruplardır. Bu gruplardan bazıları standart olarak bulunurken bazıları ise sonradan kurulan uygulamalar tarafından oluşturulmuş olan gruplardır.

wheel	root şifresini kullanabilen kullanıcılar
daemon	Birçok sistem servisi tarafından örneğin yazdırma gibi

	kullanılan grup
kmem	Kernel belleğine erişebilen uygulamaların grubudur, örneğin fstat(1), netstat(1) vb.
sys	Bir diğer sistem grubu
tty	Terminallere yazabilen wall(1) gibi uygulamaların grubudur.
operator	Sürücülere erişebilen grup. genel olarak yedekleme amaçlı olarak kullanılır.
mail:	E-posta sisteminin grubudur.
bin	Sistem genelindeki derlenmiş dosyalar ve programların grubudur.
news	Eğer kurulu ise usenet servisinin üyesi olduğu gruptur.
man	Kılavuz sayfalarının grubudur.
games	Oyunların grubudur.
ftp	ftp sunucu ve istemcilerinin grubu
staff	Sistem yöneticilerinin grubudur.
sshd:	SSH sunucusunun sahibidir.
smmsp	Sendmail Submission User grubu
mailnull	Sendmail için varsayılan grup
guest:	Konuk hesapları grubudur. (bunu hiç kullanmadım)
bind	DNS sunucusunun grubudur.
proxy	PF'in FTP proxy grubudur
authpf	PF yetkilendirme grubu
_pflogd	PF logger grubu
_dhcp	DHCP istemci işlemleri grubu
uucp	Unix-to-Unix Copy Protokolü programlarının grubudur.
dialer	Modemler ve tip(1) seri portlara erişebilen gruptur.
network	Network programlarının grubudur. Örneğin ppp(8) bu gruptadır.
audit	audit(8) bilgilerine erişebilenlerin üye olduğu gruptur.
www	Web sunucu yazılımlarının grubudur. Sunucuda kişisel vs dosyaların grubu değildir.!
nogroup	Herhangi bir yetkisi olmayan gruptur.
nobody	Nobody adlı herhangi bir yetkisi olmayan kullanıcı hesabının grubudur.
messagebus	Sistem mesajlaşma yolu grubudur.
polkit	Polkit daemon grubudur
haldaemon	HAL daemon grubdur.
cyrus	Cyrus IMAP sunucusunun grubu
avahi	avahi daemon grubu

Kullanıcı Hesapları Üzerinde Güvenlik Uygulamaları

Kullanıcıların kullanabilecekleri sistem kaynaklarını sınırlandırabiliriz. Önceki bölümlerde değindiğimiz cron ve at vb. uygulamalardaki kısıtlamaların güvenlik gerekçesi ile sınırlandırılması gerekebilir. Sınırlama ise sadece cron ve at ile kısıtlı değildir. Kullanıcının ne kadar bellek ve işlemci gücü kullanabileceği de kesin olarak belirlenebilir. Bugünkü bilgisayarlarda masaüstünde dört çekirdekli 2.6 GHz'lik işlemciler ve 8GB RAM kullanabiliyorken buna pek gereksinim duymayabilirsiniz. Ancak sistem ne kadar güçlü olursa olsun yüzlerce kullanıcının eriştiği ve kullandığı bir sistemde sistem kaynaklarının kullanımını sınırlandırmak yerinde olacaktır. Bu sınırlamada ilk adım kullanıcıların nereden sisteme giriş yapacaklarıdır.

Sisteme Girişleri (login) Sınırlandırmak

BSD sistemlerde kullanıcının sisteme giriş yapmak istemesi durumunda /etc/login.access dosyası kontrol edilir ve dosyada tanımlanan kuralların gereği yapılır. Bu dosyada standart olarak herhangi bir kural yer almaz. Diğer bir deyişle tüm kullanıcılar için bir sınırlama söz konusu değildir. Geçerli bir kullanıcı adı ve şifresi olan kullanıcı sisteme giriş yapabilir. /etc/login.access dosyasında kural iki nokta üst üste konularak ayrılmış üç alandan oluşan bir satırdan ibarettir. İlk alanda +/- yer alır. + girişe izin verilir, - ise girişe izin verilmeyen anlamına gelir. İkinci alan ise kullanıcılar veya grupların tanımlandığı alandır. Son olan üçüncü bölüm ise giriş için kullanılan bağlanma yönetimini tanımlar.

Daha ayrıntılı bir tanımlama yapmak için ALL -hepsi- ve ALL EXCEPT -hepsi ama .... hariç- tanımlamasını kullanarak daha basit ve ayrıntılı kurallar tanımlayabiliriz. Bu kurallar kontrol edilirken ilk eşleşen kural geçerlidir ilkesi uygulanır. Bir kullanıcı için birden çok kural yazabilirsiniz ama kurallar listesinde eylem ile eşleşen ilk kural uygulanır ve diğer kurallar dikkate alınmaz. Burada bir hata varmış gibi görünebilir ancak sisteme girişleri ayrıntılı olarak yapılandırırken farklı erişim kanalları için farklı politikalar izlenebilir. Bu nedenle ilgili eylem ile eşleşen ilk kuralın uygulanması ilkesi kural yazımını basitleştirir ve karmaşıklığın neden olacağı hataları ortadan kaldırır. Örneğin sistem konsolu - kasaya bağlı olan fiziksel klavyeniz- üzerinde giriş yapabilecek olan kullanıcıların giriş yapmasına izin verelim ama sadece wheel grubunda olan -asıl sistem yöneticisi giriş yapabilsin. wheel grubunun giriş kuralı aşağıdaki gibi olacaktır.

```
+:wheel:console
```

Bu kural sadece wheel grubunda yer alan kullanıcılara izin vermez. Zira diğer kullanıcıların sistem giriş yapmasını önleyen bir kural tanımlı değildir. Bu durumda geçerli bir kullanıcı adı ve şifresi olan tüm kullanıcılar sisteme giriş yapabilir. İstediğimiz ise bu değildi. Bu durumda diğer kullanıcıların sisteme giriş yapmasını önlemek için ek bir kurala gereksinim duyuyoruz demektir.

```
+:wheel: console
-:ALL:console
```

Bu yeni kuralda wheel grubundaki kullanıcılar sistem konsolu üzerinden sisteme giriş yapabilirler. Diğerleri ise kullanıcı adı ve şifreleri geçerli olsa bile giriş yapamazlar. İstediğimi yazdım ama kısaltmak için ALL EXCEPT kullanarak bunu tek satıra indirgesem daha iyi olur.

```
-:ALL EXCEPT wheel: console
```

Bu kural ile wheel grubu dışındaki tüm kullanıcı girişleri engellenmiş oldu. Bu kural tanımlama biçimi sistem yöneticisinin olası bir hatası sonucunda olabilecek bir zayıflığı ortadan kaldıracaktır. Yazılı olan tek kural bu olduğunda basitçe tüm kullanıcı girişlerini kapadık ama wheel grubu hariç diyoruz. Son alanda console dışında diğer protokoller, alan adları v.b tanımlayarak erişim yetkilerini sınırlandırabiliriz.

Sistem İsimleri-Hostnames

Sistem izinlerinin çözümlenmesi için DNS ve hosts dosyasında gerekli tanımlamaların yapılması gereklidir. Bir alan adı sunucusu servisi kullanmak istemeyebilirsiniz. Zira olası bir saldırıda servisin devre dışı kalması durumunda sisteminiz gerekli kontrolü yapamayacaktır ve gelen tüm istekleri red edecektir. DNS düzelince işler yoluna girebilir ama o zamana dek eliniz kolunuz bağlanmış olacaktır. Ayrıca saldırganlar ip adreslerini değiştirerek de erişebilir ve bu durumda sistem bağlantı isteğini kabul edecektir. Yine de aşağıdakine benzer bir kural tanımlayabilirsiniz.

```
-:ALL EXCEPT wheel:filanca.felanca.net
```

Söz konusu sistemden gelen çağrılar yanıtlanır ve girişe izin verilir ama diğerlerinin tamamı red edilecektir.

Sistem Adresleri ve Ağlar

Yukarıdaki kuralı sistem adresi olmadan doğrudan IP adresini vererek tanımlayabiliriz. Bu özellikle de yerel ağ üzerinden erişecekseniz işinizi çok kolaylaştıracaktır ve sadece sizin tanımladığınız IP'ye sahip olan bilgisayardan gelen erişim istekleri kabul edilecektir. Öte yandan DNS sorunlarının neden olacağı sıkıntıları en başından önlersiniz.

```
-:ALL EXCEPT wheel:192.168.7.8
```

Öte yandan ağ üzerindeki tüm bilgisayarlardan erişim izni vermek için IP adresini aşağıdaki gibi tanımlayabilirsiniz. 192.168.7.0-255 arası

tüm bilgisayarlardan erişebilirsiniz.

```
-:ALL EXCEPT wheel:192.168.1.
```

LOCAL Degişkeni

Erişim kısıtlamalarını tanımlarken kullanılan değişkenler içerisinde en zor olanı LOCAL'dır. LOCAL herhangi bir sunucu ismini eğer içerisinde nokta bulunmuyorsa yerel ağadaki yani aynı alan adına sahip bir bilgisayarmış gibi tanımlar. Bu özellikle de reverse DNS ile yapılan çözümlemelerde işe yarar. Uzaktan bağlanmak istediğinizde internette görünen isminiz servis sağlayıcının alan adı ile tanımlanır ve bu durumda uzaktaki sisteme bağlanamazsınız. LOCAL sizin yerel ağdaki veya aynı alan adına sahip bir sistem olmadığınız için red edilmenize neden olur. Kullandığım dizüstü bilgisayarımın adı droideka.elkotek olsa da kablosuz ağlar üzerinden herhangi bir yerden internet erişimi ile sisteme ulaştığımda, LOCAL kullanılan kural benim evdeki sunucularıma erişmeme izin vermeyecektir. Her ne kadar tüm sistemlerin adları elkotek alan adına sahip olsalar da...

Bu nedenle LOCAL eğer ağ dışından erişecek iseniz işiniz görmeyecektir. Yerel ağ tanımlamasını kolaylaştırır ama bir o kadar da sorun yaratır.

ALL ve ALL EXCEPT Degişkenleri

ALL -herşey- eşleştirir. ve ALL EXCEPT ise tanımladığınız haricindeki herşeyi eşleştirir. Bu değişkenler erişim kısıtlaması tanımlarken en sık kullanacağınız değişkenlerdir. Özellikle de yerel ağ üzerindeki kendi kontrolünüzde olan güvenli sistemlerden erişecek iseniz aşağıdakine benzer bir kural tanımlayarak sistemlere erişimi sınırlamanız çok kolay olacaktır.

```
-:ALL EXCEPT wheel:ALL EXCEPT 192.168.7.8 192.168.7.10
```

Bu kuralların amacı sistemlerin işleyişinde kontrolün tamamen sizin elinizde olmasını sağlamaktır. BSD sisteminizle DNS, FTP, WEB vb. hizmetleri sunarken sadece sistem yöneticilerinin sistem giriş yapmasına izin vereceksenız tek satırlık bir erişim kuralı yazarak bunu sağlayabilirsiniz. Bu kural ile sistem yöneticilerinin sisteme uzaktan bağlanmasını ama diğer kullanıcıların erişmesini engelleyebilirsiniz.

```
-:ALL EXCEPT wheel:ALL
```

Bu kural ile sadece sistem yöneticilerinin sisteme giriş yapmasına izin verebilirsiniz. Diğer kullanıcıların giriş yapmasına olanak yoktur. Ancak sadece sistem yöneticisinin sisteme fiziksel erişim yetkisi vermek de olası bir sorunda sistem yöneticisinin sistem konsolu üzerinden giriş yapmasını önler. Bu nedenle ek bir kural olarak sistem yöneticisinin giriş yapabilmesine de izin vermek gereklidir. Birinci kural sistem yöneticisi için tanımlama yaparken diğer kural dns ve web sayfalarının düzenlenmesinden sorumlu olan webmaster gruplarının sisteme uzaktan giriş yapabilmesine izin vermektedir.

```
-:ALL EXCEPT wheel:console
-:ALL EXCEPT wheel dns webmasters:ALL
```

Kaynak Kullanımını Kısıtlamak

Sisteme girişleri daha ayrıntılı olarak kontrol etmek için login classes -giriş sınıfları- kullanılır. Yapılandırması /etc/login.conf dosyası ile gerçekleştirilir. Bu dosya kullanıcılar için gerekli olan kaynakları ve bilgileri tanımlar. Bir sınıfa ilişkin yapılandırmayı değiştirmeniz durumunda o sınıftaki tüm kullanıcılar bir sonraki girişlerinde yeni yapılandırmaya tabi olurlar. Sistem kullanıcı eklenirken sınıf ataması yapılır. Sonradan değiştirmek için ise chpass(1) kullanılabilir.

Sınıf Tanımlamaları

login.conf dosyası standart sınıf tanımlaması ile başlar. Bu sınıf

tanımlaması eğer bir kullanıcı hesabı için sınıf atanmamış ise varsayılan sınıf olarak kullanılır. Yapılandırma gereği kullanıcıya standart sınıf sınırsız kaynak kullanımı sunar. Özellikle de az sayıda kullanıcı tarafından kullanılan uygulamaları barındıran sistemler için bu yapılandırma uygundur. Eğer bu sizin gereksinmelerinize yetiyorsa olduğu gibi bırakın, Yetersiz ise okumaya devam edin.

Bir sınıf tanımlaması kullanıcının erişebileceği sistem kaynaklarını, kota sınırlarını ve ortamı tanımlayan değişkenlerden oluşur. Sınıf tanımlamasında kullanılan değişkenlerin atanması iki nokta üst üste kullanılarak yapılır. \ karakteri ise sınıf tanımlamasının sonraki satırda devam ettiğini belirtir. Aşağıdaki standart sınıf olan default sınıfının tanımlanması görülüyor.

```
default:\
        :passwd_format=md5:\
        :copyright=/etc/COPYRIGHT:\
        :welcome=/etc/motd:\
:setenv=MAIL=/var/mail/$,BLOCKSIZE=K,FTP_PASSIVE_MODE=YES:\
        :path=/sbin /bin /usr/sbin /usr/bin /usr/games
/usr/local/sbin /usr/local/bin ~/bin:\
        :nologin=/var/run/nologin:\
        :cputime=unlimited:\
        :datasize=unlimited:\
        :stacksize=unlimited:\
        :memorylocked=unlimited:\
        :memoryuse=unlimited:\
        :filesize=unlimited:\
        :coredumpsize=unlimited:\
        :openfiles=unlimited:\
        :maxproc=unlimited:\
        :sbsize=unlimited:\
        :vmemoryuse=unlimited:\
        :swapuse=unlimited:\
        :pseudoterminals=unlimited:\
        :priority=0:\
        :ignoretime@:\
        :umask=022:
```

Yukarıdaki sınıf tanımlamasında Sistem kaynakları kullanımında bir sınırlama bulunmamaktadır. Sınıf adı default olarak tanımlıdır ve bu sınıfa ilişkin tüm değişkenler izleyen satırlarda belirtilmiştir. Şifrelerin saklanması için MD5 kullanıldığını, girişte yayınlanan -motto of the day-motd- vs bilgileri tanımlamaktadır. Kullanıcıları yine sizin tanımladığını bir diğer sınıfa atayabilirsiniz.

login.conf dosyasında tanımlı olan tüm değişkenler için bir değer atanması söz konusu değildir ama hesap ile ilişkili olan sınıf için belirlenen yapılandırma ile kullanıcı hesabının özelliklerini değiştirir. Örneğin requirehome değişkeni sınıf tanımında yer alırsa geçerlidir. Bu durumda kullanıcının sistemde tanımlı olan geçerli bir ev dizini bulunması zorunludur. Aksi halde giriş yapamaz. Aşağıda kullanımını görüyorsunuz.

```
      :requirehome:\
```

login.conf dosyasında gereken düzenlemeleri yaptıktan sonra login veri tabının güncellenmesini sağlamak için dosyanın en başında belirtildiği üzere aşağıdaki komutun çalıştırılması gereklidir.

```
# cap_mkdb /etc/login.conf
```

Bu komut /etc/login.conf.db veritabanını günceller. /etc/spwd.db gibi hızlı bir şekilde sorgulamaların gerçekleştirilmesini sağlar. /etc/login.conf dosyası birçok farklı sınıf örneği barındırır. Kendi kullanıcı sınıflarınızı oluştururken bu sınıfları esas alabilirsiniz. Bu örnek sınıflar belli sınırlamaların nasıl yapılabileceği ve kullanıcı hesabının nasıl yapılanacağını kavramanız açısından faydalıdır. login.conf(5) kılavuz sayfasında bu konuda daha ayrıntılı bilgi verilmektedir.

Kaynak Kullanımı Kısıtlaması

Sistem kaynaklarını kısıtlamak bir kullanıcı hesabının sistem kaynaklarının ne kadarını kullanabileceğini belirlemenize olanak verir.

Sınırsız kaynak kullanımı sistemdeki bir kullanıcının tüm kaynakları tüketebileceği anlamına gelir ve kritik sistemlerde izin verilmeyen bir durumdur. Bu nedenle kaynak kullanımını kısıtlayarak sistemde bulunan diğer kullanıcılarında sistem kaynaklarından adil bir şekilde yararlanmasını sağlayabilirsiniz. Aşağıda login.conf dosyasında kullanabileceğiniz sistem kaynaklarının kullanımını belirleyen değişkenleri görebilirsiniz.

cputime	Bir sürecin max kullanabileceği işlemci zamanı
filesize	Bir dosyanın en büyük boyutu
datasize	Bir süreç tarafından kullanılabilecek olan verinin büyüklüğü
stacksize	Bir süreç tarafından kullanılabilecek olan stack memory değeri
coredumpsize	Bir çökme anında çöken uygulamanın döndürebileceği max bellek dökümünün boyutu
memoryuse	Bir sürecin kullanabileceği bellek miktarı
maxproc	Kullanıcının çalıştırabileceği max süreç sayısı
openfiles	Bir sürecin açabileceği max dosya sayısı
sbsize	Bir kullanıcının çalıştırdığı uygulamanın kullanabileceği socket buffer'ın max büyüklüğü

Sistem kaynaklarını sınırlandırırken her süreç için kullanılabilecek olan bellek miktarı vs. tanımlayabilirsiniz. Ancak bu durumda bir sınıf için kullanıcı süreçlerine en çok 20 Mb bellek ayırdığını varsayalım. Bu sınıftaki kullanıcıya ait olan 40 adet süreciniz olduğunda bu sınıftaki bir kullanıcı tek başına 800MB RAM tüketecektir. Bu nedenle bu tür sınırlamalara giderken dikkatli olun. Kolaylıkla sistem kaynaklarınızı bir kullanıcı ile tüketebilirsiniz.

Bir kullanıcının kullanabileceği sistem kaynağını sınırlamak dışında bir diğer seçeneğiniz ise kullanıcının herhangi bir anda kullanabileceği sistem kaynakları için iki ayrı sınır tanımlamaktır. Bu sınırlama herhangi bir andaki kaynak kullanımını maxproc-cur ile tanımlarken yine maxproc-max ile de üst sınırı tanımlamaktır. maxproc-cur tanımlaması birden çok kullanıcının sistem kaynaklarını paylaşması durumunda yararlı olacaktır. maxproc-cur ile tanımlanan sınırı bir kullanıcının aşması durumunda sınıra yaklaştığını anımsatılacaktır. Kullanıcı macproc-cure ile tanımlanan değeri aşabilir ama maxproc-max ile tanımlanan sınırı aşamaz. BSD sistemlerde eğer maxproc-cur veya maxproc-max tanımlamadan doğrudan maxproc tanımlarsanız bu üst sınır olarak değerlendirilir. Bir sınıf için cur ve max sınırlamaları aşağıdaki gibi tanımlanabilir:

```
        :maxproc-cur: 100:\
        :maxproc-max: 200:\
```

Sınıf ile Ortam Degişkenlerini Tanımlamak

Kullanıcının çalışacağı ortamı /etc/login.conf dosyasında tanımlayabilirsiniz. Kullanıcının kullandığı kabuk ortamının yapılandırma dosyalarını örneğin .cshrc veya .profile kullanmamış oluruz. Z kabuğu bu yapılandırma dosyalarını kullanmaz bu nedenle Z kabuğu kullanan bir kullanıcı için bu tür bir yapılandırmaya gitmek işe yaramayacaktır. Ayrıca bu işlemin tüm kullanıcılar için teker taker yapılmasına gerek kalmaz. /etc/login.conf dosyasında yapılacak olan düzenleme kullanıcın sisteme bir sonraki girişinde etkin hale gelir.

İki özel karakter tüm ortam değişkenlerince tanınır. Bunlar (~) kullanıcının ev dizini ile ($) kullanıcı adı'dır. Standart sınıf tanımlamasında bu iki karakterin kullanımını aşağıda görebilirsiniz.

```
:setenv=MAIL=/var/mail/$,BLOCKSIZE=K,FTP_PASSIVE_MODE=YES:\
        :path=/sbin /bin /usr/sbin /usr/bin /usr/games
/usr/local/sbin /usr/local/bin ~/bin:\
```

$ kullanılarak kullanıcı için MAIL değişkeni /var/mail/kullanıcı_adı olarak tanımlanmıştır. Ayrıca PATH değişkeni içinde kullanıcı ev dizininde yer alan /bin dizini de tanımlanmıştır. Aşağıda login.conf dosyasında kullanılan ortam değişkenleri görülmektedir.

hushlogin	Sisteme giriş yapıldığında herhangi bir bilgi döndürülmez.
ignorenologin	/var/run/nologin çalıştırılacak şekilde yapılandırılmış olan bir hesaba giriş yapabilir.
ftp-chroot	Kullanıcılar ftp kullanılması durumunda chroot edilen dizine geçiş yapar.
manpath	$MANPATH değişkeni için kılavuz sayfalarının bulunduğu dizini tanımlar.
nologin	Kullanıcı giriş yapamaz.
path	$PATH değişkeni ile tanımlanan dizinleri belirtir.
priority	Kullanıcının süreçleri için nice değerini tanımlar.
setenv	Ortam değişkenleri ile değerleri virgüller ile ayrılarak yazılır.
umask	umask için belirtilen değerdir ve her zaman sıfır ile başlamalıdır.
welcome	Giriş yapıldığında gösterilen hoş geldin mesajının bulunduğu dizin
shell	Giriş yapıldığında çalıştırılacak olan kabuk
term	Belirttiği terminal kullanıcıya atanan terminal ortamıdır.
timezone	$TZ değişkeninin belirttiği zaman dilimidir.

shell değişkeninde tanımlanan kabuk ortamı /etc/master.passwd dosyasındakinden farklı ise bu değişkenin tanımladığı kabuk kullanıcıya atanır. Öte yandan kullanıcının asıl kabuk ortamı da geçerli olduğu için sorun olabilir. Eğer bir kullanıcıya sistemi dar etmek istiyorsanız buraya asıl kabuğundan farklı bir kabuk atayabilirsiniz.

Sistem Girişleri ve Şifre Yönetimi

Bir giriş sınıfı tanımlayarak ortam değişkenlerini tanımlayabilirsiniz. Bundan başka kullanıcının kendi ev dizinlerinde ilgili yapılandırma dosyalarında tanımlanan ortam değişkenleri ile de bir kullanıcı hesabını da yönetebilirsiniz. Ancak sistem genelindeki bu dağınık yapılandırma dosyaları ile kullanıcıların sistem erişimlerini kontrol etmek zordur. Bunun yerine bir giriş sınıfı atayarak sisteme erişimi kontrol altına alabilirsiniz. Bunun için sınıf tanımlarkan aşağıdaki değişkenleri kullanarak kullanıcının sistem erişimini kontrol altına alabilirsiniz.

minpasswordlength

Bir şifrenin uzunluğunu belirtir. Kullanıcı şifresini değiştirdiği zaman etkin hale gelir. Uzaktan erişecek kullanıcıların SSH kullanmasını zorunlu tutmak isterseniz şifre uzunluğunu 128 karakter olarak belirleyin.

```
\:minpasswordlen=128:\
```

passwd_format

Kullanıcı şifrelerinin saklanacağı şifreleme yöntemini tanımlar. /etc/master.passwd. dosyasında saklanan şifreler burada tanımlanan şifreleme yöntemine göre şifrelenerek saklanır. Varsayılan md5'tir. Kullanıcı hesaplarını bir diğer UNIX sisteme veya Linux dağıtımına taşıyacak iseniz veya bu sistemler arasında yapılandırma dosyanızı paylaştıracak iseniz DES (des) veya blf (Blowfish) kullanabilirsiniz. DES kullanmak yerine Blowfish kullanabilirsiniz. Eğer şifrelerinizin çalınmasını ve birilerinin sisteme izinsiz olarak girmesini istiyorsanız nthash (Windows NT) kullanabilirsiniz.

mixpasswordcase

Şifrenizin büyük ve küçük harflerden karma olarak oluşturulmasını gerektirir. Aksi halde hepsi küçük harf olan şifre seçen kullanıcılara sistem mızmızlanır. Kullanıcılar sizi arayıp şikayet ederse, telefonu fişten çekin.

host.allow

Bu değerin atanması durumunda kullanıcılar rlogin ve rsh kullanabilir. Bu riske girmeyin!

host.deny

Bu değerin atanması durumunda kullanıcılar yine rlogin ve rsh kullanabilir. Bu riske girmeyin!

times.allow

Bu değişkene atanan değerler ile kullanıcıların sisteme giriş yapabildikleri zaman dilimleri tanımlanabilir. GünSaat-Saat biçiminde kullanıcı girişi tanımlanmalıdır. Günler İngilizce olarak belirtilir; Pazar-Su, Pazartesi-Mo, Salı-Tu, Çarşamba-We, Perşembe-Th, Cuma-Fr, Cumartesi-Sa. Saat ise 24 saat olarak tanımlıdır. Mesai saatleri içinde yalnızca Salı günü girişi yapabilenleri tanımlarken :times.allow=Tu8-17:\

times.deny

times.allow gibidir ama tersidir. Kullanıcı sınıfı tanımlanan zaman diliminde sisteme giriş yapamaz. Eğer kullanıcı sisteme giriş yapmış ise çıkış yapmadığı sürece bağlı kalır. times.allow ile çakışan zaman dilimleri tanımlarsanız times.deny içinde yer alan tanımlamalar geçerlidir, times.allow girdileri dikkate alınmaz.

File Flags-Dosya Özniteliklerini Degiştirmek

UNIX ve benzeri tüm sistemlerde dosya sisteminde aynı yetkilendirme sistemi kullanılır. Bir dosyasının sahibi, üyesi olduğu grup ve diğer kullanıcı hesaplarının bu dosya üzerindeki yetkileri tanımlanır. BSD sistemlerde ise bu izin sistemine ek bazı özellikler kullanılarak sistemin güvenliği pekiştirilir. Bu ek özellikler "file flags" olarak tanımlanır. "File flags" Türkçe karşılığı aslında dosya bayraklarıdır. Bayrak sözcüğünü karşılık olarak kullanmak yerine bu bayrakların ne işe yaradığını açıklayacak olan öznitelik sözcüğünü kullanmayı tercih ediyorum. Dosya öznitelikleri denildiğinde bir dosyanın yazılabilir, okunabilir veya düzenlenebilir olduğu anlaşılır. Sahip, grup ve diğerlerinin bir dosya üzerinde yaptığı bu işlemler erişim yetkileri olarak adlandırılırken dosyalarda yapılan düzenlemeler ile kopyalama, taşıma ve üzerine yazma gibi işlemleri yapılabilirliği değişir. Aslında bu özniteliklerin tamamı güvenlik ile ilgili değildir. Konumuz güvenlik olduğu için bu öznitelikleri düzenleyen komutlardan sadece güvenlikte kullanılanları ele alıyoruz. Konu ile ilgili daha ayrıntılı bilgiyi chflags(1) kılavuz sayfasından edinebilirsiniz.

Bir dosyanın özniteliklerini değiştirdiğinizde BSD sisteminin güvenlik düzeyine bağlı olarak farklı işleyeceklerdir. Güvenlik seviyesi konusundan sonra ele alacağız. Özniteliklerin düzenlenmesi ile devam ediyoruz.

sappnd

(system-level append-only) Ancak root tarafından kullanılabilir. Bu özniteliğe sahip olan dosyalara sadece sonuna ekleme yaparak yazabilirsiniz ama üzerinde değişiklik yapamazsınız. Özellikle log dosyalarının güvence altına alınması için eşsizdir. kullanıcı hesaplarındaki ".history" dosyasında kullanılırsa hesabın kontrol edilmesi ve izlnemesi kolaylaşır. Bu dosya üzerinde kullanıcılar herhangi bir işlem yapamayacaktır. Script kiddie'ler bu dosyaları /dev/null'a sembolik bağlarla bağlayarak yaptıkları işlerin kayıt altına alınmamasını sağlamaya çalışırlar bu dosya üzerinde gereken düzenlemeyi yaptığınızda yaptıkları her işlem kayıt altına alınmış olacaktır. Sistemin güvenlik seviyesi bir ve üzeri ise bu öznitelik değiştirilemez.

schg

(system-level immutable) Root tarafından atanır. Sistem genelinde herhangi bir kullanıcı bu özniteliğe sahip olan dosya üzerinde herhangi bir işlem yapamaz. Dosya sistemi bu dosya üzerinde yapılacak olan tüm işlemleri engeller. Sistem güvenlik düzeyi bir veya daha üzerinde ise bu öz nitelik değiştirilemez.

sunlnk

(system-level undeletable) Bu özniteliğin atanması durumunda dosya üzerinde her türlü değişiklik yapılabilir ama dosya silinmez. Bu

durumda güvenlik açısından bir yarar sağlamıyor gibi görünebilir ama bazı durumlarda çok işe yarar. Özellikle de bir uygulama çöktüğünde kendisine ait olan log dosyasını silmeye çalışıyorsa bu öznitelik kullanılabilir. Ancak genel olarak sık kullanılmayan bir özniteliktir. Güvenlik düzeyi bir veya üzeri ise değiştirilemez.

uappnd

(user-level append-only) Dosyanın sahibi veya root tarafından atanabilir. sappnd ile benzer şekilde çalışır. Kullanıcıların kazara kendi dosyalarını silmelerini önlemek için uygundur. Root veya dosyanın sahibi kaldırabilir. Root söz konusu dosya üzerinde her türlü işlemi yapabilir.

uchg

(user-level immutable) Dosyanın sahibi veya root tarafından atanabilir. schg ile benzer şekilde çalışır. Root veya dosyanın sahibi tarafından atanabilir ve kaldırılabilir. Sisteminizi güvenli kılmak için kullanmayacağınız bir özniteliktir. Root söz konusu dosyada her türlü işlemi yapabilir.

uunlnk

(user-level undeletable) Dosya sahibi veya root tarafından atanabilir. Bu özniteliğe sahip olan dosya silinmez. Root ise her türlü işlmei yapabilir. Kullanıcı herhangi bir anda bu öz niteliği değiştirebilir.

Öznitelikleri Listelemek

chflags(1) ile herhangib ir dosyanın öznitelikleri değiştirilebilir. Örneğin çerkirdeğe bir saldırganın kendi yazdığı modüllerin eklenmesini önlemek isteyelim. Bu durumda root olarak aşağıdaki komutu verdiğimizde çekirdek dosyaları her türlü düzenlemeden bağışık olacaktır.

```
tardis # chflags schg /boot/kernel/kernel
```

Bu komut uygulanıldığında root da dahil olmak üzere hiç bir kullanıcı çekirdek üzerinde en ufak bir işlem yapamayacaktır.

Benzer olarak bazen bin dizini altındaki dosyaların değiştirilmesini veya herhangi bir işlemin yapılmasını önlemek için aynı komut öz-yinelemeli olarak kullanılabilir.

```
tardis# chflags -R schg /bin
```

Bir dizindeki dosyaların özniteliklerini listeleyebiliriz. Bunun için aşağıdaki komut kullanılmalıdır.

```
tardis # ls -lo log
-rw-r--r--  1 root  wheel  sappnd      4955  2 Kas 10:59 log
```

Eğer herhangi bir dosyada öznitelikleri değiştirmediyseniz sorguladığınız ilgili alanda bir "-" işareti görürsünüz.

```
tardis # ls -lo
-rw-r--r--  1 root  wheel  -           4955  2 Kas 10:59
test.txt
```

Herhangi bir BSD sistem kurulum sırasında yukarıdaki örneklerde görüldüğü gibi öznitelikleri düzenlenmiş dosyalar sunmaz. Bu işlem sistem yöneticisinin tercihine bırakılmıştır. Ağa bağlı olan sürekli olarak tehditlerle karşı karşıya kalacak olan sistemlerde chflags -R schg komutunu birçok dizin üzerinde uygularız. Böylelikle herhangi bir şekilde değiştirilmiş veya içinde zararlı kod barındıran dosyaların kopyalanması gibi tehditleri ortadan kaldıracaktır. Bu önlem bir saldırganın sisteme girmesini önlemeyecektir ama saldırganın komut satırından yapmaya çalıştıklarının işe yaramayacağını garantileyecektir.

Bir dosya üzerindeki özniteliği kaldırmak isterseniz örneğin çekirdek üzerindeki bağışıklığı kaldırmak için seçtiğiniz özniteliğin önüne bir "no" getirmeniz yeterli olur.

```
tardis # chflags noschg /boot/kernel/kernel
```

Bu işlemin ise güvenlik düzeyinin "-1" olması durumunda yapılabileceğini anımsatılım.

Güvenlik Düzeyi-Securelevels

GÜvenlik düzeyleri çekirdek -kernel- ayarlarıdır. Bu ayarlar ile çekirdeğin işleyişi değişir ve bazı işlevlere izin verilmez. Güvenlik düzeyini yükselttiğinizde çekirdeğin işleyişinde değişiklikler olur. Örneğin düşük düzeylerde dosyaların öznitelikleri değiştirilebilirken üst düzeylerde bu işlemler gerçekleşmez. Sistem genelinde güvenlik düzeyinin değişmesi ile sistemin işleyişi de değişeceği için bir saldırganın başarıya ulaşması zorlaşır veya önlenebilir. BSD sistemlerde güvenlik seviyeleri kurulumda varsayılan bir ayar ile gelmez. rc.conf dosyasına aşağıdaki satırın eklenip sistemin başlatılması ile etkinleşir.

```
kern_securelevel_enable="YES"
```

Güvenlik düzeyleri sistemin işleyişini değiştirdiği için sistemin bakımı ve işletilmesinde bazı güçlükler ortaya çıkarabilir. Sistem yöneticisinin yaptığı bazı bakım işlemleri aynı zamanda sisteme izinsiz giren bir kişininde izlerini saklamak için yaptığı işlemler ile aynıdır. Örneğin bazı güvenlik düzeylerinde sistem çalışırken yeni diskler ekleyemez ve bağlayamazsınız. Ancak güvenlik seviyeleri sistem yöneticisi olarak işinizi güçleştirse de saldırganların işlerini çok daha fazla güçleştirmektedir.

Güvenlik düzeyleri beş farklı düzeyden oluşur: −1, 0, 1, 2, ve 3. En düşük güvenlik düzeyi -1 ve en yüksek düzey ise 3'tür. Güvenlik düzeyini rc.conf dosyasında kern_securelevel tanımlayarak ayarlayabileceğiniz gibi sistem çalışırken de yapılandırabilirsiniz. Sistem çalışırken güvenlik düzeyini arttırabilirsiniz ama düşüremezsiniz. Güvenlik seviyesini değiştirmek için sisteminizi tek kullanıcı -single user- kipinde başlatmanız gerekir. Diğer bir değişle yeniden başlatıp tek kullanıcı kipi açmanız gereklidir. Eğer herhangi bir anda güvenlik düzeyini düşürebiliyor olsaydınız emin olun saldırgan da bunu yapabilirdi.

Güzenlik Düzeyi: −1

Bu düzeyde çekirdek ek bir güvenlik sunmamaktadır. Eğer BSD sisteminizi yeni yeni öğreniyorsanız ve sık sık yapılandırma dosyalarınızı düzenliyorsanız bu düzeyde kalmanız gereklidir. Dosya erişim izinlerini kullanmanız gerekli güvenliği sağlayacaktır.

Güvenlik Düzeyi: 0

Bu güvenlik düzeyi sistemin başlatılması sırasında uygulanır. Ekstra bir güvenlik önlemi sunmaz. Çoklu kullanıcı kipine ulaşıldığında güvenlik seviyesi otomatik olarak "1"e yükseltilir. rc.conf dosyanızda kern_securelevel=0 ile kern_securelevel=1 kullanmak arasında bir fark olmaz. Eğer daha yüksek düzeyde çalışmayan açılış scriptleri çalıştırıyorsa sisteminiz bu düzey kullanılmalıdır.

Güvenlik Düzeyi: 1

Bu en temel güvenlik düzeyidir. Bu aşamada işler değişmeye başlar. Sistemin bu düzeyde atanan öznitelik değerleri değiştirilemez. Çekirdek modülleri, yükleyemez veya çıkartamazsınız. Programlar /dev/mem ve /dev/kmem aracığı ile dosyalara yazmazlar./dev/io'ya hiç bir şey erişemez. Bağladığınız disklere -raw disk devices- doğrudan yazılamaz ama disk üzerindeki dosyalara yazılabilir. Bu düzeyin göze çarpan özelliği schg atanan dosyaların değiştirilmesinin mümkün olmamasıdır. Bu özellikle de BSD'ye özgü dosya sistemlerinin kullanılması durumunda gözlenir.

Güvenlik Düzeyi: 2

Bu düzeyde önceki düzey olan "1" için geçerli olan her şey söz konusudur ve bunlara ek olarak iki kısıtlama daha etkinleşir:

* Diskler bağlanmış olsun veya olmasın üzerine yazılmaz.
* Sistem zamanını bir saniyedan daha uzun bir süre için değiştiremezsiniz.

Bu iki özellik yeni sistem yöneticileri için önemsiz olarak düşünülür ama çok önemli bir güvenlik önlemi sunarlar. UNIX sistemler dosyaları düzenlemek için birçok araç sunarlar ama bunların hangisini kullanırsanız kullanın hiç biri dosya sistemi üzerindeki bir ve sıfırları değiştiremez. Dosya sistemi buna engel olur. Aksi olsaydı, dosyaların erişim izinlerini aşar ve dosya içeriklerini değiştirebilir, içeriğini istediğimiz gibi düzenlerdik. Aslında BSD sistemlerde bu durumun aşıldığı tek durum sisteme yeni eklenen bir disk olması durumudur. Diskin formatlanıp üzerinde dosya sistemi oluşturulması ve etiketlenmesi işleminde bu durum söz konusudur. Bunu da ancak root yapabilir. Güvenlik düzeyi 2'de ise root bunu da yapamaz.

Saldırganların sık kullandığı bir teknik ise sistem zamanını değiştirip bir dosyayı düzenleyip ardından zamanı eski durumuna almak olabilir. Sistem yöneticisi sorunu belirlemek için dosyaların düzenlenme, oluşturulma zamanlarına bakacağı için söz konusu değişikliği göremez. Değiştirilen dosya ile aylar belkide yıllardır değiştirilmemiş gibi görünür ve saldırgan izini gizlemiş olacaktır.

Güvenlik Düzeyi: 3

Bu düzey ağ güvenli -network secure- olarak adlandırılır. Bir ve ikinci düzeylerdeki önlemlere ek olarak paket filtreleme kurallarını değiştirmeniz söz konusu olmaz. Özellikle de bant genişliğini yönetmek için kurduğunuz bir BSD sitemde pf kurallarını değiştirmeyi düşünmüyorsanız güvenlik seviyesini üç olarak belirleyip kullanabilirsiniz.

Hangi güvenlik Düzeyini Kullanalım?

Tercih edeceğiniz güvenlik düzeyi bağlandığınız ağ ile tercihlerinizle ilgilidir. Eğer geliştirme ortamı olarak kullanacağınız bir BSD sistem hazırlıyorsanız bu durumda "-1" kullanabilirsiniz. Sisteme ince yaparak çalışmaya devam edebilirsiniz. Öte yandan birçok geliştirme ortamı olarak kullanılan BSD sistemlerde güvenlik düzeyi "2" olarak rahatlıkla kullanılabilmektedir. Öte yandan PF kullanarak ağ yönetimi için bir BSD sistem hazırlayacak iseniz tüm ağ trafiğine ilişkin yapılandırmanızı gözden geçirip emin olduktan sonra güvenlik düzeyi 3'e geçebilirsiniz. Değiştirmek için sistemi yeniden başlatmanız gerekecektir.

Güvenlik DÜzeyleri ile Dosya Özniteliklerinin Yapamayacakları!

Bu yazı dizisinin ilk başında Apache.org sitesinin nasıl saldırıya uğradığını yazmıştık. Sisteminizde kullandığınız uygulamalardaki bir zayıflık saldırganlara istediğini verebilir. Öte yandan güvenlik düzeyini belirlediğiniz düzeyde kullanırken bir saldırgan sisteme kendi hazırladığı çekirdeği kopyalamayabilir ama diğer yandan login(1) programınızı kendi hazırladığı ve her kullanıcının giriş yaptığı andaki şifresini kendisine yollayacak olan sürümü ile değiştirebilir. Bununla da kalmayıp diğer başka dosyaları da kendi hazırladıkları ile değiştirebilir.

Dolayısıyla güvenlik düzeyini seçerken aynı zamanda kilit öneme sahip olan dosyalarınızın özniteliklerini chflags schg -R /bin/*, chflags schg -R /usr/lib vs değiştirmeyi unutmayın. Ama bir dosyanızı -örneğin rc.d dizinindeki bir dosyayı- gözden kaçırabilirsiniz. Bu gözden kaçırdığınız dosyanın bir saldırgan tarafından değiştirilip değiştirilmediğini kontrol etmediyseniz bu durumda yapılacak bir şey yok demektir. Benzer olarak /usr/local/etc/rc.d dizinindeki her .sh

uzantılı dosya sistem tarafından açılış sırasında çalıştırılmaya çalışılacaktır. /etc/rc sistemin açılışı tamamlandıktan sonra sistemin güvenlik düzeyini değiştirmektedir. Saldırgan bu durumu değiştiren bir uygulama yazmış ve bunu sisteme kopyalamış ise /etc/rc sonlandırıp kendi programlarını çalıştırıp ardından aynı süreci yeniden başlatabilir. Bu durumda sistem ciddi zarar görebilir.

Bunun gibi daha bir çok olasılıktan söz edilebilir. Felaketin boyutları saldırganın yaratıcılığı ile ilişkilidir. Sistem güvenliği basit bir çözümü olmayan karmaşık ve zor bir iştir. işinin ehli bir saldırgan bir şekilde komut satırına ulaştıysa yaptıklarını ancak iş işten geçtikten sonra fark edebilirsiniz. Bu durumların önüne geçmek için sisteminizi güncel tutup gerekli önlemleri almak yeterli olacaktır. Güvenlik seviyelerine güvenmek ve mutlak çözüm olarak görmek en büyük hatadır.

Dosya özniteliklerini düzenlerken cömert davranıp sisteminin büyük bir kısmına schg atadıysanız kısa zamanda sisteminizi güncellemenin veya terfi etmenin olanaksız olduğunu fark edebilirsiniz. Bu durum bir saldrıganın sisteme girmesini önlerken sistem yöneticisinin işini zora sokar. Bu durumda büyük bir olasılıkla /etc/rc.conf dosyasına schg atamış olabilirsiniz. Güvenlik düzeyini düşürmek için sistemi tek kullanıcı kipinde başlatıp /etc/rc çalışmadan önce gereken düzenlemeleri yapmanız gerekir. chflags noschg komutunu kullanıp sistemi normale döndürüp sistemin bu aşamadan sonra çalışmasını sağlamalısınız. Güvenlik düzeyini kaldırmadan sadece gereken bakım çalışmalarını yapıp ardından sistemin güvenlik düzeyini yükseltebilirsiniz. Bunu yapmak için de aşağıdaki komutta uygun güvenlik düzeyini seçip çalıştırmanız yeterli olur.

```
# sysctl kern.securelevel=<uygun_güvenlik_düzeyi>
```

Öte yandan uzaktan erişmek durumunda iseniz işler daha farklı olacaktır.

Gelecek ayki bölümde güvenlik konusunda diğer yapılandırma araçları ve önlemlerini ele alacağız.

BSD - XV

Programlar ve daemonlar için kullanıcı hesabı yapılandırması

Gökşin Akdeniz
goksin@enixma.org

BSD sistemlerde bulunan araçlar, dosyaların özelliklerini değiştirmekten, kullanıcıların yetkilerini ve erişim izinlerini değiştirmeye kadar birçok işlevi yerine getirir. Bu araçları doğru kullanıp yapılandırmayı gerektiği gibi düzenlemek yetkisiz kişilerin sisteme izinsiz erişimlerinin önüne geçecektir. Bu yapılandırma işlemleri içerisinde kullanıcıların yetkilerinin düzenlenmesi temel işlemlerden birisidir. Birçok işlem için root yetkilerine gerek kalmadan normal kullanıcı hesapları ile ilişkili grup ve dosya erişim yetkileri kullanılarak bir sistemde gereken tüm işlemler yapılabilir. Bunun yanında sistemdeki kullanıcı hesaplarına göz attığınızda sistemde çalışan programların ve daemon'ların kendilerine ait bir kullanıcı hesabına veya grubuna sahip olduğunu görürsünüz. Bu hesapların da sistem yöneticisi tarafından kontrol edilip gerektiği gibi yapılandırılması zorunludur. Bir daemon veya programa atanacak bir kabuk veya ev dizini bir saldırganın sisteme girmek için kullanabileceği bir açık kapı anlamına gelecektir.

Bir daemon veya programa ait olan kullanıcı hesabı o uygulama veya daemon'un çalışması için gerekli olan yetkileri tanımlar. Bu tür hesapların bir ev dizini, şifresi, kabuk ortamı bulunmaz. Bu hesaplara sahip olan kullanıcılar sadece belirli bir işlevi yerine getirirler veya diğer programlar bu hesapları kullanarak çalışırlar.

Aslında kullanıcı hesabı dediğimizde bir kullanıcıya ait olan ev dizini ile kabuk ortamı akla gelir. Normal bir kullanıcı hesabı bu özelliklere sahip ise, dosya oluşturabilir, düzenleyebilir, e-posta alıp gönderebilir vb. birçok işlemi yerine getirebilir. Kullanıcılar için bunlar gereklidir ama bir program veya daemon için gerekli değildir. Böylelikle bir saldırganın bu hesapları kullanarak programa veya sisteme verebileceği zararı sınırlayabiliriz.

BSD sistemlerde /etc/passwd dosyasına bakacak olursanız kullanıcı hesabınız dışında birçok kullanıcı hesabı olduğunu görebilirsiniz.

```
# $FreeBSD: src/etc/master.passwd,v 1.40.20.1 2009/04/15 03:14:26 kensmith Exp $
#
root:*:0:0:Charlie &:/root:/bin/csh
toor:*:0:0:Bourne-again Superuser:/root:
daemon:*:1:1:Owner of many system processes:/root:/usr/sbin/nologin
operator:*:2:5:System &:/:/usr/sbin/nologin
bin:*:3:7:Binaries Commands and Source:/:/usr/sbin/nologin
tty:*:4:65533:Tty Sandbox:/:/usr/sbin/nologin
kmem:*:5:65533:KMem Sandbox:/:/usr/sbin/nologin
games:*:7:13:Games pseudo-user:/usr/games:/usr/sbin/nologin
```

```
news:*:8:8:News Subsystem:/:/usr/sbin/nologin
man:*:9:9:Mister Man Pages:/usr/share/man:/usr/sbin/nologin
sshd:*:22:22:Secure Shell Daemon:/var/empty:/usr/sbin/nologin
smmsp:*:25:25:Sendmail Submission
User:/var/spool/clientmqueue:/usr/sbin/nologin
mailnull:*:26:26:Sendmail Default
User:/var/spool/mqueue:/usr/sbin/nologin
bind:*:53:53:Bind Sandbox:/:/usr/sbin/nologin
proxy:*:62:62:Packet Filter pseudo-
user:/nonexistent:/usr/sbin/nologin
_pflogd:*:64:64:pflogd privsep
user:/var/empty:/usr/sbin/nologin
_dhcp:*:65:65:dhcp programs:/var/empty:/usr/sbin/nologin
uucp:*:66:66:UUCP pseudo-
user:/var/spool/uucppublic:/usr/local/libexec/uucp/uucico
pop:*:68:6:Post Office Owner:/nonexistent:/usr/sbin/nologin
www:*:80:80:World Wide Web
Owner:/nonexistent:/usr/sbin/nologin
nobody:*:65534:65534:Unprivileged
user:/nonexistent:/usr/sbin/nologin
messagebus:*:556:556:D-BUS Daemon
User:/nonexistent:/sbin/nologin
polkit:*:562:562:PolicyKit Daemon
User:/nonexistent:/sbin/nologin
haldaemon:*:560:560:HAL Daemon User:/nonexistent:/sbin/nologin
goksin:*:1001:1001:Goksin
Akdeniz:/home/goksin:/usr/local/bin/bash
cyrus:*:60:60:the cyrus mail
server:/nonexistent:/usr/sbin/nologin
avahi:*:558:558:Avahi Daemon User:/nonexistent:/sbin/nologin
clamav:*:106:106:Clam Antivirus:/nonexistent:/sbin/nologin
cups:*:193:193:CUPS Owner:/nonexistent:/usr/sbin/nologin
```

Bu dosyada göreceğiniz çeşitli programlara ait olan kullanıcı hesaplarında kullanıcı ev dizinleri /nonexistent, /var/empty vs. ve kullanıcıya ait olan kabuk ortamının /sbin/nologin olduğunu görürsünüz. Bir ev dizine sahip olmayan kullanıcı dosya üzerinde bir işlem yapamaz. Ancak bir daemon veya program için bu gerekli değildir. Bu kullanıcılar kendilerine ait olan dosyaların sahibi olsalar da genellikle bu dosyalara yazamazlar.

Benzer olarak bu kullanıcı hesapları ile sisteme giriş yapamazsınız ve yapılmamalıdır da! Örneğin bind hesabı DNS sunucusu için ayrılmıştır. Bu kullanıcı hesabına giriş yapılmasını önleyen /sbin/nologin kabuk olarak atanmış olması gereklidir. Böylelikle bind kullanıcı hesabı ile sistem giriş yapılamaz.

Peki /sbin/nologin atanmaz ise ne olur? Bunu bir örnek ile açıklayalım. Apache web sunucusu www kullanıcı hesabını kullanır. Apache'de bir güvenlik zafiyeti bulunduğunu ve saldırganın bu zafiyeti kullanarak herhangi bir komutu çalıştırabildiğini düşünelim. Bu durumda saldırgan programın yetkileri çerçevesinde istediği kodu, komutu vb çalıştırabilir. Saldırganın komut satırına erişmek isteyeceğini düşünürsek, ilk çalıştırmak isteyeceği /bin/sh olacaktır. /sbin/nologin ile kabuk ortamı atanmadığı için kullanıcı hesabını kullanmaz bu durumda sistem genelinde geçici dosyaların yer aldığı /tmp ve /var/tmp dizinlerini kullanmaya çalışacaktır. Ayrıca Apache yapılandırma dosyanızın sahibi root veya web sunucusunun yönetiminden sorumlu olan kullanıcı grubu ise Apache yapılandırma dosyaları üzerinde bir değişiklik yapılamaz. Zira Apache www kullanıcısı olarak çalışmaktadır. www kullanıcısının da sistem üzerinde herhangi bir dosya üzerinde yetkisi olmadığı için Apache dışında diğer programlara veya BSD sisteminize saldırması gereklidir.

İlk bakışta bu yapılandırma ile oluşturulan kullanıcı hesaplarının güvenlik konusundaki sorunları tamamen ortadan kaldırdığı düşünülebilir ama gerçekte yapılan saldırganın işini zorlaştırmaktan başka bir iş değildir. Saldırgan apache yapılandırma dosyasını okuyabilir. Web uygulamaları güvenli kodlanmamış ise, dosyalar içerisinde veritabanı vb. ait şifreler bulunuyorsa bunlar da saldırgana istediğini verecektir. Öte yandan sisteminizi güncel tutup gerekli yamaları kuruyorsanız sisteminizi güvenli kılıyorsunuz demektir.

Nobody Kullanıcı Hesabı

Kullanıcı hesapları içerisinde göreceğiniz nobody kullanıcısı kabuk ve ev dizini olmayan bir kullanıcı hesabıdır. Bu nedenle de çeşitli sistem servislerini kullanmak için hazır yapılandırılmış bir hesap olarak düşünülebilir. Bu durumda sistemde çalıştırılan birçok daemon ve programı nobody hesabı ile çalıştırmak yeterince güvenli olduğu düşünülebilir. Bu yaklaşımın olumsuz tarafı bir saldırganın bu hesap ile çalıştırılan bir daemon veya programı ele geçirmesi durumunda aynı hesap ile çalıştırılan tüm programları da kontrol edebilecek olmasıdır. Bu nedenle her bir program ve daemon'un kendisine ait bir kullanıcı hesabı atanarak, yapılandırılması ve bu hesaplar ile çalıştırılması ilk bakışta zor gibi gelebilir ama tek bir kullanıcı hesabında toplanan daemon ve programların gümüş tepside saldırgana sunulmasından çok daha iyidir.

Daemon ve programlar için bir kullanıcı hesabı oluşturulması için adduser(8) kullanabilirsiniz. adduser(8) yapılandırmasını gereksinimlerinize göre düzenleyerek bu süreci hızlandırabilirsiniz. Aşağıdaki adduser yapılandırma örneğinde daemon ve programlar için kullanılacak bir kullanıcı hesabına ait bazı değerler görülmektedir.

```
Username:          Program/daemon ya da işlevi tanımlayan isim.
                   (Web sunucusu için www gibi)
home directory:    kullanıcı hesabının ev dizini /nonexistent
                   atayın
shell:             Hesaba atanacak olan kabuk /usr/sbin/nologin
                   atayın
UID/GID:           Bu tür hesaplar için ayrılmış olan kullanıcı
                   ve grup numarası
full name:         Tam isim olarak ne iş yaptığını açıklayan
                   bir isim. Örneğin sshd için Secure Shell
                   Daemon gibi.
password:          İlk aşamada bir şifre atayın. Hesap
                   oluşturulduktan sonra chpass(1) ile şifreyi
                   "*" olarak atayın. Böylelikle hesabın
                   şifresi bloke edilir. Bu hesabın kullanıcı
                   adı ile sisteme giriş yapılmak istendiğinde
                   şifre olarak "*" girilse bile giriş
                   yapılamaz!
```

Ag zerinden Erişimin Kontrol Edilmesi

Ağ trafiğinin kontrol edilmesi denildiğinde bir ağ geçidi ile internetten veya diğer ağlardan ağ geçidinin arkasında kalan bilgisayarlara erişim kontrolü akla gelir. Ancak söz konusu olan, BSD sistemlerde güvenlik olduğu için bir ağ geçidi kurulumu ve yapılandırması yerine bir BSD sistemde hangi servislere kimin nasıl erişeceği, erişimin kontrol edilmesi söz konusudur. BSD sisteminizi ister kişisel bilgisayarınız ister sunucu olarak kullanın, ağ bağlantınızı ve çalıştırdığınız servisleri kontrol etmeniz zorunludur. Bilgisayar ile diğer bilgisayarlar arasındaki ağ trafiğini kontrol etmek için TCP wrappers ile BSD sistemlerde kullanılan firewall yapılandırma araçları bu iş için yeterlidir. TCP wrappers ağ servislerini çalıştıran daemon'lara erişimi kontrol etmek için kullanılır. Yazılan daemon veya program TCP wrappers'ı desteklemelidir aksi halde kullanılamaz.TCP wrapper kullanımı basittir ve ağ TCP/IP bilgisi gerektirmez.

Öte yandan firewall yapılandırması TCP/IP bilgisi gerektirebilir. Bu durum ise sistemdeki servislerin nasıl ve ne şekilde çalışacağının ayrıntılı bir şekilde belirlenmesi söz konusu ise gerekecektir. BSD sistemlerde /etc/rc.conf dosyasına ekleyeceğiniz birkaç satır ile gereken yapılandırmayı uygulamaya geçirebilirsiniz.

Yapılandırmanızı hazırlamadan önce bir karar vermeniz gerekir. Bu karar sisteminize ağ üzerinden gelecek isteklere ilişkin politikanızın ne olacağıdır. Politikanız "hep evet" veya "hep hayır"dan bir tanesi olmalıdır. Politikanız neden bu seçeneklerden birisi olmalı? Kısmen evet veya kısmen hayır ya da herhangi bir politika belirlemesem olmaz mı?

Belirleyeceğiniz politika güvenlik konusundaki politikanızın veya kabul

edilebilir olarak gördüğünüz risklerin ölçüsüne göre değişir. Herşeye evet yani gelen tüm isteklere yanıt verilecek politikası ağ üzerinden gelen tüm trafiği kabul ettiğiniz anlamına gelir. Bu durumda olası zayıflıkların ortaya çıkmasının ardından sistem yöneticisi olarak yapmanız gereken bütün bu yamaları zamanında ve hızlı bir şekilde uygulamak olacaktır. Zira politikanız her türlü trafiğe izin veriyorsa birilerinin bu zayıflıkları öğrendiğinde denemeye kalması olasılığı oldukça yüksektir. Bu politika sizin süpermen hızına sahip olmanızı gerektirir. Bu hıza sahipseniz bu politikayı benimseyin, tam size göre... Bir saldırganın sisteme erişip poposunun 10 Megapixel çözünürlükteki taranmış görüntüsünü internet sitenize koyuyorsa, sistemdeki kullanıcı adları ve şifrelerini yayınlıyorsa, e-posta sunucunuzu kullanıp herkese yaptıklarını takdir etmeleri gerektiğini belirten birer e-posta mesajı yolluyorsa bu durumu nasıl açıklayacağınıza da önceden karar vermeniz gerekir. Ama önemli değil bunlar. Sizde süpermen hızı var. haberler duyulmadan önce siz çok uzaklara gitmiş olabilir ve zabıtalardan hızlı koşan seyyar satıcı kariyerinize başlamış olursunuz.

Her şeye hayır politikasını benimsiyorsanız bu durumda sizin belirleyeceğiniz koşullara göre gelen isteklere yanıt verecek bir sistem tasarlayacaksınız demektir. Bu tasarımın evet politikasına göre üstün yanı sisteminizin sadece ve sadece gerektiğinde konuşacak olmasıdır. Saldırganların yapacakları her türlü girişimi başından büyük bir ölçüde savuşturmuşsunuz demektir. Bu politika size gerektiğinde evet demenize olanak tanır. Kabul edeceklerinizi dışındaki kalan her şey geçersizdir. Bu durumda size gelen erişim açılması vb. istekleri değerlendirip uygun görüyorsanız izin verir, gerekli açıklamayı yaparak bunun güvenlik politikanız gereği olduğunu açıklarsınız. Ertesi gün ve sonraki günler kariyerinize aynı şekilde devam edersiniz.

O zaman ağ trafiğini kontrol etmek için firewall olarak hangi aracı seçeceğiniz sorusu akla gelecektir. BSD sistemlerde birçok firewall uygulaması bulunur. IPFW, IP Filter ve PF en yaygın olarak kullanılanlarıdır. Bu uygulamalar için geliştirilmiş grafik arayüzler de bulunur. IPFW FreeBSD sistemlerde kullanılır. /etc/rc.firewall (ipv4 için) ve /etc/rc.firewall6 (ipv6 için) dosyalarındaki yapılandırma kullanılarak çalıştırılır. Diğer uygulama olan IP Filter BSD sistemler için geliştirilmemiştir. Darren Reed tarafından geliştirilmiş ve birçok UNIX sisteme aktarılmıştır. PF veya pf -packet filter-, ilk olarak OpenBSD için geliştirilmiş ve kısa sürede tüm diğer BSD sistemlerde de kullanılmaya başlanmıştır. Diğer firewall uygulamaları ile karşılaştırıldığında pf diğerleri kadar güçlüdür ve bazı konularda diğerlerinden çok daha başarılıdır.

PF, ağ arayüzüne gelen tüm TCP/IP paketlerini tanımlanmış olan kurallar ile karşılaştırır. Bir kural paket ile ilgili olarak şu eylemleri tanımlar: İzin verebilir, red edebilir, değiştirebilir veya çöpe atabilir. Eşleşen kuralın gereği çekirdek tarafından yerine getirilir. PF işleyişi ve kuralların tanımlanması son derece basittir. Bu basitlik ise temel düzeyde TCP/IP bilgisine sahip birisi için geçerlidir. Aksi halde PF için yazdığınız kuralın neden istediğiniz gibi işlemediğini anlamak için uğraşmanız gerekecektir.

PF Yapılandırması

BSD sistemlerde PF işlemesi için pf.ko çekirdek modülünün yüklenmesi gereklidir. Bu modül standart olarak yüklenmediği için açılışta yüklenmelidir. /etc/rc.conf dosyasına

```
pf_enable="YES"
```

Satırını ekleyerek PF çekirdek modülünü yükleyebilirsiniz. PF çekirdek modülü yüklenmesi ile işlem tamamlanmış olmamaktadır. PF'in çalışması için kuralların tanımlanması gereklidir. Eğer bir kural tanımlamaz iseniz PF standart olarak gelen her bağlantıyı kabul edecektir. Diğer bir deyişle hep evet politikası geçerlidir.

Ağ üzerinden gelen isteklerin hangisinin kabul edilip hangisinin

edilmeyeceği önemlidir. Her bağlantıyı kabul edecekseniz bu durumda saldırıları engelleyecek olan kuralları oluşturmanız gerekir. Öte yandan hepsini red edip izin verdiğiniz bağlantıların gerçekleşmesini kabul edeceğiniz bir politika, ağınızı ve/veya sisteminizi birçok saldırıdan koruyacaktır. Bu politikanın zor olan yanı politikanızı uygularken uzak sistemlere ssh ile bağlanıyorsanız yapacağınız bir hata sonucunda uzak sistem ile olan bağlantınızı kaybedebilecek olmanızdır. Diğer bir deyişle uzak sistem erişilmez olur. Bu hatayı hemen hemen tüm sistem yöneticileri en az bir defa yapmıştır. Başınıza gelirse utanılacak bir şey yok.

Gelen bağlantıların tümden red edilmesi politikası ilk bakışta hata yapmaya eğilimli bir politika gibi görünebilir ama her saldırı için kural yazmaktan daha kolaydır. Ancak PF işleyişini yeni öğreniyorsanız ve sisteme doğrudan erişiminiz varsa hatalarınızı görüp düzelterek daha çok şey öğrenirsiniz. Uzak sistemler için kural tanımlıyorsanız ve uzak sisteminiz kuş uçmaz kervan geçmez bir yerde bulunan veri merkezinde ise bu durumda oradaki sistem yöneticilerinin yeterli bilgi ve beceriye sahip olduğundan emin olun. M$ sertifikalı sistem yöneticilerine durumu anlattığınızda size uzaylı gibi bakacağından emin olabilirsiniz

Paket Filtreleme ve SPI

Bir TCP bağlantısı gerçekleştirildiğinde istemci bağlantı kuracağı sunucuya bir SYN paketi yolar. SYN paketi istemci ile sunucu arasında kurulacak olan bağlantı isteğini sunucuya iletir. (Connection synchronization) Sunucu söz konusu isteğin yönlendirildiği portu dinliyorsa bu gelen isteğe bir SYN-ACK paketi ile yanıt verir. Bu paket istemciye SYN paketinin alındığını anlatır ve bağlantı kurulabilmesi için gerekli olan bilgiler istemciye ACK paketi ile iletilir. İstemci sunucudan gelen SYN-ACK paketine ACK paketi yollayarak (acknowledged) karşılık verir. Bu işleme üçlü el sıkışma (three-way handshaking) adı verilir. Bir istemci ile sunucu arasında iletişim kurulabilmesi için bu üçlü el sıkışma işleminin gerçekleşmesi zorunludur. Eğer sunucunuzdaki paket filtreleme kuralları SYN-ACK paketlerinin iletilmesine izin vermiyorsa sunucunuz çalışıyor olsa bile istemciler bağlantı kuramaz.

Doksanların ilk yarısında kullanılan paket filtrelerinde istemcilerden sunuya ulaşan paketlerin her biri teker teker kontrol ediliyordu. Bir paket, paket filtresinde tanımlanan bir kurala uyuyorsa filtre tarafından iletilmesine izin veriliyordu. Filtreler gelen, geçen paketlerin hangi bağlantıya ait olduğunu kontrol etmemekteydi. Örneğin gelen paket bir SYN-ACK paketi ise ve paket içerisinde hedef adresi de yer alıyorsa bu durumda paket filtresi bu paketin önceden yollanmış olan bir SYN paketin karşılık geldiğini kabul edip paketin geçmesine izin veriyordu. Paket filtresi önceden yollanan SYN paketinin hangi bilgisayardan geldiğini bilmediği için bu saldırganların kendi hazırladıkları SYN-ACK paketlerini sunuculara yollayarak sunucudan gelen yanıtları, inceleyerek uzak sistemlere erişmesine olanak veren bir zayıflık olarak ortaya çıkıyordu.

Sonraki yıllarda geliştirilen paket filtreleri ile sunucular ve istemciler arasında gelen ve giden paketleri kontrol altına alıp gelen ve giden paketlerin hangi bağlantıya ait olduğu kontrol edilmeye başlandı. Eğer gelen SYN-ACK paketi var olan bir bağlantıya ait ise geçmesine izin veriliyor eğer tersi söz konusu ise paket çöpe gönderiliyordu. Bu durum çekirdekteki kodun karmaşıklaşmasına neden olsa da dinamik filtreleme kuralları eski tip kurallara göre hem daha kolay yazılmakta hem de daha etkin bir kontrol sağlamaktadır.

Birçok kullanıcı paket filtresinin işleyişine ilişkin bu bilgileri öğrendiğinde paket filtresinin bir “firewall – güvenlik duvarı” olduğunu düşünür. Aslında firewall – güvenlik duvarı kavramı sadece paket filtresi demek değildir. Güvenlik duvarları hem yazılım hem de donanım olarak bir ağın içeriden ve dışarıdan gelebilecek olan tehditlere karşı korunmasını üstlenmiş olsa da aslında ifade edildiğinden

çok daha karmaşık ve gelişmiş sistemlerdir. Güvenlik duvarı kavramı bugün için bu kavramdan uzak olup sadece pazarlama faaliyetlerinde kullanılan içi boşaltılmış bir kavrama dönmüştür.

Aslında BSD sistemler mükemmel bir firewall olarak kullanılabilir. Ağ protokollerini, PF, güvenlik uygulamalarını ve sistemin işleyişini iyi bilen bir sistem yöneticisi elinde onbinlerce dolarlık Checkpoint veya Cisco PIX sistemlerin yerine koyabileceğiniz sıradan bir bilgisayar da aynı işi fazlasıyla görecektir. Bu tür bir sistemi kurmak için /usr/ports/net ve /usr/ports/security altındaki uygulamaları kullanabilirsiniz.

PF yapılandırması

PF, yapılandırma dosyası /etc/pf.conf dosyasıdır. Bu dosyanın içeriğinin doğru sıra ve biçimde hazırlanması zorunludur. Aksi durumda hatalı tanımlamalar yaparak sistemin ulaşılmaz veya tamamen sağır, dilsiz ve kör olmasına neden olabilirsiniz.

pf.conf dosyasına kuralları yazarken doğru sıralama aşağıdaki gibidir. Bu sıralamayı anımsamak yerine dosya içerisine yorum satırı olarak ilgili bölümler halinde eklerseniz ileride yapacağını düzenlemeler için size bir referans sağlayacaktır.

Makrolar
Tablolar
Seçenekler
Paket kontrolü
Bant genişliği yönetimi
Ağ adres dönüşümleri
Yönlendirme
Paket filtreleme kuralları

Görüldüğü gibi PF bir firewall'dan daha fazlasını yapmaktadır. PF aslında genel amaçlı bir TCP/IP yönetim aracıdır. Burada sadece basit bir filtre oluşturacağımız için bütün bu konuları ele almıyoruz.

Makrolar

Makrolar, PF için kuralların yazılmasını ve okunmasını kolaylaştıran değişken tanımlamalarıdır. Böylelikle ağ yapılandırması değiştiğinde gerekli değerleri makroları düzenleyerek tüm kuralları kısa zamanda yeni duruma uyarlayabilirsiniz. Aşağıdaki makrolar ağ kartınızı ve bu ağ kartına atanmış olan ip adresini tanımlamaktadır. Senaryo gereği sistemde bir sunucu kurulu olduğunu ve buna uzaktan ssh ile eriştiğimizi kabul ediyoruz.

```
ag_karti="bge0"
sunucu_adres="192.168.7.2"
```

Sonradan yazılacak olan kurallarda ağ kartınızı $ag_karti ve sunucu adresini de $sunucu_adres değişkenleri tanımlayabilirsiniz. Bu değişkenlerin değerlerini her seferinde yeniden yazmanıza gerek yoktur. Sonradan olacak değişikliklerde sadece ilgili kısımları değiştirmek yeterlidir.

Tablolar ve Seçenekler

PF birçok adresi bir tabloda saklayabilir. Bu kullanım şekli burada kullanacağımız sistem için gerekli değildir. Ancak bilgi olarak bilmekte yarar var. Daha sonra PF'i ele alan bir yazıda bu konuya yeniden döneceğiz.

Seçenekler kısmında ağ bağlantılarına ilişkin yapılandırma seçenekleri bulunur. Bu değerleri değiştirmenize gerek yoktur. Standart ayarlar yeterli olur.

Paket kontrolü

TCP/IP protokolünde paketler bölünerek bilgisayarlar arasında iletilir. Bu parçaların her birisini sunucunun teker teker işlemesi ve bu sırada gelen isteklere yanıt vermesi sistem üzerinde büyük bir yük oluşturur. Bu nedenle sistemdeki ilgili programa paketler birleştirilip tek bir paket olarak iletilir. PF için bu birleştirme işlemi scrubbing olarak tanımlanır. Bu işlem ile gelen paketler birleştirilir ve geçerli bağlantıya ilişkin olan paketler ilgili programa aktarılır ve diğerleri çöpe gönderilir. Bunu sağlayan kural aşağıdaki gibi yazılır.

```
scrub in all on $ag_karti
```

Bu kural bilgisayara ulaşan tüm paketler için geçerlidir. İlk bakışta birleştirme işlemi faydalı bir işlemmiş gibi görünebilir. Ama faydalı olmaktan öte zorunlu bir işlemdir. PF kuralları bütün paket üzerinde uygulanır. Parçalara ayrılmış paketler üzerinde kuralların uygulamaya konulması sadece bağlantı sorunları yaratır. Yukarıdaki kuralı yazmasanız sorunu bulmak için zaman kaybedersiniz.

Bant genişliği, Ağ Adres Dönüşümleri ve Yönlendirme

PF ile bant genişliği kontrolü uygulayabilirsiniz. Özellikle de ağ geçitleri üzerinde işletim sistemini esas alıp kısıtlama getirebilirsiniz. Örneğin Windows çalışan bilgisayarlar sadece 256 K kullansın gibi veya bir ip adresinin belirli bir bant genişliğinden fazlasını kullanmaması gibi.

Ağ Adres dönüşümü – NAT – Network Address Translation – ve bir porta gelen trafiğin başka bir porta yönlendirilmesi işlemleri bu kısımda tanımlanır. Bu bölüm ancak firewall kuruyorsanız ilgili kuralları barındıracaktır.

Paket Filtreleme Kuralları

Bu bölümde gelen ve giden bağlantılara ilişkin kurallar yer alır. Aşağıdaki iki kural sunucuya gelen ve sunucudan istemciye giden paketlere ilişkin genel kurallardır.

```
pass in on $ag_karti proto tcp from any to ($ag_karti) port 80
flags S/SA keep state
pass out on $ag_karti proto { tcp, udp } all keep state
```

Bu iki kural genel olarak bir sunucudan gelen ve giden paketlere ilişkin yazılacak PF kurallarının genel yapısını göstermektedir. Bir kuralda gelen paketlerin geçmesine izin verilecek ise pass in çıkan paketlerin geçmesine izin verilecek ise pass out terimleri kullanılır. Bunları izleyen terimlerde on $ag_karti ise ilgili ağ arayüzünü/ağ kartını tanımlar. Ardından gelen terim olan proto adından da anlaşılacağı üzere protokolü tanımlar. Bu protokoller BSD sistemde desteklenen herhangi bir protokol olabilir. Gelen paketlerin kaynağı from <ip.ad.re.si> ile tanımlanır. From ardından eğer bir ip adresi yazılırsa bu durumda yazılan ip adresinden gelen paketlere uygulanan bir kural olur. Örneğin from <ağ_geçidi_adresiniz> olarak yapacağınız bir tanımlama sunucunun sadece ağ geçidinden gelen isteklere yanıt vermesine ve diğerlerini yanıtlamamasına neden olur. Yukarıdaki kural da her isteğe yanıt vermesi için from any olarak tanımlanmıştır. Ardınan gelen to {$ag_karti} terimi ağ kartına gelen paketleri tanımlar. Bu alandaki from any to {$ag_karti} terimi herhangi bir ip adresinden ağ kartına gelen tüm paketlere uygulanacağını tanımlamaktadır. Port 80 tanımlaması ise gelen isteklerin 80 numaralı porta yönlendirilmiş olması gerektiğini tanımlar. Bu da standart olarak bir web sunucusunun gelen istekleri dinlemekte olduğu portun numarasıdır. Eğer sunucunuz 80 değil de farklı bir portu dinlemekte ise, ilgili port numarası burada belirtilmelidir. Flags S/SA keep state terimleri kuralların son öğeleridir. Flags S/SA tcp/ip paketlerindeki bayrakları belirtir. “/” işaretinden önceki harf bayrakların gelen paketlerde tanımlı olması gerektiğini belirtir. “S/”tanımlamasının

ardından gelen SA ise SYN veya ACK paketlerinden en az birisinin belirtilmiş olmasını tanımlar. Keep state terimi ise PF'in söz konusu bağlantıyı izlemesini yani dinamik filtreleme yapmasını belirtir. Bu kuralı bütün olarak yorumladığınızda şu ortaya çıkmaktadır:

“Ağ kartına ulaşan paketler içerisinde 80 numaralı porta yönlendirilmiş olan paketler varsa, ve eğer SYN veya ACK bayrağını taşıyorlarsa geçmelerine izin ver, bağlantıyı izleyip dinamik filtreleme uygula!”

Yukarıda verilen bilgiye dayanarak ikinci kuralın sunucudan diğer sistemlere ağ kartı üzerinden giden tüm UDP ve TCP paketlerine izin vermekte ve bağlantıları izleyip dinamik filtreleme uygula anlamına geldiği görülebilir.

Kurallarda gördüğünüz makro tanımlaması {$ag_karti} ağ kartına atanmış statik ip bulunmuyorsa kullanılmalıdır. “{...}” arasında makro tanımlaması ip adresinin DHCP sunucu tarafından alındığını belirtmektedir. Bu durumda ip adresi sunucu tarafından atanacağı için ayrıca bir ip adresi tanımlamaya gerek olmayacaktır. Pf gerekeni yapacaktır.

Bazı pf örneklerinde “flags...” ile tanımlanan bayrakların kullanılmadığını görebilirsiniz. FreeBSD 6.0 ve öncesi sürümlerde “flags...” tanımlası gerekli iken, FreeBSD 7.0 ve sonrasında ise standart olarak yer almaktadır. OpenBSD veya NetBSD kullanıyorsanız da aynı durum geçerlidir. Pratikte uygularken flags kullanmanız daha kesin bir kontrol sağlayacaktır.

Örnek Bir PF Yapılandırması

Ufak bir ağ üzerinde yer alan bir web sunucusu için hazırlanmış olan pf kuralları aşağıda yer alıyor. Bu sunucuya ssh ile erişilebiliyor. Üzerinde web ve e-posta sunucusu çalışıyor. Kendi gereksinmelerinize göre uyarlayarak kullanabilirsiniz.

```
ag_karti=”bge0”

scrub in all

set block-policy drop

block in all on $ag_karti

#yerel ag uzerinden sysadmin icin ssh erisimine izin ver

pass in on $ag_karti proto tcp from 192.168.1.12 to $ag_karti
port 22

#POP3 erisim izni

pass in on $interface proto tcp from 192.168.1.0/24 to
$interface port 110

#SMTP (25), HTTP (80),HTTPS (443) izin ver

pass in on $ag_karti proto tcp from any to $ag_karti port 25
pass in on $ag_karti proto tcp from any to $ag_karti port 80
pass in on $ag_karti proto tcp from any to $ag_karti port 443

# DNS sunucusuna erisim izni

pass in on $ag_karti proto tcp from any to $ag_karti port 53
pass in on $ag_karti proto udp from any to $ag_karti port 53

# sunucudan disari erisim izni

pass out on $ag_karti proto { tcp, udp } all
```

Yukarıdaki örnek filtre kuralları önceden belirtilen kural yazımına uygun olarak tanımlanmıştır. İlk olarak ag_karti adlı bir makro tanımlanmıştır. Eğer ağ kartını değiştirmek durumunda kalırsak sadece ilgili alanı değiştirip işimizi tamamlıyoruz. Ardından gelen tüm paketleri bir hizaya sokup sonra işleme alıyoruz. Bundan sonra ise yukarıda geçmeyen “block in all” ve “set block-policy drop” terimleridir. Bunun

anlamı izin verilmeyen paketlerin çöpe gönderileceğidir. SSH erişimi sadece 192.168.1.12 ip adresinden yapılabilmektedir. Diğerlerinden gelen istekler dikkate alınmayacaktır. POP3 izni sadece yerel ağ için tanımlanmıştır. Yerel ağ aralığı ise 192.168.1.0/24 olarak tanımlıdır. SMTP, HTTP ve HTTPS istekleri ise dışarıya da açıktır. DNS sunucusu da benzer olarak hem yerel ağa hemde yerel ağ dışına açıktır. DNS sunucusuna erişim hem TCP hemde UDP ile olabildiğinden her iki protokol de desteklenmektedir. Gerekli görüyorsanız birisini kapatabilirsiniz.

Yukarıdaki paket filtre kuralları örnek kurallardır. Bir saldırganın sunucuyu taraması durumunda sadece çalışan servisler olarak DNS, WEB, POP VE SMTP ile ağ kartınızın mac adresi dışında bir bilgi edinmesi söz konusu olmayacaktır. Daha da ileri gidersek bir şekilde ssh erişimi elde etse bile standart port dışındaki bir porta yönlendirmesi durumunda dahi ssh erişimi çalışmayacaktır. Zira söz konusu port kapalıdır.

PF Kurallarını Çalıştırmak

PF kurallarını yazdığınızda bir hata içerip içermediğini kontrol etmeniz gereklidir. Hatalı yazılmış kurallar istediğiniz sonucu vermeyecektir. Ayrıca yapılan imla hataları da kuralı geçersiz kılacaktır. Bu durumda imla denetimi yapmak için bunu pfctl(8)'e bırakıyoruz. Pfctl'i root olarak çalıştırıp kurallarınızı kontrol edebilirsiniz. Eğer hatalı bir satır bulunuyorsa bu durumda hata döndürülecektir. Kontrol etmek için aşağıdaki komutu verin.

```
# pfctl -nf /etc/pf.conf
```

-n seçeneği dosyadaki hataları denetler. F seçeneği de yazdığınız kuralların tanımlı olduğu dosyayı belirtir. Kurallarınızın hatasız olduğundan emin iseniz kurallarınızı aşağıdaki komutu verip aktif kılabilirsiniz.

```
# pfctl -f /etc/pf.conf
```

PF yapılandırmanızı kolaylıkla ve çabuk bir şekilde pfctl ile değiştirebilirsiniz. Farklı durumlarda kullanılmak üzere birçok kural tanımlayabilir ve bunları pfctl ile istediğiniz anda uygulamaya koyabilirsiniz. Hatta erişim kısıtlamalarını günün belli saatlerinde uygulanacak şekilde pfctl ve cron ile uygulamaya koyabilirsiniz.

Herhangi bir anda çalışan pf kurallarınızı sorgulamak için /etc/pf.conf dosyasını okumanıza gerek yoktur. Bunun yerine

```
# pfctl -sr
```

Komutunu kullanabilirsiniz. O sırada geçerli olan kuralların bir listesini size döndürecektir.

Eğer o sırada geçerli olan kuralları sonlandırmak isterseniz yine pfctl kullanabilirsiniz.

```
# pfctl -Fa
```

Komutu geçerli olan kuralların hepsini sonlandırır ve sanki pf.conf dosyanızda kural bulunmuyormuş gibi işlemsini sağlar. Bu komutu yeni kuralları yükleyeceğiniz zaman kullamanıza gerek yoktur. Yapmanız gereken yeni kuralları yüklemektir. Yeni kurallar yüklendiğinde eski kurallar geçerliliğini kaybeder.

TCP Wrappers

TCP wrappers, bir sunucuda ağ üzerinden gelen istekleri dinleyen programların nasıl hareket edeceğini tanımlar. TCP Wrappers adı yanıltıcı olabilir. Düşünülenin aksine hem TCP hem de UDP desteğine sahiptir. TCP Wrappers eski bir UNIX stantardıdır ve BSD sistemlere ilk sürümlerinde eklenmiş ve bugüne dek kullanıla gelmiştir. BSD sistemlerin temel yapısında yer alan tüm programlar TCP wrappers

destekler. Bunun dışında kalan diğer uygulamalar destekleyebileceği gibi destekleyemeyedebilir. Bir programın TCP wrapper kullanabilmesi için libwrapper paylaşımlı kütüphanesini kullanması yeterlidir. Wrappers'ın yaptığı iş, bir programın ağ üzerinden gelen istekleri değerlendirirken wrapper yapılandırmasını dikkate alarak gelen isteklere nasıl yanıt vereceğini tanımlamasıdır. Yapılandırma isteği geri çevirmesini tanımlıyorsa uygulama olumsuz yanıt döndürecek veya isteği yanıtlamayacaktır.

TCP Wrappers İşlevi

Wrappers'ın yaptığı temel işlev aslında inetd(8)'yi korumaktır. inetd(8)'nin yaptığı iş diğer programlar için ağ üzerinden gelen istekleri dinleyip ilgili uygulamaya iletmektir. inetd(8) kullanıyorsanız dikkatli olunması gereken bir nokta söz konusudur. inetd(8) ile birçok uygulamayı ağ üzerinden gelen isteklere yanıt verecek şekilde yapılandırdıysanız sadece ve sadece sizin belirlediklerinizi dışında başka bir uygulamanın çalışmadığından emin olmanız gereklidir. inetd(8) tarafından çalıştırılmayan bir uygulama da tcp wrapper destekliyorsa burada anlatıldığı şekilde wrapper ile kullanılabilir.

TCP Wrapper Yapılandırması

Wrappers gelen istekleri /etc/hosts.allow dosyasında belirtilen kuralları dikkate alarak değerlendirir. Bu dosyada tanımlanan kuralların sırası önemlidir. Çünkü wrapper ilk eşleşen kuralı uygulayacak ve diğer kuralları dikkate almayacaktır.

/etc/hosts.allow dosyasındaki kuralların hepsi birer satır uzunluğundadır ve ":" ile ayrılmış olan üç alandan oluşur. Birinci alanda daemon adı tanımlanır. İkinci alanda istemcinin adı veya adresi, üçüncü alanda da ilgili işlem tanımlanmıştır. Örneğin ftp daemon için tanımlanan kural aşağıdaki gibi olabilir.

```
ftpd : all : deny
```

Bu kural ftpd daemondan gelen istekler nereden gelirse gelsin red edilecek demektedir. Eğer daha önce bu kuralın aksini tanımlayan bir kural bulunmuyorsa ftpd gelen tüm isteklere yanıt vermeyecektir. Eğer izin veren bir kural tanımlamak istersek yukarıdaki kuralda tanımlı olan deny kısmını allow olarak değiştirip yazmamız gereklidir.

```
ftpd : all : allow
```

Bir daemon eğer inetd(8) tarafından başlatılabiliyorsa yukarıdaki örnek kuralda görüldüğü gibi adı yazılarak inetd tarafından çağrılabilir. Apache web sunucusu wrappers destekler ama inetd(8) tarafından başlatılmaz. Eğer wrappers destekli tüm daemonların çalıştırılmasını isterseniz bunun için özel olarak tanımlanmış bir daemon adını; "ALL"u kullanabilirsiniz.

Benzer olarak sisteminizde birden çok IP adresi tanımlanmış ise bu ip adreslerinin her birisi için ayrı wrapper yapılandırmaları tanımlayabilirsiniz. Aşağıdaki örnekte ağ geçidi olarak 10.0.0.1 adresi tanımlıdır. Ftp daemon ağ geçidi üzerinden gelen isteklere yanıt vermeyecek ama yerel ağa bakan bacak olan 192.168.7.1 adresine sahip olan karta gelen isteklere yanıt verecektir.

```
ftpd@10.0.0.1 : ALL : deny
ftpd@192.168.1.1 : ALL : accept
```

İstemcilerin Tanımlanması

İstemciler denildiğinde belirli bir ip adresi, ağ adres bloğu (192.168.1.0/24 gibi) veya sistem isimleri (hostnames) vb. kullanılır. Bunların teker teker tanımlanmasına gerek yoktur. Birden fazlası aralarında boşluk bırakarak ardışık olarak yazılarak kullanılabilir. Aşağıdaki örnek kuralda bu durum gösterilmektedir.

```
ALL : tardis.elkotek droideka.elkotek 192.168.7.3 192.168.7.5
```

```
192.168.7.7 : allow
```

Benzer bir kural dışarıdan saldırı düzenleyen bir dizi ip aralığının benzer olarak yasaklanması için kullanılabilir. Aşağıdaki kuralda 195.87.18 ile başlayan ip aralığı bloke edilmiştir.

```
ALL : 195.87.18.0/255.255.255.0 : deny
```

Eğer çok sayıda ip adresi veya alan adını engellemek isterseniz bunları bir dosyaya yazıp dosyanın bulunduğu dizini /etc/hosts.allow dosyasında tanımlayarak da aynı işlemi yapabilirsiniz. Öte yandan wrappers bazı terimler kullanarak istemcileri kolayca tanımlamamıza olanak verir.

Terim	Kullanım
ALL	Tüm istemciler
LOCAL	Tüm sistem adlarını (hostname) belirtir. Sistem adında nokta bulunmamalıdır. Yerel ağınızdaki tüm sistemlere karşılık gelir.
UNKNOWN	Sistem adı tanımlı değilse veya çözümlenemiyorsa. Bu seçeneğin kullanılması durumunda dikkatli olmak gerekiyor. Zira DNS yapılandırmasındaki bazı hatalar sisteminizin adının tanımlanmasında sorunlar yaratabilir. identd(8) istemcide çalışıyorsa bu sorun çözülür ama çalışmıyorsa UNKNOWN olarak tanımlanır.
KNOWN	İstemcini ip adresi ile sistem adı çözümlenebiliyorsa
PARANOID	Eğer sistem adı ile ip adresi eşleşmiyorsa bu kural geçerlidir. İstemcinin ip adresi ile reverse dns veya dns çözümlemesi wrapper tarafından sorgulanır ve kontrol edilir. Eğer eşleşmiyorsa sunucu bu kuralı eşleştirir. Bu durum ile özellikle DNS sunucularındaki kayıtların saldırganlar tarafından ele geçirilmesi durumunda karşılaşılır.

Bu terimleri kullanarak wrappers yapılandıracak iseniz alan adı sorgulamasının wrapper tarafından değil kendi DNS sunucunuz tarafından yapılacağını, DNS sunucusundan döndürülen sorgulama sonuç-ları ile sistemlerinizin veya diğer istemcilerin isimlerinin/alan adlarının eşleşmesi gerektiğini unutmamanız gereklidir. Eğer DNS sunucunuzda veya diğer sunucularda bir sorun varsa veya ulaşılamaz olmaları durumunda UNKNOWN kuralı geçerli olacaktır. Bu kuralın eşleşmesi durumunda gelen istekler sunucu tarafından yanıtsız bırakılacaktır. Eğer DNS tabanlı bir wrapper kuralı tanımlarsanız bu durumda saldırgan veya saldırganlar DNS sunucunuzu devre dışı bıraktığında sunucunuz gelen isteklere yanıt vermeyecektir. Bir DNS saldırısına kurban giden DNS olsa da tüm sunucularınız devre dışı kalacaktır.

TCP wrappers bunlardan başka terimleri de tanımaktadır. Bu terimler bağlantılara nasıl yanıt verileceğini tanımlayan terimlerdir. Daemon isimlerinde olduğu gibi istemcilere ilişkin sınırlamalar ALL ve ALL EXCEPT ile tanımlanabilir. Örneğin /etc/hosts.allow dosyasında gelen tüm isteklere tüm daemon'ların yanıt vermesini istiyorsanız aşağıdaki kuralı kullanabilirsiniz.

```
ALL : ALL : accept
```

Veya benzer biçimde yukarıdaki kural geçerli olmak üzere belirli bir sistemi engelleyerek diğerlerine erişim izni verebilirsiniz. Örneğin 192.168.1.1 ve 192.168.1.255 ip adresi dışındakileri engelleyebiliriz.

```
ALL : 192.168.1.1 192.168.1.255 : deny
```

TCP wrappers işleyişini dikkate aldığınızda tüm isteklere izin vermek veya engellemek uygulamada sorunlara neden olabilir Wrappers ilk eşleşen kuralı uygulayıp gerisini işlemedikleri için karmaşık yapı-landırmaları uygulamaya koymak daha fazla dikkat gerektirir. Benzer yapılandırmalarda ALL veya ALL EXCEPT kullanmak kuralları yaz-manızı kolaylaştırır. Evdeki sunucuma erişim yetkisini sadece droideka ve tardis sistemleri ile sınırsız kılarken diğerlerine sadece FTP sunucusuna erişim yetkisi vermek istiyorum. Aşağıdaki kurallar

istediğimi yapacaktır.

```
ALL : 192.168.7.2 192.168.7.4 : accept
ftpd : ALL : accept
ALL : ALL : deny
```

tardis ve droideka'nın ip adresleri tanımlıdır ve bu ip gelen erişim istekleri kabul edilirken diğerleri red edilecektir. Ftp sunucusuna erişin kısıtlanmamıştır ve herkes erişebilir ama diğer servislere erişim kısıtlanmıştır.

Yukarıdaki kuralları ALL EXCEPT ile yazabiliriz. Böylece aşağıdaki kuralları elde ederiz.

```
ALL : 192.168.7.2 192.168.7.4 : accept
ALL EXCEPT ftpd : ALL : deny
```

ALL EXCEPT terimi tanımlı olan daemon -örnekte ftpd- dışında kalanlara gelen isteklerin dikkate alınmayacağını belirtir. Wrappers işleyişi gereği kuralların yazılması sırasında bazı sistem yöneticileri ALL EXCEPT yerine sık sık ALL kullanırlar. Bu yaklaşım ilk eşleşen kuralın uygulanıp diğerlerinin işlenmediğini düşündüğünüzde hatalı yapılandırmalara neden olacağını kolaylıkla görebiliriz.

TCP wrappers öğrenirken ilk kural olarak /etc/hosts.allow dosyasına aşağıdaki kuralı birinci kural olarak ekleyin.

```
ALL : localhost : allow
```

Eğer bir hatalı yapılandırma yaparsanız sisteme erişiminizin olmasını sağlar. Erişiminizi kapatırsanız düzeltmek zaman alacaktır.

Diğer Seçenekler

Erişim izinleri allow-izin ver ve deny -red et olarak örneklerde tanımlandı. Birden çok kural tanımlayacaksanız örneklerde gördüğünüz gibi ilk kural tüm daemonlar için geçerlidir. İlk kuralınız allow olabilir. Bundan sonra gelen kurallar da kısıtlamaları tanımlayıp en son kural olarak da

```
ALL : ALL : deny
```

kullanabilirsiniz. Bu tcp wrappers kullanıyorsanız yeterli bir güvenlik önlemi sağlayacaktır.

Kuralların yazımı konusunda dikkat edilecek bir diğer nokta da kısıtlamaları yazarken tek satırda bitirmek olanaklı olmayabileceğidir. Bu durumda bir sonraki satırdan devam etmek için satırın sonuna \ ekleyip alttaki satırdan devam ederek kuralınızı tamamlayabilirsiniz.

Kayıt Tutma

Bağlantılara izin vermek veya vermemek konusunda politikanızı belirledikten sonra isterseniz bağlantıları kayıt altına alabilirsiniz. Örneğin kullanıcılarınızın erişimlerini izlemek isteyebilirsiniz. DNS sunucunuzdaki problemlerin kaynağını belirlemek isteyebilirsiniz. Hatta bu sorundan kaynaklanan ve PARANOID seçeneğini kullandığınız kurallarda red edilen bağlantıları belirlemek isteyebilirsiniz. Bu bilgiler size sorunların kaynağını bulmada yardımcı olur.

Kurallarda tanımlanacak olan severity terimi syslogd(8) tarafından kayıt tutulmasını sağlar. Aşağıdaki kuralda telnet daemon-telnetd gelen isteklerin syslogd tarafından kayıt altına alınmasını sağlayan kuralı görüyorsunuz

```
telnetd : ALL : severity auth.info : allow
```

Bağlantıya Mesaj Göndermek

TCP wrappers ile twist seçeneği kullanılarak sisteme bağlanmak istenildiğinde bir kabuk programını veya komutu çalıştırabilirsiniz. Bu

işlem ise ancak TCP bağlantıları için yapılabilir. UDP vs için söz konusu değildir. Twist tanımladığınız komutu argüman olarak işler ve red işlemine ek olarak da komutun gereğini yapar. Twist kullanmak istiyorsanız kabuk programlama konusunu da bilmeniz de yarar var.

Aşağıdaki örnekte telnet bağlantısını kullanmak isteyen kullanıcılara mesaj yollanmaktadır.

```
telnetd : 192.168.7.2 192.168.7.4 : accept
ALL EXCEPT telnetd : ALL : deny
telnetd: ALL : twist /bin/echo "erisim yetkiniz yok"
```

Aşağıdaki mesajda PARANOID kullanımı ile alt satıra geçiş görülmektedir.

```
ALL : PARANOID : twist /bin/echo \
    "DNS probleminiz var. Daha sonra yeniden deneyin."
```

Bu tür mesajların kullanımı yetkisiz kullanıcıların ısrarla denemesine neden olabilir. Hatta kendi kullanıcılarınız bile bu tür mesajları alabilirler. Bu mesajların içeriğinin ne anlama geldiğini açıklayan bir mesajı yukarıdaki örnekte olduğu gibi açıklayacak şekilde yazabilirsiniz.

Bu tür mesajların kullanılması twist'in tüm komutu işleyene dek bağlantıyı açık tutmasına neden olur. Uzun mesajların kullanılması sunucuda açık olan bağlantı sayısını arttıracağı için hem sunucuya yük olur hem de bant genişliğinizi tüketebilir. Bir saldırganın bu durumu kendi lehine kullanıp sisteminize bir DoS saldırısı düzenlemesi de söz konusu olabilir. Bu nedenle twist kullanacaksanız mesajınızı kısa tutun veya kullanmayın.

Spawn

Bu seçenek kullanıldığında twist aksine uzak sisteme bir mesaj gönderilmez ama tanımladığınız komut çalıştırılır. Özellikle uzak sistemlerin sistemlerden gelen istekleri kayıt altına almak vb işlemler için kullanılabilir.

Örneğin PARANOID seçeneği ile spawn kullanarak uzak sisteme ait olan bilgileri toplayabiliriz. Bu bilgileri toplamak için spawn ile birlikte bazı değişkenlerin tanımlanması gereklidir. Aşağıda değişkenler ile işlevleri gösterilmektedir.

%a	İstemci adresi
%A	Sunucu adresi
%c	İstemciye ait olan tüm bilgiler
%d	Bağlanılan daemon adı
%h	İstemci sistemin adı veya ip adresi
%H	Uzak sunucunun adı veya ip adresi
%n	İstemcinin adı, eğer belirlenemzse UNKNOWN olarak tanımlanır. Eğer ip adresi ile istemci adı uyuşmaz ise PARANOID kuralı tanımlıymış gibi işlem yapılır.
%N	Buseferistemcideğilsunucuiçinyukarıdakikural uygulanır.
%p	Daemon'un süreç numarası (PID)
%s	Sunucuya ait olan tüm bilgi
%u	İstemcideki kullanıcı adı
%%	Tek bir % karakteri yazar.

```
telnetd : 192.168.7.2 192.168.7.4 : accept
ALL EXCEPT telnetd : ALL spawn (/bin/echo %a >>
/var/log/telnet_erisim : deny
```

Yukarıdaki örnekte telnet ile bağlanmaya çalışan istemcilerin adresleri kayıt edilmektedir. Spawn ile yapılan kayıt işlemleri içerisinde / ve \ karakterleri tanımsızdır. Halbuki internet üzerinde bu karakterler bol bol kullanılmaktadır. Bu durumda spawn, / ve \ karakterini “_” karakteri ile yer değiştirir. Kayıt dosyanızda bu durumu dikkate alarak incelemenizde yarar var. Bu tür “_” karaktere sahip olan kayıtlar sisteme izinsiz erişim girişimine işaret edebilir veya kullanıcı kazara

bir işlem yapmış olabilir.

Örnek Bir TCP Wrappers Yapılandırması

TCP wrappers kullanarak yerel bir ağda bulunan bir sunuya erişimi TCP Wrappers ile sınırlayalım. Bu ağda statik ip kullanıldığını IP aralığının 192.168.7.0/24 olduğunu, yerel bir DNS sunucusunun kullanıldığını, dışarıdan sadece POP3, HTTP ve FTP servislerinin erişilebildiğini, yerel ağ üzerinden ise SSH da dahil olmak önceki servislerin erişilebildiğini var sayalım. Bu sunucu için tanımlayacağımız /etc/host.allow dosyasının içeriği aşağıdaki gibi olacaktır.

```
# izin verilmeyen istekler:
# DNS kaydi ile ip tutumuyorsa

ALL : PARANOID : deny

# sunucu servisleri iletisim kurabilir.

ALL : localhost : allow

# servislerin erişim izinleri:
# bu servislere herkes erisebilir.

POP3, httpd, ftpd : ALL : allow

# ssh sadece sysadmin icin izin verilir.

sshd : 192.168.7.12 192.168.7.10: allow

# bunlarin disindakilere izin verilmez.

ALL : ALL : deny
```

Bu basit örnek dışında /etc/hosts.allow dosyanızda yukarıdaki yer verilenleri de kapsayan birçok örnek bulabilirsiniz. Bunlardan başka man (5) hosts.allow(5) ve hosts_access(5) kılavuz sayfalarında daha ayrıntılı bilgi bulunabilir.

TCP Wrappers ve PF

PF ile TCP Wrappers işlevleri birbirinden farklıdır. Birçok sistem yöneticisi ikisini birlikte kullanmayı tercih eder ancak ağ yapılandırmasındaki değişimler vb durumların neden olacağı yapılandırma işlemlerinin getireceği yük ve aynı zamanda servislerin yapılandırılmasında alınacak olan bazı önlemler ile kontrol sağlanabilir. Ayrıca wrapper kullanmış olsanız da pf ile bir servisi engelleyebilirsiniz. Bu nedenle wrappers ile pf ve diğer önlemlerin kullanılması sistem yöneticisinin işini zorlaştıracaktır. Bu nedenle tercihinizi yaparken iş yükünüzü azaltacak ve kontrol ve güvenlik sürecini basitleştirecek olan bir çözüme gitmeniz gereklidir. Uzun sözün kısası wrapper kullanırsanız PF, PF kullanırsanız wrappers kullanmayın. Tercih sizin. Gelecek ayki bölümde güvenlik konusuna devam edeceğiz. Önümüzdeki ayki bölümde JAILS ve diğer ek güvenlik uygulamalarını ele alacağız.

BSD - XVI

Jails

Gökşin Akdeniz
goksin@enixma.org

UNIX'in en eski güvenlik mekanizması chroot'tur. Chroot basitçe bir programı dosya sisteminde bir bölüm içerisinde sınırlandırmaktır. Böylelikle program dosya sistemi üzerinde yer alan diğer programlar ve dosyalardan ayrılmış olu. Bu uygulama named(8) gibi servisler için uygun olurken diğer programlara ve dosyalara erişmek durumunda olan programlar ve servisler için değildir. Örneğin bir e-posta sunucu programını chroot ile çalıştırmak kolay bir işlem değildir. Bunu yapmanız durumunda birçok başka programı vs chroot ettiğiniz dizin altına taşıyıp yeniden yapılandırmanız gerekir. Bu işlemler hem zaman alıcıdır, hem de hatalı yapılandırılmış uygulamaların neden olacağı sorunlar, problemler ile karşı karşıya kalmak kaçınılmazdır.

Sunucu ve /veya barındırma hizmeti veren şirketlerde müşteriler sık sık kullanılan bazı uygulamaların yeni sürümlerinin kurulması, kullanılması, yapılandırmada bazı değişikliklerin ypılması isteklerinde bulunurlar. Bu istekler, eğer müşteri sunucunun işletim sisteminin özelliklerini biliyorsa, teknik bilgisi de varsa hayır diyerek sonlandırılmaz. Bu istekler genelde root yetkileri ile yapılan işlemlerdir. Müşterilere veya diğer kullanıcılara root parolasını vermek söz konusu olamaz. Ancak bir BSD sistemi kullanıyorsanız bu durumda kullanıcılarınızı kendi “evlerinde root” gibi hissetmelerini sağlayabilirsiniz.

FreeBSD geliştiricileri bu sorunlar ile uzun zaman önce karşılaşıp çözüm geliştirdiler. Geliştirilen çözüm chroot'u bir üst seviyeye taşımak oldu. Böylelikle sistemin geri kalanından yalıtılmış, hafif bir sanal sunucu ortaya çıkmış oldu. Bu sunucu sistemine hapishane anlamına gelen “JAILS” adı verildi. Jails, bir anlamda istemci – sunucu yapısı olarak düşünülebilir. Jails kurulumu yapılan sistem sunucu olurken, jails içerisindeki sistem istemci olmaktadır. Jails içerisinde kalan sisteme istediğinizi yapabilirsiniz ve yaptıklarınız asıl sistemi etkilemez. Bu önerme ancak ve ancak Jails içerisinde sistemin diskinizi tüketmesi durumunda geçersizdir.

Kullanıcı açısından bakıldığında jails hemen hemen tam bir BSD sistemdir, sadece bazı donanımlar bulunmaz. Jails içerisindeki sistemdeki kullanıcılar root yetkilerini kullanabilir, istedikleri programı kurabilir ve istedikleri gibi yapılandırabilir. Jails içerisinde çalışan süreçler sadece ve sadece JAILS ortamı/environment ile sınırlıdır ve asıl sistemden yalıtılmış durumdadırlar. Çekirdek/kernel de JAILS içerisinde çalışan süreçlere sadece kendi ortamlarına ilişkin bilgi verir ve jails dışında kalan sisteme ait bir bilgi sunulmaz. Jails içerisindeki dosya sistemi de sadece kendi ortamına ilişkin bilgi sunar, asıl sisteme ait olan dosya sistemi erişilmezdir ve jails içerisindeki sistem tarafından bilinmez. Jails dışında kalan sistem bilinmez/görünmez olduğu için jails içerisindeki süreçler jails dışındaki kaynaklara erişemez.

Diğer bir deyişle jails adının gereğini yerine getirip içerisindeki kullanıcıyı, sistem,, süreçleri vs. hapis eder. Jails içerisindeki sistemi bir saldırgan ele geçirecek olsa bile yapabilecekleri jail ile sınırlıdır. Bu durumda ele geçirilen jail'i kaldırıp yenisini devreye sokabilirsiniz. Bugün için bilgisayar donanımları eskiye göre çok daha ucuz olduğu için yüksek kapasiteli bir sisteme onlarca jail kurup kullanabilirsiniz. Bu durum özellikle sunucu hizmetleri sunan bir şirket için kullanıcılarına "root" yetkisi verip istedikleri servisleri istedikleri gibi kullanabilecekleri bir barındırma hizmeti satabilmek demektir. Hele hele bunu özel bir sunucu kiralama ile paylaşımlı sunucu kullanımı ile karşılaştırırsanız aslında bir BSD sunucunun ticari olarak çok uygun bir seçenek olduğunu görebilirsiniz.

Jail Kurulumuna Hazırlık

Bir jail kurabilmeniz için, jails'i barındıracak olan sistemde bazı düzenlemeler yapanız gerekir. Bu düzenlemelerin yapılmasının gereği sistemde çalışan daemon'ların farklı ip adreslerini dinleyecek şekilde yapılandırılmak zorunda olunmasıdır. Jails kurmadan tek bir sistemde daemonları sadece ve sadece tek bir ip adresini dinleyecek şekilde yapılandırabilirsiniz. Ancak jails için durum farklıdır. Jail içerisinde yer alan daemonlar sistem tarafından kullanılmayan ilgili jail'e atanmış olan bir ip adresine gereksinim duyarlar. Eğer asıl sistemde çalıştırılan bir daemon kullanılabilir olan tüm ip adreslerini kullanıyorsa bu durumda sistemdeki tüm jail başlatılamaz. Bu durumu kontrol etmek için sockstat(1) kullanılarak kontrol edilmelidir. Eğer komutun çıktısında herhangi bir daemon dinlediği ip adresi "*" olarak görünüyorsa bu daemon sistemde bulunan tüm ip adreslerinden gelen çağrıları dinlemektedir.

```
tardis# sockstat -4
USER  COMMAND   PID   FD  PROTO  LOCAL ADDRESS FOREIGN ADDRESS
root  sendmail  1096  4   tcp4   127.0.0.1:25    *:*
root  ntpd      938   20  udp4   :123            *:*
root  ntpd      938   22  udp4   192.168.7.3:123 *:*
root  ntpd      938   25  udp4   127.0.0.1:123   *:*
root  cupsd     880   6   tcp4   127.0.0.1:631   *:*
root  cupsd     880   8   udp4   *:631           *:*
root  syslogd   709   7   udp4   *:514           *:*
tardis#
```

FreeBSD 7.0, 7.1 ve 7.2'de Jails için bu durum söz konusu iken 8.0 için durum farklıdır. 8.0'da bir Jail'in bir ip adresine sahip olması zorunlu değildir. Hatta bir jail birden çok ip adresine sahip olabilir. Bu özellikler vimage desteği ile sağlanmaktadır. Vimage standart çekirdek yapılandırmasında yer almakla birlikte GENERIC çekirdek ile gelmemektedir. Çekirdeğin vimage desteği ile derlenmesi gereklidir. Böylelikle jail yapılandırmasında yukarıda adı geçen ip adresi yapılandırması ve diğer sanal ağ özellikleri aktif kılınmış olur. Ancak vimage, geliştiriciler tarafından halen deneysel kod olarak göz önüne alındığı düşünülürse yukarıda açıkladığımız gibi her bir jail kendisine ait bir IP adresi ile çalıştırılıp kullanılması daha uygun olacaktır. Yazının bundan sonraki bölümleri vimage desteği olmayan GENERIC kernel kullanan sistemler dikkate alınarak hazırlanmıştır. Dolayısıyla vimage ile gerçekleştirilecek olan yapılandırma yazının kapsamı dışında kalmaktadır.

Yukarıda gördüğünüz çıktıda cupsd, ntpd ve syslogd'nin tüm ip adreslerini dinlemekte olduğunu görüyoruz. Bu durumda bu servislerden hangilerinin sunucudan gelen çağrıları dinleyeceğine karar verip ilgili yapılandırmayı düzenlememiz gereklidir. Bir jails sunucusu kurmayı planlıyorsanız bu durumda asıl sisteme ait herhangi bir servisin çalışmaması gereklidir. Jail sunucusunda çalışması gereken bazı servislere gereksinim duyuyorsanız bu durumda bu servisleri çalıştıran "service jails" kurulumları yapmanız gereklidir. Böylelikle bu servisleri de güvence altına alırken hemde bu servisleri de kolaylıkla yönetebileceğiniz bir ortam hazırlamış olursunuz.

Jail bir ip adresine gereksinim duysa da jail içerisinde çalışan programların bu tür bir yapılandırmaya gereksinimleri yoktur. Dolayısıyla

da jail içerisinde çalışan daemon için tüm ip adreslerin gelen çağrıları (“*”) dinlemesinde bir sorun yoktur.

Jails barındıran sunucuda çalışan daemonların yapılandırmasında aşağıdaki yapılandırma örnek olarak kullanılabilir. Jails barındıran sistemin ip adresi olarak 192.168.7.2 kullanılmıştır.

syslogd

Sistem kayıtları açılan bir socket üzerinden log dosyasına aktarılır. Bu socketin sadece syslogd(8) tarafından tanımlanan bir ip adresini dinlemesi sağlanmalıdır. Bunu rc.conf dosyasına aşağıdaki satırı ekleyerek yapabiliriz.

```
syslogd_flags=”-b 192.168.7.2”
```

inetd

Inetd(8) kullanılması pek tercih edilmez ama gene de gerekiyorsa bir jail içerisinden çalıştırılması en uygun çözümdür. Inetd yapılandırması /etc/defaults içerisinde tanımlanan yapılandırma ile çalıştırılır. Ancak kendi yapılandırmanızı da ekleyebilirsiniz. Aşağıda standart yapılandırma görülüyor.

```
tardis# cat /etc/defaults/rc.conf | grep "inetd"
inetd_enable="NO"               # Run the network daemon
dispatcher (YES/NO).
inetd_program="/usr/sbin/inetd" # path to inetd, if you want a
different one.
inetd_flags="-wW -C 60"         # Optional flags to inetd
tardis#
```

Yukarıdaki yapılandırmada inetd aktif kıldığımda ip adresi olarak yukarıda belirttiğimiz flags kısmına ekliyoruz.

```
inetd_flags="-wW -C 60 -a 192.168.7.2"
```

sshd

Sshd yapılandırmasının sadece jail barındıran sunucunun ip adresi olarak yapılması gereklidir. Ssh yapılandırma dosyası içerisine

```
ListenAddress 192.168.7.2
```

Satırını eklediğimizde gereken işlem tamamlanmış olur. Bir jail sunucusunda ideal olan sadece ve sadece sunucuda çalışan ve dışarıya açılan tek servisin sshd(8) olmasıdır.

NFS

rpcbind(8) ve nfsd(8) vb. programlar bir sistemdeki tüm ip adreslerinden gelen çağrıları dinlerler. Bu programların işleyişini değiştirmek zor olduğu için bir jails sunucusu kuruyorsanız bu daemonları kullanmamanız yerinde olacaktır. Öte yandan jails içerisinde NFS kullanabilirsiniz. Böylece sunucda yer alana paylaşımları jail içerisinde çalışan NFS ile uzak sistemlere sunabilirsiniz.

Jails için Çekirdek Parametreleri

Ağ yapılandırmasının tamamlanmasının ardından kurulacak olan jails için çekirdek parametrelerini ayarlayabiliriz. Genel olarak jails için ön tanımlı olarak gelen çekirdek parametreleri yeterlidir. Sistem yöneticisi bu parametreleri uygun gördüğü şekilde değiştirebilir. Bu parametreler sadece sunucu sistem üzerinden değiştirilebilir. Yapılacak olan değişiklikler ise sunucu sistemdeki tüm jails için geçerli olacağı için dikkat edilmesi gereklidir. Bu parametrelerde yapılacak olan değişikliklerin istenmeyen sonuçlara -örneğin saç dökülmesi ve kronik baş ağrısı gibi- neden olabileceğini de bir kez daha anımsatmakta yarar var.

Aşağıdaki çıktılarda FreeBSD 7.2 (workfreebsd) ve FreeBSD 8.0

(tardis) sistemlerindeki çekirdek jail parametreleri ön tanımlı olan değerleri görülmektedir.

FreeBSD 7.2 sistemdeki çekirdek parametreleri:

```
workfreebsd# sysctl -a | grep "jail"
security.jail.jailed: 0
security.jail.jail_max_af_ips: 255
security.jail.mount_allowed: 0
security.jail.chflags_allowed: 0
security.jail.allow_raw_sockets: 0
security.jail.enforce_statfs: 2
security.jail.sysvipc_allowed: 0
security.jail.socket_unixiproute_only: 1
security.jail.set_hostname_allowed: 1
workfreebsd#
```

FreeBSD 8.0 sistemdeki çekirdek parametreleri:

```
tardis# sysctl -a | grep "jail"
security.jail.param.cpuset.id: 0
security.jail.param.host.hostid: 0
security.jail.param.host.hostuuid: 64
security.jail.param.host.domainname: 256
security.jail.param.host.hostname: 256
security.jail.param.children.max: 0
security.jail.param.children.cur: 0
security.jail.param.enforce_statfs: 0
security.jail.param.securelevel: 0
security.jail.param.path: 1024
security.jail.param.name: 256
security.jail.param.parent: 0
security.jail.param.jid: 0
security.jail.enforce_statfs: 2
security.jail.mount_allowed: 0
security.jail.chflags_allowed: 0
security.jail.allow_raw_sockets: 0
security.jail.sysvipc_allowed: 0
security.jail.socket_unixiproute_only: 1
security.jail.set_hostname_allowed: 1
security.jail.jail_max_af_ips: 255
security.jail.jailed: 0
tardis#
```

Bu parametrelerin varsayılan değerleri ile bırakılması yeterlidir ama bazılarını kendi tercihlerimize göre değiştirmemizde sakınca yoktur.

Parametre: security.jail.set_hostname_allowed

Öntanımlı olarak jail içerisindeki root kullanıcısı, jail'a ait olan sistem adını -host name- değiştirebilir. Jails içerisindeki uygulamalar sistem adını kullanarak iletişim kurdukları için jail içerisindeki sistem adının değiştirilmesi jails sunucu sisteminin yöneticisi için sorun yaratabilir. Ön tanımlı değeri 1 iken 0 yaparsanız sistem adı değiştirilemez.

Parametre: security.jail.socket_unixiproute_only

Bir jail standart olarak yerel UNIX soketleri ve IP üzerinden iletişim kurar. Ancak jail içerisindeki bir kullanıcı farklı bir protokol kullanmak isteyebilir ve farklı bir protokol kullanılmasını sınırlayan bir durum söz konusu değildir. Fakat jail sadece IP destekler. Eğer jail içerisindeki programların farklı bir protokol kullanılmasına izin verirseniz bu durumda jail içerisindeki uygulamaların jail dışına çıkmasına da neden olabilirsiniz. Jails sadece IP protokolünü sanallaştırmaktadır. Diğer protokollerin sanallaştırılması söz konusu değildir. Her ne kadar jail içerisindeki uygulamaların IP dışındaki bir protokolü kullanmasının görünüzde bir sakıncası olmadığı düşünülse de bir saldırganın yaratıcılığının boyutlarını tahmin edip bir şey olmayacağını umut etmek sistem yönetimi açısından yapılacak en kötü hatadır! Öntanımlı olan değer 1'dir ve jail için öntanımlı protokol olan IP dışındakilerin kullanımını önler. 0 yapılırsa tüm protokollere izin vermiş olursunuz.

Parametre: security.jail.sysvipc_allowed

Sistem V IPC, bir UNIX standardıdır ve paylaşımlı belleğin kulla-

nılması ile süreçler arası iletişim kurulmasını sağlar. Birbiri ile ilişkili olan programlar bellek üzerindeki bir alanı bilgi paylaşmak için kullanır. Örneğin birçok veritabanı programı IPC kullanır. Her bir jail için ayrı ve paylaşılan bellek ayrılmaz. IPC izin verirseniz bir jail'deki programın diğer jail içerisinde çalışan programlara bilgi sızdırması olasılığını da gözönüne almanız gerekir. Bu nedenle varsayılan değer olan 0 IPC'yi engellerken 1 değerinin atanması izin verir.

Parametre: security.jail.enforce_statfs

Bu değişken, jail içerisindeki kullanıcıların dosya sistemine ilişkin edinebileceği bilgiyi sınırlar. Ön tanımlı değeri 2'dir. Bu değer jail içerisindeki kullanıcıyı jail içerisindeki dizinler ile sınırlar. Böylelikle jail'in disk üzerinde nerede olduğu, bağlanma noktası vb. bilgiler öğrenilemez. Eğer bu değer 0 olarak atanırsa jail içerisindeki kullanıcı sunucu sistemin dosya sistemine ait tüm bilgileri görebilir. Bu nedenle jail içerisindeki kullanıcının sadece ve sadece jail içerisindeki dosya sistemine ait bilgiler dışındakileri görmesine izin verilmemesi gereklidir. Bu değer 2 olarak kalmalıdır.

Parametre: security.jail.allow_raw_sockets

Raw socket – ham soketler, gelen paketleri işlemeden doğrudan ilgili veya istekte bulunan programa yollanmasını sağlar. Ping ve treaceroute gibi programlar ham soketleri kullanır. Jail kullanıcılarınıza güvenmemeniz için yeterince neden olduğu için kullanıcıların ham soketleri kullanmalarına izin vermemek gerekir. Standart değer 0'dır ve ham soketlerin kullanılmasını önler. İzin vermek için bu değerin 1 yapılması yeterlidir.

Parametre: security.jail.chflags_allowed

Standart olarak jail içerisindeki kullanıcılar dosya sistemi özniteliklerini değiştiremezler. Chflags(1) de kullanamazlar. Chflags(1) kullanılması durumunda ancak tek kullanıcı kipinde özniteliklerde değişiklik yapılabilir. Bu durum jail için de geçerlidir. Jail'in de mutlaka tek kullanıcı kipinde başlatılması ve chflags'ın yeniden tanımlanması gerekir. Standart değer olan 0 chflags kullanımına izin vermez. 1 yaparsanız kullanıcılar bunu kullanabilir. Ancak izin verirseniz hatalı yapılandırmaların düzeltilmesi için sistem yöneticisinin müdahale etmesi gerekir. Değiştirmeden önce ruh sağlığınızı düşünün!

Parametre: security.jail.jailed

Bu değer çekirdek parametrelerini düzenlemek için kullanılan sysctl'un sunucu sistemden mi -0- yoksa jail içerisinden mi -1- çalıştırıldığını belirtir. Bu değer her zaman “0” olmalıdır.

Parametre: security.jail.list

Sunucu sistemde bulunan jail listesinin görüntülenip görüntülenmeyeceğini tanımlar.

Temel Jail Kurulumu

Bir jail kurmadan önce sistemde nerede bulunacağına karar vermeniz gerekir. Bir jail sistem yöneticisinin tercihlerine göre /var veya /usr altında yer alabilir. Nerede yapılandıracağınız tamamen size kalmış. Kullanım sırasında bir veya daha çok sayıda jail çok fazla disk alanı tüketebileceği için ayrı bir disk üzerinde de kurulması tercih edilebilir. Aşağıdaki kurulum senaryosunda jail, /usr/jail/jail1 dizini altıda gerçekleştirilmiştir.

Bir jail kurmak, temel olarak BSD sisteminizin kaynak kodunun güncellenip yeniden derlenerek kurulum yapılması işlemi ile aynıdır. Bu konuyu güvenlik ile ilgili olan yazıların tamamlanmasının ardından işleyeceğiz. Bir jail kurulumunda BSD sistemin kaynak kodun derlenmesi ile oluşan ikili dosyaların sırayla jail içerisine aktarılması

işlemi söz konusudur. Sistemin kaynak kodlarını kurmadıysanız sysinstall ile bunları kurmanız, gerekli güncellemeleri yapmanız gereklidir. Kaynak kodu sysinstall ile kuramıyorsanız bu durumda cvsup ile de yapabilirsiniz. Güncel kaynak kodlar /usr/src dizini altında yer alacaktır. Bu dizine geçip ardından aşağıdaki komutu veriyoruz.

```
# cd /usr/src
# make buildworld
```

Bu komutun tamamlanmasının ardından jail içerisine kaynak kodların kurulması gerekecektir. Bunun için DESTDIR seçeneği ile jail'in kurulacağı dizinin tanımlanması gerekir.

```
# make installworld DESTDIR=/usr/jail/jail1
# make distribution DESTDIR=/usr/jail/jail1
```

İkinci komut ile jail içerisinde /etc/, /var dizinlerini oluşturuyoruz. BSD sisteminizi terfi ederken bu basamağa gerek yoktur. Jail kurulumunda ise zorunludur. Bir diğer işlem de device nodes yani donanımlara ait olan dosyaların oluşturulmasıdır. Bu dosyalar sürücüler ve diğer uygulamaların donanım ile standart girdi/çıktı sistem çağrıları kullanılarak etkileşimini sağlar. Bunun için de devfs dosya sistemi kullanılır. Jail içerisinde minimal bir BSD sistem kurduğumuz için bu adım zorunludur.

```
# mount -t devfs devfs /var/usr/jail1/dev
```

Bu adım terminal, rastgele sayı üreteci vb. temel donanımlara gerek duyan uygulamalar için gereklidir. Bu adımın da tamamlanmasının ardından jail kurulumun temel işlemleri tamamlanmış olur. Jail için gerekli olan donanım dosyalara jail içerisinde çalıştıracağınız birçok program tarafından gereksinim duyulur. Bu dosyaların bulunmaması durumunda programlar çalışmayabilir. Yukarıdaki komut ile gereken dosyalar oluşturulur. Bu işlemin ardından söz konusu donanım dosyalarının erişimine ilişkin kuralların devfs.rules dosyası ile tanımlanması gereklidir. Aksi halde jail içerisindeki kullanıcılar ve programlar gereksinim duydukları donanım dosyalarından farklı olanlara da erişebilir. Bunun önüne geçmek için devfs.rules dosyasındaki tanımlı olan kuralları uygun şekilde düzenlemek gereklidir. /etc/defaults dizininde yer alan devfs.rules dosyası gereksinim duyulan kuralları tanımlamaktadır. Bu dosyadaki kurallar temel jail işletimi için yeterlidir.

Bunun ardından jail içerisinde kullanıcıların oluşturulması ve diğer uygulamaların kurulması işlemine geçilebilir.

Jail içerisinde kullanıcıların eklenmesi vs. işlemlerin yapılabilmesi için kurduğunuz jail'in başlatılması gerekir. Bunun için jail(8) aracı kullanılır. Bir jail2i başlatmak için vereceğiniz komut aşağıdaki gibi olacaktır.

```
# jail /jail/dizini jail_adı jail_IP komut
```

Yukarıdaki kurulumda /usr/jails/jail1 dizini kullanılmıştı. Jail'in kendisine ait bir Ip adresi olmadan başlatılması söz konusu olamayacağı için eğer yapmadıysanız jail'in kullanacağı ip adresini alias olarak sitemdeki ağ kartına atmanız gerekir. Eğer yapmadıysanız aşağıdaki komut ile yapabilirsiniz. Kurulum yaptığım örnek jail için ip adresi olarak 192.168.7.3 kullanıyorum. Ağ kartına alias olarak bu adresi de atıyorum.

```
# ifconfig ağ_kartınız alias 192.168.7.3/32
```

Bu işlemin ardından jail'e isim olarak jail1.tardis.elkotek ismini atayacağım. Bu durumda jail'i başlatmak için gerekecek olan komut aşağıdaki gibi olacaktır.

```
# jail /usr/jails/jail1 jail.tardis.elkotek 192.168.7.3
/bin/sh
```

Bu komut ile jail başlatılmış olur ve komut satırına geçersiniz. Bu işlem basamağının tek kullanıcı kipinde sistemi başlatmak ile aynı

olduğu düşünülebilir. Aslında tek kullanıcı kipinde başlatılmamaktadır. Zira bu aşamada jail içerisinde herhangi bir kullanıcı bulunmamaktadır. Yukarıdaki komut sadece jail içerisinde /bin/sh çalıştırmaktadır.

Komut çalıştırılması ile jail'e erişebiliyorsunuz. Jail içerisine baktığınızda minimal bir BSD kurulumda bulabileceğiniz tüm araçlar yer alırken her hangi bir kullanıcı vb. bulunmamaktadır. Jail'i kullanmaya başlamadan önce gerek duyabileceğiniz diğer bileşenleri oluşturmanız örneğin fstab dosyası gibi hazırlamanız gerekir.

Ek Bileşenlerin Kurulması

Birçok program çalışması için /etc/fstab dosyasına gereksinim duyar. Jail kurulumunda /etc/fstab dosyası oluşturulmadığı için bunu kendimiz oluşturmamız gerekir. Dosyanın içeriğinde bir şey yazmasına gerek olmadığı için boş olarak oluşturulması yeterlidir.

```
# touch /etc/fstab
```

DNS Ayarları

DNS yapılandırmasına gereksinim duyacağınız için bu dosyayı doğrudan sunucu sistemden kopyalayarak alabiliriz. Bu jail içerisindeki uygulamaların sunucu sisteminin /etc dizinine erişemeyeceği için yapılması zorunludur.

```
# cp /etc/resolv.conf /usr/jail/jail1
```

Sendmail Ayarları

İlk etapta sendmail(8) yapılandırmanıza gerek olmayacaktır. Ancak “out-of-date aliases” uyarıları alabiliriz. Bu uyarıları sonlandırmak için newaliases(8) çalıştırmak yeterli olur.

```
# newaliases
```

komutun sona ermesi ile artık hata mesajları döndürülmeyecektir.

/etc/rc.conf Yapılandırması

Jail'in başlatılması sırasında çalıştırılacak olan servisler için rc.conf yapılandırılması zorunludur. Aşağıdaki satırları jail içerisindeki rc.conf dosyasına ekliyoruz.

```
network_interfaces=""
sshd_enable="YES"
```

Ağ kartını jail içerisinden yapılandırmak söz konusu değildir. Bu nedenle ilgili olan satırı boş olarak tanımlamak gereklidir. Jail'e sadece ağ üzerinde nssh ile erişebileceğiniz için sshd başlatılması zorunludur.

Root Şifresi ve kullanıcı Hesapları

Jail içerisinde herhangi bir kullanıcı hesabı bulunmadığı için bu hesapları kendimiz oluşturuyoruz. Passwd(1) ile root şifresini atayıp ardından bir normal kullanıcı hesabı oluşturabilirsiniz. Bunun için adduser(8) ile bir tane normal kullanıcı hesabı oluşturun.

Diğer Yapılandırma Seçenekleri

Jail içerisinde çalışacak olan programlar için ayrıca zaman dilimini de yapılandırmanızda yararlı olur. Bunun için tzsetup(8) yeterli olmaktadır. Ayrıca gerekiyorsa ports ağacını da jail içerisine kurup buradan gerek duyduğunuz ek programları da kurabilirsiniz. Hatta kendi ev dizininizden kişisel dosyalarınızı da jail içerisine taşıyabilirsiniz. Ancak bu işlemler bu aşamada yapılması gereken işlemler değildir. Kullreklidir. Benim yaptığım kurulumda jail1 olduğu için aşağıdaki satırı da ekliyorum.

```
jail_list="jail1"
```

Eğer birden çok sayıda jail kurduysanız bu durumda her birini aralarında boşluk bırakarak eklemeniz gerekir. Aşağıdaki satır bu duruma örnek vermektedir.

```
jail_list =”jail1 jail2 www dns ftp”
```

Aslında jail_enable=”YES” satırını eklediğinizde diğer satırları eklemenize gerek yoktur. Sistem başlatıldığında veya jail servisini başlattığınızda sistemde kurulu olan tüm jail taranacak ve bulunan her bir jail sırayla başlatılacaktır. Bu süreci kısaltmak için jail_list tanımlaması yapılması yararlı olmaktadır. Böylelikle tüm jail'ler otomatik olarak başlatılmak yerine tanımlı olan jail'ler başlatılabilir. Bunlara ek olarak her bir jail için gereken rc.conf yapılandırmasının da tanımlanması zorunludur. Böylelikle BSD sisteminiz söz konusu jail için gerçekleştirilecek yapılandırma ve başlatma sürecini sizin tanımladığını biçimde gerçekleştirecektir.

```
jail_jail1_rootdir="/usr/jails/jail1"
jail_jail1_hostname="jail.tardis.elkotek"
jail_jail1_ip="192.168.1.3"
jail_jail1_devfs_enable="YES"
jail_jail1_devfs_ruleset="devfsrules_jail"
```

Yukarıdaki yapılandırma jai1.tardis.elkotek jail'i için tanımlanan yapılandırmayı göstermektedir. Bu yapılandırma dosyası esas alınarak -varsa- diğer her bir jail için yapılandırma tanımlanabilir. Her kurduğunuz jail için isim, ip adresi ve root dizini tanımlanması zorunludur. Bu nedenle yapılandırmada bu satırların bulunup bulunmadığının kontrol edilmesi gerekir. Ayrıca /etc/defaulst/rc.conf dosyası içerisinde temel jail işletimi için gereken yapılandırma tanımlıdır. Bu yapılandırma kendi gereksinmelerinize göre düzenleyip /etc/rc.conf dosyasına kayıt ederek kullanılabilir.

Jail Başlatmak ve Sonlandırmak

Sisteminizi yeniden başlattığınızda hazırladığınız yapılandırmanın gereği yerine getirilip kurduğunuz her bir jail çalıştırılacaktır. Ancak jail başlatmak için sistemi yeniden başlatmaya gerek yoktur. Tüm kurulu jail'leri başlatmak için

```
# /etc/rc.d/jail start
```

Jail1 başlatmak için ise aşağıdaki komutu vermek gereklidir.

```
# /etc/rc.d/jail start jail1
```

Jail başlatılmış olacaktır. Bu aşamadan sonra ssh ile jail'e erişebilirsiniz. Ssh ile giriş yaptıktan sonra gereksinmelerinize göre programları kurup, kaldırabilir, bir BSD sistemde yapabildiğiniz her şeyi gerçekleştirebilirsiniz.

Jail'deki root ve normal kullanıcı hesapları ile sunucu sistem üzerindeki bir dizine erişmeniz söz konusu değildir. Hatta sunucuda çalışan bir süreci görmek veya kontrol etmeniz dahi söz konusu değildir. Sanki normal bir BSD sisteme uzaktan bağlanmaktan bir farkı olmayacaktır. Ports ağacını kurup, PERL gibi güçlü araçları kullanarak bile sunucu sistemde bir yük oluşturmanız veya zorlamanız söz konusu değildir. Jail'in kısıtlı ve kontrolü dünyasında sadece elinizdekiler ile jail içerisinde sınırlı bir dünyada kalırsınız.

Bazı kullanıcılar jail içerisinde cvsup kullanıp kaynak kodları indirip yeniden make buildworld kullanarak sistem derlemeye de kalkabilir. Bu işlem yine sunucu sisteme bir zara vermez. Ancak jail içerisindeki sistem ile sunucu sistemin “kod” olarak uyumlu olması gereklidir. Uyumsuz olması durumunda sunucu sistem etkilenmez ama jail içerisindekiler “pek beklediğiniz gibi” çalışmayacaktır.

Jail Yönetimi

Jail kullanımı güvenlik açısından büyük katkılar sağlamakla birlikte bir sistemde süreç yönetimini zorlaştırmaktadır. Sunucu sistemde baktığınızda tüm süreçleri bunlara da jail içerisindekiler de dahil olmak üzere görebilirsiniz. Sorun ise bu süreçlerin hangisinin jail hangisinin sunucuda çalıştığının ayrımının yapılmasının gerekmesidir.

Eğer sisteme az sayıda jail kurduysanız ve her birinin hangi amaçla kurulduğunu biliyorsanız "ps -ax" çıktısına bakıp süreçler hakkında bir tahminde bulunabilirsiniz. Komutun çıktısında STATS sütununda göreceğiniz J değeri sürecin jail içerisinde çalışmakta olduğunu belirtmektedir. Bu durum çok sayıda jail olduğu durumda tahmin etmek doğru araç değildir. Bunun yerine jls(8) ve jexec(8) kullanılarak jail içerisindeki süreçler izlenebilir.

jls(8) uygulaması sistemde çalışmakta olan tüm jail'lerin listesini döndürür.

```
tardis# jls
   JID  IP Address   Hostname              Path
   1    192.168.7.3  jail1.tardis.ekotek   /usr/jails/jail1
   2    192.168.7.4  jail2.tardis.elkotek  /usr/jails/jail2
```

her jail bir ID değeri ile tanımlanır. Eğer bir jail'i sonlandırıp yeniden başlatırsanız ve yeniden jls komutunu çalıştırsanız JID sütununda göreceğiniz değerler değişecektir. Dolaysıyla bir önceki sorgulamada JID değeri 2 ise daha sonra bu değer 3 veya 4 olarak döndürülebilir. JID değeri değişse de her jail kendisine ait olan isim (Hostname) ve dizini değişmez (Path)

jexec(8) aracı sunucu sistemdeki yöneticiye seçili jail içerisinde, söz konusu jail'e giriş yapmak durumunda kalmadan belirttiği komutun çalıştırılmasını sağlar. Bu şekilde jail'i kullanan kullanıcıyı rahatsız etmeden jail için atanan root şifresini bilmemize ve sisteme giriş yapmamıza gerek kalmadan gereken işlemleri yapmamızı sağlar. jexec(8) kullanabilmek için jls(8) ile söz konusu jail'a ait olan JID değerini öğrenmemiz zorunludur. Örneğin JID değeri 1 olan jail içerisinde ps -ax komutunu çalıştırmak için aşağıdaki komut kullanılır.

```
# jexec 1 ps -ax
```

Komutun çıktısı jail içerisinden çalıştırılan ps-ax ile aynıdır. jexec(8) ile istediğiniz komutu istediğiniz jail içerisinden çalıştırabilirsiniz. Böylelikle jail içerisinde program kurabilir, bir servisi sonlandırabilir veya yeniden başlatabilir, hatta kullanıcıların şifrelerini de değiştirebilirsiniz.

Procfs Dosya Sistemi ve Süreç Yönetimi

Procfs dosya sistemi, sistemdeki tüm süreçlerin bilgisinin tutulduğu dosya sistemidir. Güncel işletim sistemlerinde bu dosya sistemine gerek duyulmamakla birlikte bazı uygulamaların gereksinim duyması ile kullanılır. BSD sistemlerde procfs dosya sistemi /proc altına bağlanır. /etc/fstab dosyasında bu girdinin bulunması durumunda sistem tarafından bağlanır. Procfs dosya sisteminin tüm süreçlere ait bilgiyi barındırması ve geçmişindeki güvenlik zaafiyetleri nedeni ile güncel BSD sistemlerde fstab dosyasında girdi olarak yer almaz. Procfs dosya sisteminin bir BSD sistemdeki en çok işe yaradığı yer bir jail içerisinde yer alan bir sunucu uygulamasının sistem kaynaklarını tüketmesi gibi bir durumda sorunun kaynağının çabuk ve kolay yoldan belirlenmesidir. Söz konusu sürecin PID değeri /proc dosyasında tanımlıdır. Bu değere bakılarak söz konusu sorunlu uygulamanın yer aldığı jail belirlenebilir.

/proc dizini altında bağlanmış olan procfs dosya sistemine göz attığınızda çalışan her sürece ait olan bir dizin bulunduğu görülür. Bu dizinler süreç numaralarıdır. Sorunlu sürecin bulunduğu jail'i belirlemek için sürecin PID değerini belirleyip /proc/süreç_no dizinine geçin. Bu dizinde status adlı bir dosya bulunmaktadır. Bu dosyanın içeriğine göz attığınızda son girdinin söz konusu sürecin bulunduğu

jail'in adının olduğunu görebilirsiniz. Eğer söz konusu süreç bir jail içerisinde çalışmıyorsa bu durumda isim yerine bir “tire” işareti bulunur.

Jail'i Sonlandırmak

Sunucu sisteminizi kapattığınızda bu sistemde bulunan tüm jail'ler de sistemle birlikte sonlanır. Aslında bir jail'i sonlandırmak için sistemi kapatmanıza gerek yoktur. Nasıl her bir jail veya bir tanesini başlatıyorsak sonlandırma işlemlerini de aynı şekilde yapabiliriz. Sistemdeki tüm jail'i aynı anda sonladırmak için aşağıdaki komutu kullanın.

```
# /etc/rc.d/jail stop
```

Ancak belirli bir jail'i sonlandırmak için örneğin jail1'i sonlandırmak istiyorum

```
# /etc/rc.d/jail stop jail1
```

Komutunu kullanmak yeterlidir. Bu komutlar jail içerisindeki standart sonlandırma sürecini başlatır. Sürecin tamamlanması ile de jail sonlandırılmış olur. Diğer bir deyişle “kapanır”. Sonlandırılan bir jail artık jls komutunun çıktısında görülmez. Bir BSD sistemi sonladırmak için kullanacağınız shutdown ve reboot bir jail için kullanılmaz. Bu araçların asıl görevi bellek ve diskleri eş zamanlamak, ağ bağlantılarını sonlandırmak, çalışan süreçleri sonlandırıp diskleri ayırıp sistem kapatma veya yeniden başlatmaktır. Sanal bir sistemin bu işlemleri yerine getirmesine gerek yoktur.

Service Jails – Servisler için Jail

Jail, kurarken bir diğer seçenek sisteminizde çalıştıracağınız servisleri barındıracak servis jail'i kurmaktır. Servis jail kurulumu normal bir jail kurulumundan farklı değildir. Yapılandırılmasında çalıştırılacak servisin jail içerisinde çalıştırılması ve birden çok servisin jail içerisine alınmasından kaynaklanan bakım ve güncelleme zorluklarının üstesinden gelinmesi yatmaktadır.

Sistemde kurulan her jail disk üzerinde yer kaplamasının yanında sistemin kaynak kodu ile jail içerisindeki sistemin kaynak kodunun aynı olmasını gerektirir. Yayınlanan güvenlik yamalarının kurulması ile sunucu sistem güncellenirken, jail içerisindeki sistemlerin de güncellenmiş kaynak kod kullanılarak yeniden kurulması gerekir. Bir tane jail kurulu olan sistemlerde bu süreç sorun olmayabilir ama birden çok jail kurulu olması durumunda zaman alıcı ve zor bir işe dönüşecektir.

Bu yönetim sürecinin zorluklarının aşılması için sunucu sistemde kurulu olan tüm jail arasında ortak bileşenlerin nullfs dosya sistemi ile salt-okunur olarak paylaşılıp, her bir sistem servisinin de kendisine ait olan jail içerisinden çalıştırılması tercih edilir. Bu yapılandırmaya servis jail yapılandırması adı verilir. Böylelikle gerektiğinde yeni servislerin sunucuda jail içerisine alınıp çalıştırılması kolaylaşır. Bu aynı zamanda tüm jail güncellenmesini de kolaylaştıracaktır.

Bir veya daha çok servis jail kurulumu yapmak için her bir jail tarafından kullanılacak olan bir temel sistem gereklidir. Tüm jail bu temel yapıyı kullanacağı için öncelikle temel sistemi nereye kuracağınıza karar vermeniz gereklidir. Yukarıdaki örneklerde olduğu gibi temel yapıyı /usr/jail/temel altında oluşturarak başlıyoruz. Diğer kurulacak olan jail ise /usr/jail/ altında jail1, jail2 veya www, dns vb. isimler ile kurulabilir.

/usr/jail/temel dizini minimal bir BSD sistem kurulumu yapılacak olan dizin. Jail kurulumu için sistem kaynak kodunu yukarıda anlatıldığı gibi derledikten sonra temel dizini altında kurularak temel sistemin kurulumu tamamlanmaktadır.

```
# mkdir /usr/jails/temel
# cd /usr/src
# make installworld DESTDIR=/usr/jails/temel
```

Temel sistem kurulumu tamamlandıktan sonra ports ağacı ve FreeBSD kaynak kodunu barındıran dizinlerin oluşturulması gereklidir. Bu işlemler mergemaster ile yapılandırma dosyalarının oluşturulması için gereklidir. /usr/ports dizinini oluşturduktan sonra ports ağacının güncel sürümü indirilip kurulmalıdır.

```
# cd /usr/jails/temel
# mkdir /usr/ports
# portsnap -p /usr/jails/temel/usr/ports fetch extract
```

FreeBSD kaynak kodunun da cpdup ile kopyalanmasında yarar var. cp ile kopyalanabilse de, büyük miktarlarda dosyanın aktarılması sırasında oluşabilecek bir hatanın önüne geçilmesi için cpdup kullanılması uygun olacaktır. Kurulumu ports ağacı üzerinde sysutils/cpdup üzerinde yapabilirsiniz.

```
# cpdup /usr/src /usr/jails/temel/usr/src
```

Kaynak kodun kopyalanmasının ardından ise temel sistem ile diğer kurulacak olan jail arasında paylaşımlı olarak kullanılacak olan sistem bileşenlerinin kopyalanacağı dizinlerin oluşturulması gerekir. Sistemdeki temel yapılandırma dosyalarının yer aldığı skel dizininden yola çıkarak bu dosyalar /usr/jails/temel/skel vs olarak oluşturulmalıdır.

```
# mkdir /usr/jails/temel/skel /usr/jails/temel/skel/home
/usr/jails/temel/skel/usr-X11R6
/usr/jails/temel/skel/distfiles
```

Bu aşamadan sonra /usr/jails/temel atında yer alan dosyaları skel dizinine kopyalayabiliriz. Bu dosyalar diğer jail kurulumları ile ortak kullanılacak olan dosyalardır.

```
# mv etc /usr/jails/temel/skel
# mv usr/local /usr/jails/temel/skel/usr-local
# mv tmp /usr/jails/temel/skel
# mv var /usr/jails/temel/skel
# mv root /usr/jails/temel/skel
```

Bu adımdan sonra mergemaster ile eksik olan yapılandırma dosyalarının kurulması gerekir.

```
# mergemaster -t /usr/jails/temel/var/tmp/temproot -D
/usr/jails/temel -i
```

mergemaster birçok dizini yeniden oluşturacak. Ancak bu dizinlere ve dosyalara gereksinim olmadığı için silinebilirler.

```
# cd /usr/jails/temel/skel
# rm -R bin boot lib libexec mnt proc rescue sbin sys usr dev
```

/usr/jails/temel dizininde kurulan jail asıl bileşenleri barındırmaktadır. Bu dizinde yer alan dosyaların diğer kurulacak olan jail ile paylaşımlı olarak kullanılabilmesi için salt okunur olarak diğer jail dosyalarına bağlanması uygun olur. Ancak asıl dosyalar ve dizinler oku-yaz olarak bağlandıkları için ek bir güvenlik önlemi olarak salt okunur olarak bağlanacakları bir dosya sistemine sembolik bağlar ile bağlanabilirler. Böylelikle asıl dosyalarda gerekli düzenlemeler yapıldığında diğer jail için de geçerli olacaktır. Salt okunur dosya sistemi sembolik bağlar ile bağlanan asıl dosya ve dizinlerin bir yansısı olacağı için buna uygun bir isme sahip olmalıdır. Genelde sembolik bağları ve salt okunur çağrıştırması için s adlı bir dizin kullanıyorum. Bu dizine dosya ve dizinleri sembolik bağlar ile bağlarken aynı dizin ve dosya isimlerini kullanmanız gereklidir. Farklı veya hatalı tanımlanan isimler jail çalışmasını önleyecektir.

```
# cd /usr/jails/temel
# mkdir s
# ln -s s/etc etc
# ln -s s/home home
# ln -s s/root root
# ln -s ../s/usr-local usr/local
# ln -s ../s/usr-X11R6 usr/X11R6
# ln -s ../../s/distfiles usr/ports/distfiles
# ln -s s/tmp tmp
```

```
# ln -s s/var var
```

Bu işlemlerin ardından ports ağacını kullanarak kurulacak olan uygulamaların her bir jail içerisinde derlenerek kurulabilmesi için /usr/jails/temel/skel/etc/make.conf dosyasının da düzenlenmesi gerekir. Dosyanın içeriğine

```
WRKDIRPREFIX?=  /s/portbuild
```

yazılması yeterlidir. Bu girdi ports ağacının salt okunur dizin olmasına karşılık her bir jail içerisindeki oku-yaz olan dosya sistemi üzerinde port ağacından kurulum yapılmasına olanak vermektedir.

Servisler İçin Jail Kurulumu

Bu aşamaya kadar jail ile kullanabileceğimiz "temel" sistemi oluşturduğumuz için /etc/rc.conf ve /etc/fstab dosyalarında gereken düzenlemeleri yaparak yeni jail kurmaya başlayabiliriz. Aşağıdaki örnek yapılandırmada web ve alan adı sunucularının barındırıldığı iki jail tanımlanmıştır. Aşağıdaki yapılandırma ile salt okunur ve oku-yaz olarak bağlanan iki dosya sistemi oluşturulmaktadır. Nullfs dosya sistemi kullanıldığı için fsck tarafından kontrol edilmelerine ve dump ile yedeklenmelerine gerek olmadığı için fstab girdisinde dump ve pass sütunlarında "0" olarak belirtilmiştir.

```
# Device    Mountpoint  Fstype      Options     Dump  Pass#
/dev/ad4s1b none        swap        sw          0     0
/dev/ad4s1a /           ufs         rw          1     1
/dev/ad4s1e /tmp        ufs         rw          2     2
/dev/ad4s1f /usr        ufs         rw          2     2
/dev/ad4s1d /var        ufs         rw          2     2
#/dev/acd0  /cdrom      cd9660      ro,noauto   0     0
proc        /proc       procfs      rw          0     0
/usr/jails/temel/ /usr/jails/dns nullfs ro      0     0
/usr/jails/temel  /usr/jails/www nullfs ro      0     0
/usr/jails/sj/dns /usr/jails/dns/s  nullfs rw   0     0
/usr/jails/sj/www /usr/jails/www/s  nullfs rw   0     0
```

Bu aşamanın ardından /etc/rc.conf dosyasına jail'e ilişkin bilgileri tanımlamak gerekiyor. Bu aşamanın farklı bir tarafı yok. Aşağıda rc.conf dosyasında ilgili jail için örnek yapılandırma görülüyor

```
jail_enable="YES"
jail_set_hostname_allow="NO"
jail_list="dns www"
jail_dns_hostname="dns.elkotek"
jail_dns_ip="192.168.7.2"
jail_dns_rootdir="/usr/jails/temel/dns"
jail_dns_devfs_enable="YES"
jail_www_hostname="www.elkotek"
jail_www_ip="192.168.7.3"
jail_www_rootdir="/usr/jails/temel/www"
jail_www_devfs_enable="YES"
```

Kullanılacak olan jail için gerekli olan bağlanma noktalarını oluşturalım.

```
# mkdir /usr/jails/dns /usr/jails/www
```

Bu işlemin ardından cpdup ile dosyaların her bir jail için söz konusu dizinlere kopyalanması gereklidir.

```
# mkdir /usr/jails/sj
# cpdup /usr/jails/temel/ /usr/jails/sj/dns
# cpdup /usr/jails/temel/ /usr/jails/sj/www
```

Dosyaların kopyalanmasının ardından dosya sistemlerinin bağlanarak jail çalıştırılabilir.

```
# mount -a
```

Dosya sistemlerini bağladıktan sonra jail'i başlatabiliriz.

```
# /etc/rc.d/jail start
```

Jail başlatıldıktan sonra jls(8) ile çalışan jail listesini döndürdüğümüzde çalışan jail'lerin bir listesini görebiliriz. Yapılandırmanıza bağlı olarak aşağıdaki çıktıya benzer sonuç görebilirsiniz.

```
# jls
  JID  IP Address      Hostname           Path
    2  192.168.7.2     ns.elkotek   /usr/jails/dns
    1  192.168.7.3     www.elkotek  /usr/jails/www
```

Bu aşamadan sonra her bir jail'e giriş yapabilir, kullanıcı ekleyebilir ve ilgili diğer işlemleri yapabilirsiniz.

Jail Terfisi

BSD sistemin yeni sürüme güncellenmesi durumu bir güvenlik yaması, bazı yeni özelliklerin sistemde kullanılmasına karar verilmesi veya yeni BSD sürümüne terfi edilmesi durumunda söz konusudur. Bu durumlarda var olan jail'e terfi edilmesi gerekir. Burada anlatılan jail kurulumu ve yapılandırması sunucu sistemin güncellenmesinin ardından eski sistemden kalan jail'lerin sadece yeni sisteme terfi edilmesinin ardından yeniden başlatıldığında geçici olarak kapatılıp ardından yeniden çalıştırılmasını gerektirir. Böylelikle servislerin çalışmadığı zaman dilimi asgari düzeyde gerçekleşir.

Sunucu sistemin güncellenmesinin ardından yeniden başlatıldığında sistemde kurulu olan jail'in kapalı olması gereklidir. Yeni derlenmiş olan sistemdeki kodlar bir jail kurulumunda olduğu gibi işlem görecektir. Bu sefer terfi edilecek olan jail için yeni “temel” sistem /usr/jails/temel yerine /usr/jails/temel_guncel vb. bir dizine kurulabilir.

```
# mkdir /usr/jails/temel_guncel
# cd /usr/src
# make installworld DESTDIR=/usr/jails/temel_guncel
# cd /usr/jails/temel_guncel
# cpdup /usr/src usr/src
# mkdir s
```

make installworld komutunun oluşturduğu dizinlerden bir çoğuna gereksinim olmayacaktır. Bunları silebilirsiniz.

```
# chflags -R 0 var
# rm -R etc var boot usr/local tmp
```

Gerekli olan dosyalar kaldığı için yeniden asıl sistemden paylaşımlı kullanılacak olan dosyalar ve dizinlere sembolik bağlar oluşturulmalıdır.

```
# ln -s s/etc etc
# ln -s s/root root
# ln -s s/home home
# ln -s ../s/usr-local usr/local
# ln -s ../s/usr-X11R6 usr/X11R6
# ln -s s/tmp tmp
# ln -s s/var var
```

Öncelikle oku-yaz olarak bağladığımız dosya sistemleri ile bunlara sembolik bağla bağlı olan dosya sitemlerini ayırmamız gerekir. Önce asıl sonra sembolik bağlı olanlar sırası ile ayırma işlemi gerçekleştirilir.

```
# umount /usr/jails/dns/s
# umount /usr/jails/dns
# umount /usr/jais/www/s
# umount /usr/jails/www
```

Bu işlemin ardından paylaşılan jail dosyaları güncelleneceği için disk alanından tasarruf etmek amacı ile ports ağacını silebiliriz. Güncelleme işleminin tamamlanmasının ardından yeniden kurabiliriz. Eski “temel” yapılandırmamızı kaldırmamıza gerek yok. Bu yapılandırmaya ait dosyalar yedek olarak bulundurmak gerektiğinde “geri dönmek” için kullanılabilir. Yedek alırken eski yapılandırmayı ayırt etmek için size anlamlı gelen bir isim kullanmak uygun olacaktır. Örneğin 7'den 8'e terfi anlamına gelen temel.78 veya yapılan güvenlik güncellemelerine işaret eden temel.78p0, temel.8p1 vb gibi.

```
# cd /usr/jails
# mv temel temel.8p1
# mv temel_guncel temel
# mv temel_guncel/usr/ports temel/usr
```

Yeni “temel” sistem ile bu sisteme ait olan port ağacı da kurulduktan sonra gerekiyorsa var olan servisleri terfi ederek kurulumlarını gerçekleştirebilirsiniz. Bu işlemlerin ardından salt-okunur dosya sistemi hazırlanmış oluyor. Yapılması gereken yeniden dosya sistemlerini bağlayıp jail'i başlatmaktır.

```
# mount -a
# /etc/rc.d/jail start
```

Jail başlatma işleminin sorunsuzca gerçekleşip gerçekleşmediğini kontrol etmek için jls(8) ile kontrol etmek gereklidir. Güncelleme sorunsuz biçimde gerçekleşmiş ise bu durumda sistemde kurulu olan tüm jail jls(8) ile listelenecektir. Bu işlemden sonra her bir jail içerisinde mergemaster çalıştırılması gerekli olan yapılandırma dosyalarının güncellenmesi gereklidir. Böylelikle jails terfi süreci tamamlanmış olur.

Jail kullanıcı birçok güvenlik ve kaynak kullanımı sorununa kesin çözüm gibi görünebilir. Ancak yukarıda anlatılan standart ve servis jail kurulumları sistem yönetimi bilgisini gerektirmektedir. Yapılandırma ve kurulum vb. işlemlerin root yetkilerine gerek duyması nedeni ile yapılacak işlemlerin dikkatle yapılması gereklidir. Hatalı bir komut veya yapılandırma jail'in çalışmamasına veya istenmeyen bir duruma va farklı sorulara neden olacağı unutulmamalıdır. Bu nedenle jail yapılandırması ve yönetimi için farklı jail araçları örneğin sysutils/ezjails kullanılabilir. Yukarıdaki temel jail ile diğer kurulan jail arasında dosyaların paylaşımlı kullanılması ezjails ile yapılabilir. Ezjails, jail yönetimini kolaylaştırsa da sorumluluğu azaltmamaktadır. Sistem yöneticisinin gereken ayarları yapması zorunludur.

BSD - XVII

mtree ile sistem bütünlüğünün kontrolu

Gökşin Akdeniz
goksin@enixma.org

Bir sisteme yetkisiz kişilerin erişebildiği ve bu sisteme giriş yapıp sistem üzerinde oynadıkları düşüncesi rahatsız edicidir. Bu durumu nasıl kanıtlayabileceğiniz ise en zor olan kısmıdır. Böyle bir durumdan şüpheleniyorsanız, şüphelerinizi haklı çıkaracak olan, yani sistemin yetkisiz kişilerce erişildiğinin işaretleri /tmp dizininde tanımadığınız dosyaların bulunması veya /usr/local/sbin altında yeni komutların yer alması vb. durumlardır. Sistem sık sık yeniden başlıyorsa birileri sisteme erişmeye çalışıyor veya kendi derledikleri çekirdeği kurup sistemi bununla başlatıyor da olabilir. Daha kötüsü de sistemdeki derleyicinin saldırganın kendi hazırladığı derleyici ile değiştirilmiş olmasıdır. Böylelikle her derlediğiniz kod saldırganın istekleri doğrultusunda derlenecektir.

Yukarıdaki senaryoları düşünmek kuşkuyu daha arttıracağı gibi böyle bir durumun söz konusu olmadığını kanıtlayabilmenin de oldukça zor olmasıdır. Sisteme yetkisiz giriş yapanlar kendi izlerini gizlemek için gereken her şeyi yapacaktır. Hatta kendilerini gizlemek için kullandığınız araçları da değiştirip bulunmalarını önleyebilirler. Bu durumda yetkisiz bir girişin gerçekleşip gerçekleşmediğini belirlemek olanaksızdır. İşletim sistemlerinin yapısının karmaşıklığını ve yazılımların ise daha da karmaşık olduğunu düşündüğünüzde /tmp dizininde rastladığınız dosyanın aslında bir uygulama ile ilişkili olması da söz konusudur. Öte yandan bir yetkisiz girişin izi olmadığı da kesin değildir.

Eğer sisteme yetkisiz bir giriş söz konusu ise bu durumu düzeltmenin kesin yolu her şeyi baştan kurmaktır. Bu kurulumda elinizdeki güvenilir ortamı kullanarak -bu durumda elinizdeki -CD/DVD seti- ile daha önceden aldığınız yedekleriniz olacaktır. Ancak bu kesin çözüm olmayacaktır. Zira elinizdeki CD/DVD seti yayınlandıktan sonra yamalar yayınlanmış olacağı için yamaları yeniden kurup yetkisiz girişe neden olan açığın kapatıldığını umut etmek durumunda kalacaksınız. Önceki bölümlerde ele alınan güvenlik önlemleri yetkisiz girişleri kesin olarak önlemeyecektir ama buna niyetlenenlerin işini çok ama çok zorlaştıracaktır. Yetkisiz girişlerin önünü almak için nihai bir çözüm, reçete bulunmadığı için birilerinin bunu deneyeceği gerçeği ile yaşamaktan başka çareniz yoktur. Zamanla alışırsınız. En kötüsü ise bana bir şey olmaz düşüncesini kendi kendinize telkin edip kendinizi kandırmaktır veya ne olursa olsun diyerek işler ters gittiğinde her şeyi baştan kurmak gibi bir süreci olağan kabul ederek günlük yaşamanıza dönmektir. Bunları yapmayın!

Birçok işletim sisteminde saldırganlar sistemdeki dosyaları değiştirip yetkisiz girişlerin önünü açar veya izlerini kaybettirmeye çalışırlar. BSD sistemlerde ise mtree(1) sistemin çalıştırıldığı zamanlardaki dosya izinlerini, boyut-larını ve zaman damgasını md5, sh1 ve ck formlarında

saklayarak sonradan çalıştırıldığında bu bilgiler arasındaki farkları karşılaştırmanızı sağlar. Böylelikle olası bir sorunu belirlemeniz kolay olacaktır. Sistemi ilk kurduğunuzda mtree ile kaydını aldığınızda sistemin ilk anındaki durumunu kesin olarak bilmeniz olanaklı olacaktır. Eğer bir yetkisiz girişten şüpheleniyorsanız yeni bir mtree taraması yaparak iki dosyayı karşılaştırabilirsiniz.

mtree(1) İle Sistem Kontrölü

mtree genel olarak / disk bölümü üzerinde çalıştırılmalı. / dizininden başlayarak diğer dizinleri de kapsayacak biçimde bir tarama yapabilirsiniz. Ancak sunucularda tüm dizinlerin taranması zaman alıcıdır. Ayrıca bazı dosyalar ve dizinler sık sık değişecektir. Örneğin web sunucularında kullanıcı dizinleri, veritabanı sunucularında ise veritabanının bulunduğu dizinler ve ilgili dosyalar sık sık değişeceği için bu dizinlerin mtree ile taranması uygun olmayacaktır. Bu dizinler için uygun araçların kullanılması gerekir. Bu başka bir yazının konusudur.

Aşağıdaki komut mtree uygulamasını her bir disk bölümünde bağlanma noktası düzeyinde çalıştıracaktır. mtree daha derine inip tarama yapmayacaktır. Dosya içeriğinde cksum, md5 ve sha256 bütünlük kontrolü algoritmaları ile elde edilen kontrol değerleri yer alacaktır. Sonuç ise /tmp dizininde mtree.txt dosyasına yazılacaktır.

```
tardis# mtree -x -ic -K cksum -K md5digest -K sha256digest -p / > /tmp/mtree_tarama.txt
```

Yukarıda gördüğünüz komut genel olarak birçok kullanıcı tarafından kullanılır. -x seçeneği bağlanma noktalarından daha aşağıya inilmeden tarama yapılmasını belirtir. -ic seçeneği tarama sonuçlarını ekrana yansıtır ve aynı zamanda alt dizinlerin çıktıda girintiler halinde gösterilemesini sağlar. Bu okuma kolaylığı sağlar. -K seçeneği ise kullandığımız anahtar sözcükleri tanımlar. Örneğin cksum, md5sum ve sha256sum gibi. -p seçeneği hangi disk bölümlerinin kontrol edileceğini belirtir.

Yukarıda belirtildiği gibi bazı dizinlerin içeriği sık sık değişiyorsa bu durumda bu dizinlerin kontrol edilmemesini sağlayabiliriz. Bunun için ise -X seçeneği ile taranmayacak dizinlerin belirtilmesini sağlayabilirsiniz. Tek bir dizin için -X ./dizin_adı olarak tanımlama yapılabilir ama birden çok dizin olması durumunda bunu bir dosya ile tanımlamak daha kolaydır. Aşağıdaki komut yukarıdaki örneğin bu sefer taranmayacak olan dizinlerin belirtildiği bir dosya kullanılılan biçimidir. Taranmayacak olan dizinlerin belirtildiği dosya mtree_taranmayacak.txt olarak belirtilmiştir. Çıktı da durumu belirtecek şekilde seçilmiştir.

```
tardis# mtree -x -ic -K cksum -K md5digest -K sha256digest -p / -X /home/goksin/mtree_taranmayacak.txt > /tmp/mtree_atla.txt
```

Taranmayacak olan dizinleri seçerken /tmp veya başka dizinleri belirtebilirsiniz. Bir dizinin taranmasını istemiyorsanız bunu belirtirken dosyaya aşağıdaki örnekte olduğu gibi ekleyebilirsiniz

```
[goksin@tardis /usr/home/goksin]$ cat mtree_taranmayacak.txt
./dev
```

```
./proc
./var
./tmp
```

Bir dizinin tanımlanması için ./ ile başlayarak dizin yolunu belirtmeniz daha uygun olacaktır. Tam dizin yolunu kullanırsanız bu durumda taramanın başlatıldığı dizin dikkate alınmadan taranan dizinin tam yolu kontrol edilecektir. Bu ayrım sistemin bir bölümünü taradığınız durumlarda dosyaların tanımlanmasında tam dizin yolu yerine dosya adının esas alındığı durumlarda önem kazanabilir. Tam dizin yolu tanımlanmadıysa mtree sadece dosya adını dikkate alacaktır. Örneğin /etc/hosts tam dizin yolu (absolute paths) iken hosts sadece dosya adı (basename) olarak tanımlanır.

mtree(1) Çıktılarının Yorumlanması

mtree, BSD sistemlerde sistemin kurulumu için kullanılmak üzere tasarlanmıştır. Mtree çıktısı bu nedenle spec olarak adlandırılır. BSD sistemimiz kurulu olduğuna göre yapacağımız kurulu olan sistemin kontrolü için spec dosyasını kullanmaktır. mtree çıktısı yani spec dosyası aşağıdakine benzeyecektir:

```
#        user: goksin
#     machine: tardis.elkotek
#        tree: /
#        date: Mon Apr 26 13:25:52 2010

# .
/set type=file uid=0 gid=0 mode=0755 nlink=1 flags=none
.                type=dir nlink=20 size=512 time=1272200336.000000000
    .cshrc           mode=0644 nlink=2 size=798 time=1258813896.000000000 \
                     cksum=3965810837 \
                     md5digest=353dc642ee0870d133b58eb9f9a0964b \
                     sha256digest=e1450d7b3c8b61f305b55792a653a547492b949d315ff7168205bdf279bd2f3e
    .profile         mode=0644 nlink=2 size=265 time=1258813896.000000000 \
                     cksum=890004139 \
                     md5digest=6587df187310db9f4fc108181bae2440 \
                     sha256digest=1b3b2de38739ad0949e948de0fe25c7f71a496f17e2be017711a11ec39ca5435
    COPYRIGHT        mode=0444 size=6206 time=1258813896.000000000 \
                     cksum=1940480824 \
                     md5digest=c2e7362d8d51695c44889c0748af3baf \
                     sha256digest=d7e4470fa8db194c0a474f9ddc061be240b7d96f0fe501bc5ef588b81e8e0c9c
    compat           type=link size=10 time=1270670671.000000000 \
                     link=usr/compat
    entropy          mode=0600 size=4096 time=1272200335.000000000 \
```

```
                    cksum=1336999312 \
                    md5digest=4badab1c79c4edea5364ed59029d6a5a \
                    sha256digest=b6338b9c13bc8543ca9a9a7150da0e23bdab2486fcb8bcea8f86f9110ad55689
    home            type=link size=8 time=1270671404.000000000 \
                    link=usr/home
    sys             type=link size=11 time=1258813908.000000000 \
                    link=usr/src/sys

# ./.snap
    .snap           type=dir gid=5 mode=0775 nlink=2 size=512 \
                    time=1270670408.000000000
    # ./.snap
    ..

........
```

İlk satır komutun kim tarafından, hangi sistemde çalıştırıldığını, dosya sistemi hakkındaki bilgiyi ve tarihi belirtir. İkinci girdi bilgisi ise /set ile başlayan satırdır ve mtree tarafından temel değer olarak kullanılır. Bu değerler kullanıcı tarafından oluşturulmaz, sisteme aittir. Dosya sistemindeki ilk nesne bir dosyadır ve sahibi ile grubunun numarası sıfır olarak tanımlıdır. İzinler 0755 olarak belirtilmiştir. Dosya sistemine ilişkin bir özel seçenek tanımlanmamıştır. (chflgs) root bölümüne ait olan bilgiler ise sonraki satırda yer almaktadır.

```
.               type=dir nlink=20 size=512 time=1272200336.000000000
```

(.) taramanın başlatıldığı dizindir, örneğimiz için bu root bölümüdür yani (.). Burada 24 tane hardlink bulunmaktadır. Dizinin boyutu 512 byte olarak verilmiştir. En son düzenlendiği zaman ilk UNIX sistemin ortaya çıkmasından bugüne dek geçen zaman olarak tanımlanmaktadır. Asıl adı ile EPOCH TIME. Bunu anlayacağımız hale dönüştürmek için date kullanabilirsiniz.

```
[goksin@tardis /etc/mtree]$ date -r 1272200336
25 Nis 2010 Paz EEST 15:58:56
[goksin@tardis /etc/mtree]$
```

root bölümündeki dosyalara ilişkin bilgiler ise bu satırın ardından gelmektedir. Aşağıdaki yukarıdaki çıktıdan alınan cshrc ve profile dosyalarına ilişkin bilgiler görülmektedir.

```
    .cshrc          mode=0644 nlink=2 size=798 time=1258813896.000000000 \
                    cksum=3965810837 \
                    md5digest=353dc642ee0870d133b58eb9f9a0964b \
                    sha256digest=e1450d7b3c8b61f305b55792a653a547492b949d315ff7168205bdf279bd2f3e
    .profile        mode=0644 nlink=2 size=265 time=1258813896.000000000 \
```

```
                    cksum=890004139 \
                    md5digest=6587df187310db9f4fc108181bae2440 \
                    sha256digest=1b3b2de38739ad0949e948de0fe25c7f71a496f17e2be017711a11ec39ca5435
```

Aynı şekilde profile ve cshrc dosyaları için bilgi verilmektedir. Bütünlük kontrolü için cksum-checksum, md5-md5sum ve sha256-sha256sum bilgileri ek olarak yer almaktadır. Bu değerler dosyanın içeriğine bağlı olarak hesaplanmaktadır. Bir saldırganının bu değerlere sahip olan ama içeriği farklı olan bir dosya oluşturması pratikte olanaksızdır. mtree taramasının çıktısını yukarıdaki örneklerde /tmp dizinine yönlendirmiştik. Bu dosyalardaki bilgiler bir yetkisiz giriş durumu şüphesinin kontrol edilmesini sağlamak için kullanılır. Bundan dolayı da söz konusu çıktıların sistemde saklanması yerine bir disket, CD/DVD veya flash disk üzerinde saklanması güvenlik açısından zorunludur. Bir yetkisiz erişim durumunda bu dosyaların içeriği değiştirilip saldırganın izlerini gizlemesi olanaklıdır. mtree ile yapacağını tarama değiştirilmiş olan tarama kaydı ile eşleneceği için sistem yöneticileri sistem güvende diyerek ve yetkisiz erişimi gerçekleştiren kişi ise izini kaybettirmenin huzuru ile mutlu olacaktır.

mtree Değerleri Eşleşmezse...

Sistemin çalışması sırasında bazı beklenmedik durumlar ile karşı karşıya kalınabilir. Bu durumda ilk akla gelen olası bir yetkisiz erişim durumu olabilir. Bunu kontrol etmenin en kolay yolu mtree ile yapılan kontrollere ilişkin spec dosyalarının karşılaştırılarak sorunun belirlenmesi olacaktır. Bunun için iki farklı mtree taramasına ait olan çıktılar karşılaştırılmalıdır. Karşılaştırma işlemi için diff kullanarak iki dosyayı karşılaştırabilirsiniz. Ayrıca buna ek olarak mtree kendi çıktılarını karşılaştırabilmektedir. Bu karşılaştırmanın sonucunu daha kolay inceleyebilmek için çıktısının bir dosyaya yönlendirilmesi gereklidir.

```
tardis# mtree -f dosya_1 -f dosya_2 > mtree_kontrol.txt
```

dosya_1 ve dosya_2 adından da anlaşılacağı üzere karşılatırma yapılacak olan dosyalardır. Çıktılarını ise mtree_kontrol.txt dosyasına yönlendiriyoruz. mtree ile yapılan karşılaştırmalarda değişmemiş olan dosyalar, dizinler adları ile gösterilir. Örneğin /usr/sbin/mtree satırı söz konusu dosya olan mtree dosyasının değişmediğini tanımlar. Ancak dosyalarda bir değişim söz konusu ise aşağıdakine benzer bir satır görebilirsiniz. Örneğin /sbin/devd değişmiş olsun.

```
                sbin/devd file cksum=3950957068 size=328396 md5digest=14d0cc1dbe
       a86c69ebd4af1dec2312d0 sha1digest=7acd1bdf46581b1d6a3231ca62b2c47e6e1dcf07
                sbin/devd file cksum=4141842183 size=328428 md5digest=5ab5ec4213
       03ca770ac2cccd546bcf11 sha1digest=51f5fbc24d47f11115474714040b8cce0eb9b6c9
```

Çıktıya bakıldığında bütünlük kontrolleri ile dosya boyutlarının eşleşmediğini görebiliriz. Bu durumda söz konusu dosya üzerinde değişikliğin kaynağını belirlemek gerekecektir. Sistem erişim yetkisi olan kişilerin bu değişiklik konusunda sorgulanması, sisteme erişim kayıtlarının kontrol edilmesi ve değişikliğin kaynağının belirlenmeye çalışılması yapmanız gereken ilk işlemlerdir. Eğer söz konusu değişiklik sisteme erişim

yetkisine sahip olan kişiler tarafından yapılmadıysa sisteminizi yeniden kurmak için gereken hazırlıkları yapmaya başlayabilirsiniz.

Sistem Güvenliği İçin Ek Önlemler

Sistemin dosya özniteliklerini değiştirmek, kullanıcıların erişim yetkilerini düzenlemek ve duyurulan her yamayı anında uygulamak sisteminizin güvenliğini sağlamak için yeterlidir. Birçok sistem yöneticisi gereken önlemleri almakla kendisinin güvende olduğu ve sistemlere yetkisiz erişim durumu ile karşılaşmayacağı düşüncesinin rahatlığına kapılabilir. Bu rahatlık duygusu tehlikelidir. Yayınlanan her yamanın uygulanmamış ve belirlenen her güvenlik açığının kapatılmamış olduğunu düşünerek, karşılaştığı her sistem üzerinde denemekten geri kalmayan ve hedef belirlediği sistemleri yakından izleyen ve açığını bulmak isteyen kişilerin işini kolaylaştıracağı gibi beklenmedik bir durum karşısında ne yapılacağı ve ne yapılmayacağını da belirleyecektir. Aşağıdaki öneriler sistem yöneticisi olmak isteyen veya sistem yöneticisi durumunda olan okuyuculara yol gösterici olacaktır.

* Sisteminizi iyi tanıyın. Sistemde sık sık ps -axx komutunu çalıştırıp normal koşullarda hangi uygulamaların çalıştığını kontrol edin. tanımadığınız bir uygulama çalışıyorsa bunu incelemeye alın.

* Sistemdeki açık olan portları netstat -na ve sockstat ile belirleyip, kontrol altına alın. Hangi TCP ve UDP portlarını kullandığınızı iyi bilin. Tanımadığınız bir portun açık/dinlemede olduğunu belirlerseniz bunun hangi uygulama veya servis tarafından kullanıldığını bulmaya ve bu port hakkında bilgi edinmeye çalışın. Bazen açık/dinlemede olan bir port bir yetkisiz erişimin kanıtı olabilir.

* Beklenmedik durumlar, sisteminizin sık sık yeniden başlaması, bir servisin farklı çalışması olası bir saldırıya işaret edebilir. Söz konusu sorunun kaynağının sistemden mi yoksa dış nedenlerden mi kaynakladığını belirlemeye çalışın. Bazen bu durumlar olası donanım sorunlarından da kaynaklanabilir. Sorunsuz sistemler herşeyin yolunda gittiği anlamına gelmez. Sorunsuz olarak çalışan sistemlerin de kontrol edilmesi gerekir. Teknik olarak bilgili ve uzman bir saldırgan sisteme giriş yaptığında geride kendi izini bırakmamaya özen gösterip tüm kanıtları yok edecektir.

* BSD sistemler düzenli olarak her gün sistem yöneticisine sisteme ilişkin bir raporu e-posta ile yollar. Posta kutunuzdaki bu raporu her gün okuyun ve bu raporları güvenli bir yerde saklayın. Bu raporlarda dikkat çekici bir şeyler varsa acilen araştırmaya başlayın. Önlem almanız gerekiyorsa hemen alın.

Port ağacından lsof(8) kurup çalıştırdığınız uygulamaların hangi dosyaları kullandığını izlemeye alın. Böylelikle çalıştırdığınız uygulamalar hakkında daha fazla bilgi edinirsiniz. Eğer tanımadığınız bir dosyaya rastladıysanız ya kullandığınız uygulamayı yeterince tanımıyorsunuz demektir ya da birileri bir şeyler yapıyor olabilir. Bu nedenle lsof(8)'u kendinizi eğitmek içinde kullanabilirsiniz. Bir diğer araç ise nessus'tur. Sisteminizin güvenlik açıklarını tarayarak raporlar. Ports ve pkgsrc ağacı üzerinde eski sürümleri bulunduğu için kendi sitesinden hazır paketleri indirip kullanabilirsiniz. Bunlardan başka birçok güvenlik aracı pkgsrc ve ports üzerinde bulunmaktadır. Bunları kullanarak da sisteminizin

kontrolünü yapabilirsiniz.

Yetkisiz Erişimlerin Belirlenmesi Durumunda...

Sisteme yetkisiz bir erişim olduğunu belirlerseniz sisteminizdeki dosyaların güvenli olduğunu ileri süremezsiniz. Saldırgan bir veya daha fazla dosyayı değiştirmiş ve sisteme ait bilgileri twitter, irc vb. başka birçok kanal üzerinden kendisine ulaşmasını sağlayacak bir düzenleme yapmış veya araçları sisteme yüklemiş olabilir. Bu durumda sisteme giriş yapılmasına neden olan açığının yamanması işlerin yolunda olduğu anlamına gelmeyecektir. Yapılması gereken tek şey sisteminizi güvenilir kaynaklardan yeniden kurmak, yamaları uygulamak ve gerekli güvenlik önlemlerini alıp yedeklerden dosyaları geri yüklemek olacaktır. Güvenlik konusunda titiz davranmak sisteminizi %100 güvenli kılmayacaktır ama sizi birçok sorundan uzak tutacaktır.

BSD - XVIII

Çekirdek - Kernel

Gökşin Akdeniz
goksin@enixma.org

Çekirdek/kernel nedir?

Okuduğunuz bir yazıda veya bir konuşma sırasında "kernel" veya "körnıl" terimini duymuşsunuzdur. Yeni çıkan Linux sürümünden söz ederken 2.6.a.b.c sürümünden söz edersiniz. UNIX ve türevi işletim sistemlerinin egemen olduğu özgür yazılım/açık kaynak kod dünyasında kernel erişilebilir durumdadır. İnternetten indirebilir, kaynak kodlarını elinizdeki CD/DVD içerisinde bulabilirsiniz. Mac OS X veya Windows için kernel pek bilinmeyen ve bir o kadar da erişilebilir olmayan bir bilinmezdir. Windows ailesinde kernel ortalıkta görülmez, adı geçmez ve sadece ileri düzey kullanıcılar tarafından bilinir. Windows, deyim yerinde ise kerneli gizlemek, gözlerden uzak tutmak ve bilinmemesi için her şeyin yapıldığı bir sitemdir. Ancak kernel yine de erişilebilir. Kendisi C:\windows\system atında yer alan çalıştırılabilir bir dosyadır. Mac Os X her ne kadar şeffaf olduğu ileri sürülse de üzerinde çekirdek yine gizlenmiştir. Çekirdek dizin içerine gizlenmiştir ve yine dizin içerisindeki bir gizli bir dizinde bulunur. Eski Apple sistemlerde ise dosya sisteminden bile gizlenmiş halde bulunur.

Birçok işletim sisteminde çekirdek gizlenmiş olsa da bilgisayarı ilk çalıştırdığınızda çalıştırılan program çekirdektir. Disk üzerindeki açılış sırasında okunan bloklar üzerinde yer alır. Çekirdek bilgisayarın ilk çalıştığı andan kapandığı ana dek çalışır. Çekirdeğin işlevi çalışmakta olan süreçlerin gereksinim duyduğu sistem kaynaklarını yönetmek ve paylaştırmaktır. Belleğe gereksinim duyan bir uygulamanın gereksinim duyduğu belleği ayırmak ve atamaktan, bilgisayarın diğer bilgisayarlar ile iletişim kurmasını sağlayan ağ protokollerine kadar geniş bir yelpazede bilgisayarınızın çalışması için gereken her işlemi, süreci ve sistem kaynaklarını kontrol eder ve yönetir. Bütün bunları gerçekleştirirken donanım ile diğer yazılımlar arasında yer alır, yani kısaca donanım ile yazılım arasında bir arayüz olarak da tanımlanabilir. Bu arayüz çalıştırılan tüm yazılımların gereksinim duyduğu sistem kaynaklarını paylaştırmak, bu programları ve sistem kaynaklarını yönetmek ile yükümlüdür. Bilgisayar bilimi açısından bu tanım tam olmasa da çekirdeğin işlevini açıklamakta yeterlidir.

Çekirdek Mimarisi

BSD sistemler diğer işletim sistemlerinden farklı olarak mikro kernel mimarisini kullanırlar. Linux sistemler halen monolitik kernel mimarisini kullanmakta olsalar da mikro kernel mimarisinin modüler yapısını da kullanmaktadır. Windows eski sürümleri NT, 2000 ve sonrası ile Mac OS X ve sonrası sürümleri de microkernel mimarisini kullanmaktadırlar. Mikro kernel mimarisi monolitik kernel mimarisine göre bazı üstünlüklere sahiptir. Sisteme eklenen veya

sistemden çıkarılan donanımlar, cihazlar için çekirdeğin yeniden derlenmesine gerek duyulmaz. Bunun yerine ilgili çekirdek modülü yüklenir veya çıkartılır. Monolitik çekirdek mimarisinde (Linux'un eski sürümlerinde durum budur) çekirdek kodu daha hızlı çalışması ve çok görevlilik için çok sayıdaki süreçler arasında işlemcinin bir süreçi işlerken bir diğer sürece geçip onu işlemeye kaldığı yerden devam etmesini sağlayan geçişlerin -context switching- en az sayıda olması için tasarlanmıştır. Bu tasarım çekirdeğin daha hızlı çalıştırılmasını sağlarken geliştirici tarafında ise geliştirme sürecini oldukça zorlaştırmaktadır. Ayrıca sistem yöneticilerinin sisteme ekledikleri her yeni donanım için çekirdek yapılandırmasını değiştirip yeniden çekirdek derlemelerini zorunlu kılmaktadır.

Bugün için mikro kernel ile monolitik kernel arasındaki fark giderek azalmaktadır. Monolitik kernel sistemler modüler yapıya dönüşmektedir. Mikro kernel ile monolitik kernel arasındaki fark temelde tasarıma yönelik yapılan seçimlerden kaynaklanmaktadır. Görünürdeki fark donanım ekleme/çıkarmadaki üstünlük gibi görünse de aslında bundan daha fazladır. İki mimari arasındaki en büyük fark mikro kernel mimarisinin ana sisteminin görevi olmayan bazı sistem çağrılarının kullanıcı düzeyinde işlenmesine olanak vermesidir. Böylelikle çekirdeğin asıl işletim sisteminin yapması gereken işlemleri yerine getirecek şekilde tasarlanması ve çekirdeğin bu temel sistem çağrılarına yanıt verecek şekilde tasarlanarak etkin bir şekilde çalışan küçük boyutlu bir çekirdeğe dönüşmesi sağlanır. Bu mimari sadece en üst düzeyde yetki gerektiren işlemlerin çekirdek tarafından kontrol edilip işlenmesini sağlar. Böylelikle sadece sistemin işlemesi için gerekli olan temel işlevlere sahip olan çekirdek daha basit bir yapıya sahip olurken, donanım sürücülerin kodu modüler bir yapıya kavuşur ve de çekirdek düzeyinde güvenlik açıklarının oluşma riski azalır ve doğru şekilde yazılan kod ile bu açıklar sıfır denecek kadar azalır. Bkz: OpenBSD. Uzun lafın kısası mikro çekirdek mimarisi nesne yönelimli programlama ile birçok benzerliğe sahiptir: prosedürel veya monolitik bir program kadar hızlı çalışmayabilir ama temel bileşenleri kullanarak kolaylıkla yeni gereksinmelere uyum sağlayabilir ve geliştirilmesi daha kolaydır. Ayrıca sistem çalışırken çekirdek modülleri yüklenebilir veya çıkarılabilir. Modüler yapının sağladığı olanaklar çekici olsa da işletim sistemi tasarımı açısından bakıldığında işletim sisteminin bileşenleri arasında gerçekleşen iletişimin izlenmesi ve ortaya çıkan hataların giderilmesi de o derece zor olmaktadır.

BSD Sistemlerde Çekirdek

BSD sistemlerde çekirdek monolitik veya modüler olarak tasarlanır. NetBSD ve OpenBSD monolitik çekirdek kullanırken FreeBSD mikro kernel mimarisini ve çekirdek modüllerini kullanır. Bu farklılıklardan dolayı çekirdek konusu gelecek bölümde de yer alacak. Bu bölümde FreeBSD çekirdek yapılandırması ve kontrolünü esas alırken gelecek sayıda NetBSD ve OpenBSD sistemlerde çekirdek konusunu işleyeceğiz.

FreeBSD Çekirdeği

FreeBSD'de çekirdek mikro kernel mimarisi ile modüler olarak tasarlanmıştır. Çekirdek ve modüller /boot/kernel adlı dizinde yer alır. Bu dizin altında yer alan tüm dosyalar FreeBSD sisteminizin çekirdeğini oluşturur. Sistem ve kullanıcı tarafı olarak yukarıda kısaca değindiğimiz ayrım burada daha kolay görülebilir. İşletim sistemi tarafından yapılması gereken tüm işlevleri gerçekleştiren çekirdek ve ilgili modüller yani "çekirdek tarafı" yer alan bileşenler bu dizindedir. Bunların dışında kalan ve "kullanıcı tarafı" olarak adlandırılan ve çekirdek işlevlerini kullanmayı gerektiren ve bu dizin dışında kalan dosyalar – araçlar kullanıcı tarafı olarak adlandırılır.

Çekirdeğin yönetimi iki şekilde gerçekleştirilebilir. Birincisi çekirdeğe ait bazı parametrelere atanan değerler ile çekirdeğin işleyişinin doğrudan kontrol edilmesi ve ikincisi de çekirdek modüllerinin yüklenmesi

ve kaldırılmasıdır. Çekirdek parametrelerine atama sysctl(8) ile gerçekleştirilir.

sysctl(8) programı çekirdek parametrelerine değer atanması için kullanılır. Bu değerler çekirdek parametreleri -kernel variables- olarak adlandırılsa da bazen "sysctls" olarak da adlandırılır. Çekirdek parametrelerinin doğrudan kontrol edilebilmesi çekirdeğin işleyişi üzerinde özellikle de performans açısından tam bir kontrol kurulmasını sağlar. Böylelikle çekirdeğin özel durumlar/donanımlar ve sistemler için yeniden derlenmesine gerek kalmadan dağıtım ile gelen hali ile kullanılmasını sağlanır. Bu kontrol sadece çekirdek performansı üzerinde değil aynı zamanda başka programlar üzerinde de ek bir yapılandırmaya gidilmeden dolaylı olarak ince ayar yapılmasını da sağlar. Çekirdek üzerindeki bu doğrudan kontrol bazı durumlarda hatalı yapılandırmalara neden olabilir.

sysctl ile kontrol edilebilen ve edilmeyen parametrelerin listesini döndürmek için root olarak sysctl kullanılması gerekir. Aşağıda sysctl -A çıktısının bir bölümü görülüyor

```
tardis # sysctl -A
kern.ostype: FreeBSD
kern.osrelease: 8.0-RELEASE-p2
kern.osrevision: 199506
kern.version: FreeBSD 8.0-RELEASE-p2 #0: Tue Jan  5 21:11:58
UTC 2010
    root@amd64-
builder.daemonology.net:/usr/obj/usr/src/sys/GENERIC
kern.maxvnodes: 100000
kern.maxproc: 6164
kern.maxfiles: 12328
kern.argmax: 262144
kern.securelevel: -1
kern.hostname: tardis.elkotek
kern.hostid: 1925341659
kern.clockrate: { hz = 1000, tick = 1000, profhz = 2000,
stathz = 133 }
```

```
kern.proc.all: Format:S,proc Length:184960
Dump:0x40040000000000000a0875e3600ffffff...
kern.proc.proc: Format:N Length:121856
Dump:0x40040000000000000a0875e3600ffffff...
kern.proc.kstack: Format:N Length:1096
Dump:0xc0860100000000002330203078666666...
kern.proc.proc_td: Format:N Length:184960
Dump:0x40040000000000000a0875e3600ffffff...
kern.file: Format:S,xfile Length:324000
Dump:0x50000000000000000cf05000000000000...
kern.posix1version: 200112
kern.ngroups: 1023
kern.job_control: 1
.....
```

Yukarıdaki çıktıda sadece çok küçük bir kısmını görebilirsiniz. Tümünü görmek için çıktının bir dosyaya yönlendirilmesi gerekecektir. Çıktıyı incelediğinizde parametrelere ait değerlerin bazılarının anlaşılabilir olduğu gibi bazılarının da anlaşılması güç sayı ve harf birleşimleri olduğunu görebilirsiniz. Örneğin

```
kern.ostype: FreeBSD
```

sistemin adını belirtmektedir. Bu kolaylıkla anlaşılabilir, ancak aşağıdakinin ne olduğu ilk bakışta anlaşılmayabilir.

```
kern.ipc.msqids: Format: Length:4480
Dump:0x8cfe330356291872377df2e7fb6571...
```

Bunun ne anlama geldiğini bilmeyebilirsiniz. Kısa bir araştırma sonucunda ne anlama geldiğini bulabilirsiniz. Bunun gibi bazı satırlar donanım ile ilgili bilgileri barındırıyor olabilir. Bu durumda bu konuda donanım üreticisinin kaynaklarına başvurmak yerinde olacaktır.

sysctl(8) ile kontrol edilebilen veya okunabilen parametrelerler Management Information Base – MIB – adı verilen bir isimlendirme ile tanımlanırlar. Bu isimlendirme sistematiği çekirdeğin ilgili özelliği veya fonksiyonunu tanımlayan bir kısaltma ve bu fonksiyon ile ilişkili olan

diğer fonksiyonlar şeklinde belirtilir. Aşağıda sistem ile gelen çekirdek yapılandırmasında sysctl(8) ile görebileceğiniz bazı çekirdek işlevleri gösterilmektedir.

```
kern        Temel çekirdek işlevleri ve özellikleri
vm          Sanal bellek sistemi
vfs         Dosya sistemi
net         Ağ
debug       Hata ayıklama
hw          Donanım
user        kullanıcı tarafına ait bilgi
compat      Uyumluluk kipi bilgisi (Linux emülasyonu vs)
security    Güvenlik özellikleri
dev         Donanım sürücülerine ilişkin bilgi
```

Bu özellikler/işlevler yine ardından gelen ek işlev ve özellikler ile kendi içerisinde yeniden ayrışır. Bu ayrışma kendi içerisinde sadece ve sadece belirli bir tek işlevi kontrol eden özel ve tek parametre olarak sonlanır. Aşağıdaki örnekte bunu görebilirsiniz.

```
tardis# sysctl -a | grep "security"
security.jail.param.cpuset.id: 0
security.jail.param.host.hostid: 0
security.jail.param.host.hostuuid: 64
security.jail.param.host.domainname: 256
security.jail.param.host.hostname: 256
security.jail.param.children.max: 0
security.jail.param.children.cur: 0
security.jail.param.enforce_statfs: 0
security.jail.param.securelevel: 0
security.jail.param.path: 1024
security.jail.param.name: 256
security.jail.param.parent: 0
security.jail.param.jid: 0
security.jail.enforce_statfs: 2
security.jail.mount_allowed: 0
security.jail.chflags_allowed: 0
security.jail.allow_raw_sockets: 0
security.jail.sysvipc_allowed: 0
security.jail.socket_unixiproute_only: 1
security.jail.set_hostname_allowed: 1
security.jail.jail_max_af_ips: 255
security.jail.jailed: 0
security.bsd.map_at_zero: 0
security.bsd.suser_enabled: 1
security.bsd.unprivileged_proc_debug: 1
security.bsd.conservative_signals: 1
security.bsd.see_other_gids: 1
security.bsd.see_other_uids: 1
security.bsd.unprivileged_read_msgbuf: 1
security.bsd.hardlink_check_gid: 0
security.bsd.hardlink_check_uid: 0
security.bsd.unprivileged_get_quota: 0
security.mac.labeled: 0
security.mac.max_slots: 4
security.mac.version: 4
security.mac.mmap_revocation_via_cow: 0
security.mac.mmap_revocation: 1
tardis#
```

sysctl(8) ile döndürülen değerler çekirdek tarafından kullanılan bir özellik, sistem yapılandırması veya çekirdeğin işleyişi için önemli olan bir özelliğe ait değerlerdir. Bu parametreler üzerinde yapılan değişiklikler doğrudan çekirdeğin işleyişini etkiler. Örneğin bir sürecin açabileceği dosya sayısı, bir ağ bağlantısının kullanabileceği bellek miktarı veya ağdan gelen paketlerin nasıl işleneceği gibi değerler sistemin doğrudan işleyişini etkiler. Sistemin bulunduğu ağda ortaya çıkan problemlerin sistemi etkilememesi için bu değerler uygun şekilde ayarlanarak sorun çözülebilir.

sysctl(8) ile kontrol edilen parametrelere atanan değerler tamsayı veya dizge olabilir. Bunların dışında sysctl(8) kılavuz sayfasında belirtildiği ve sysctl(8) çıktılarından da görebileceğiniz gibi bazı parametrelerin değerleri ise sadece ilgili donanım veya yazılım tarafından anlaşılabilecek şekilde döndürülür. sysctl(8) ile kontrol edilebilen parametrelerin tamamını tanımlayan tek bir kaynak, belge bulunmamaktadır. Bu nedenle ilgili parametre hakkında bilgi ilişkili

olduğu kılavuz sayfasından veya bazı durumlarda ise kaynak kod içerisindeki yorum satırlarından çıkabilir. Örneğin kern.securelevel ile ilgili bilgi init(8) kılavuz sayfasında açıklanmaktadır. Belirli bir çekirdek parametresine ilişkin bilgi edinmek için "-d" parametresi ile sysctl(8) kullanılabilir.

```
tardis# sysctl -d kern.sched.preemption
kern.sched.preemption: Kernel preemption enabled
tardis# sysctl kern.sched.preemption
kern.sched.preemption: 1
tardis#
```

sysctl(8) ile belirli bir özellik ile ilgili olan tüm alt işlevleri ve özellikleri de döndürebilirsiniz.

```
tardis# sysctl user
user.cs_path: /usr/bin:/bin:/usr/sbin:/sbin:
user.bc_base_max: 99
user.bc_dim_max: 2048
user.bc_scale_max: 99
user.bc_string_max: 1000
user.coll_weights_max: 0
user.expr_nest_max: 32
user.line_max: 2048
user.re_dup_max: 255
user.posix2_version: 199212
user.posix2_c_bind: 0
user.posix2_c_dev: 0
user.posix2_char_term: 0
user.posix2_fort_dev: 0
user.posix2_fort_run: 0
user.posix2_localedef: 0
user.posix2_sw_dev: 0
user.posix2_upe: 0
user.stream_max: 20
user.tzname_max: 255
tardis#
```

Belirli bir özelliğe ilişkin tüm parametreler bu şekilde döndürülebilir. Zaman içerisinde MIB değerlerine hakim oldukça doğrudan ilgili değeri çağırabilirsiniz. Örneğin kasadaki sıcaklık:

```
tardis# sysctl hw.acpi.thermal.tz0.temperature
hw.acpi.thermal.tz0.temperature: 11,0C
tardis#
```

sysctls ile Parametre Düzenlemek

sysctl(8) kılavuz sayfasında okuduğunuz üzere bazı parametrelerin değerleri değiştirilemez. Diğer bir deyişle bunlar salt okunurdur. Örneğin işlemciniz, işlemci mimarisi ve işlemci sayısı gibi değerler salt okunur değerlerdir ve bunlar değiştirilemez. Ancak donanımı değiştirmeniz durumunda değişirler.

```
tardis# sysctl hw.machine
hw.machine: amd64
tardis# sysctl hw.model
hw.model: AMD Athlon(tm) Dual Core Processor 5000B
tardis# sysctl hw.ncpu
hw.ncpu: 2
tardis#
```

Öte yandan normal kullanıcıların optik sürücüler ile disketler vb depolama araçlarını bağlama/ayırmalarına izin veren vfs.usermount parametresi doğrudan değiştirilebilir. Varsayılan değeri sıfırdır ve kullanıcıların adı geçen depolama birimleri üzerinde bağlama/ayırma işlemleri yapmasına izin vermez. Bu değere atanan 1 değeri ile kullanıcılar adı geçen işlemleri yapabilir. Değişiklik yapılabilen çekirdek parametreleri çekirdek üzerinde herhangi bir işlem yapmadan çekirdeğin işleyişini değiştirir. vfs.usermount parametresine 1 değerini atamak için sysctl'dan yararlanılır.

```
tardis# sysctl vfs.usermount=1
vfs.usermount: 0 -> 1
tardis#
```

sysctl(8) işlemi gerçekleştirdiğinde önce eski değeri ve ardından atanan yeni değeri döndürür. Bu şekilde sistem çalışırken değiştirilebilen parametrelere çalışma zamanı parametreleri – run-time variables adı verilir.

sysctl(8) ile sistemin çalışması sırasında değiştirilebilen parametrelerin sistem çalışır durumda olduğu ve/veya değiştirilmediği sürece değeri korunur. Bu değerlerin kalıcı olması için /etc/sysctl.conf dosyasında tanımlanması gerekir. Bu dosyayı tercih ettiğiniz bir metin düzenleyicisi ile ya da doğrudan komut satırından sözkonusu MIB ile değerinin yazılması değerin kalıcı olmasını sağlar.

```
tardis# echo "vfs.usermount = 1" > /etc/sysctl.conf
tardis#
```

Çalışma zamanı parametreleri gibi bazı MIB sistemin ilk çalıştırıldığı anda tanımlanması ile değişebilir. Bunların alacağı değerler /boot/loader.conf dosyasında aynı sysctl.conf(5) dosyasında olduğu gibi tanımlanarak kullanabilir. Bu tanımlanabilecek olan parametreler /boot/defaults/loader.conf dosyasında tanımlıdır. Bu dosyadan yararlanarak sistemin açılışı sırasında tanımlanabilecek olan parametrelerin değerlerini belirtebilirsiniz. Bu dosya tüm parametreleri tanımlamaz. Bunlara ek olarak ayrıca tanımlanabilecek olan parametreler device.hints(5) doyasında yer alır. Bu dosya /boot dizininde yer alır. Bu dosyadaki değerler donanım ile ilişkili çekirdek parametelerini belirtir.

```
# $FreeBSD: src/sys/amd64/conf/GENERIC.hints,v 1.21.2.1.2.1
2009/10/25 01:10:29 kensmith Exp $
hint.fdc.0.at="isa"
hint.fdc.0.port="0x3F0"
hint.fdc.0.irq="6"
hint.fdc.0.drq="2"
hint.fd.0.at="fdc0"
hint.fd.0.drive="0"
hint.fd.1.at="fdc0"
hint.fd.1.drive="1"
hint.atkbdc.0.at="isa"
hint.atkbdc.0.port="0x060"
hint.atkbd.0.at="atkbdc"
hint.atkbd.0.irq="1"
hint.psm.0.at="atkbdc"
hint.psm.0.irq="12"
hint.sc.0.at="isa"
hint.sc.0.flags="0x100"
hint.uart.0.at="isa"
hint.uart.0.port="0x3F8"
hint.uart.0.flags="0x10"
hint.uart.0.irq="4"
hint.uart.1.at="isa"
hint.uart.1.port="0x2F8"
hint.uart.1.irq="3"
hint.ppc.0.at="isa"
hint.ppc.0.irq="7"
hint.atrtc.0.at="isa"
hint.atrtc.0.port="0x70"
hint.atrtc.0.irq="8"
```

loader.conf(5) ve device.hints(5) dosyalarında tanımlanan çekirdek parametreleri düşük seviyeli çekirdek işlemleridir. Bunlar genel olarak donanım yapılandırması vb işlevlerdir. Öte yandan sistem çalışırken değiştirebildiğiniz parametreler ise yüksek seviyeli çekirdek işlevleridir. Bu parametrelere atayacağınız değerler root yetkileri ile yapılabildiği için sistem yöneticisinin sorumluluğunda olan işlemlerdir. Yapılan düzenlemelerin sisteminizin performansı üzerinde etkili olacağını gözardı etmemek gereklidir. Parametrelere atanan değerler, atandığı şekilde değiştirilebilir.

FreeBSD Çekirdek Modülleri

Çekirdeğin modüler yapısı farklı çekirdek modüllerinin sistem çalışırken yüklenebilmesini veya çıkarılmasını sağlar. Böylelikle bir çekirdek modülüne gereksinim duyduğunuzda bunu çağırıp yükleyebilir ve işiniz bittikten sonra gereksinim duymuyorsanız ilgili modülü

kaldırabilirsiniz. Böylelikle kablosuz ağ kartları vb. donanımların kullanırken gereken modüller donanımın kaldırılmasının ardından kaldırılabilir.

Burada loader.conf(5) veya device.hints(5) arasında fark yok gibi görülebilir. Aslında iki dosyanın farklı olan yanı device.hints(5) dosyasının loader(8) tarafından açılış sırasında okunarak çekirdeğe dosyada yer alan donanımların yapılandırmasını bildirmesidir. loader.conf(5) dosyası da bazı çekirdek parametrelerini tanımlamak için kullanılabilir ama loader(8) yapılandırması asıl görevidir. Bu nedenle donanıma ilişkin yapılandırmanın device.hints(5) dosyasında olması daha uygun olmaktadır.

BSD sistemlerde çekirdek dosyası /boot/kernel/kernel ve çekirdek modül dosyaları da /boot/kernel dizini altında yer alır. Bu dizine göz atacak olursanız birçok çekirdek modülü dosyasını görebilirsiniz. Çekirdek modül dosyalarının uzantısı ko'dur. Çekirdek modülü isimleri de ait oldukları modülün adını taşır. Bu isimlendirme ile çekirdek ve modülleri kolaylıkla ayırt edilebilir.

Sisteminizde çalışmakta olduğunuz sırada yüklenen modüllerin bir listesini döndürmek için kldstat(8) kullanılır.

```
tardis# kldstat
Id Refs Address            Size     Name
 1   21 0xffffffff80100000 d188c0   kernel
 2    1 0xffffffff80e19000 23ed0    snd_hda.ko
 3    2 0xffffffff80e3d000 75708    sound.ko
 4    1 0xffffffff80eb3000 52d0     atapicam.ko
 6    1 0xffffffff81022000 a3c      pflog.ko
 7    1 0xffffffff81023000 2bbc1    pf.ko
 8    1 0xffffffff8104f000 5ad4a    radeon.ko
 9    1 0xffffffff810aa000 11454    drm.ko
tardis#
```

Sisteme yüklü olan çekirdek modülleri ses sistemi ve ses kartı sürücüsü, CD/DVD yazıcılar için gerekli olan atapicam, grafik kartı modülü ile pf ve pflog modülleridir. Sisteminizdeki donanıma bağlı olarak kldstat(8)'ın döndüreceği çekirdek modülleri farklılık gösterebilir. Örneğin ses sistemi modülünün yanında kullandığınız ses kartının sürücüsü farklı olacaktır. Çekirdek, ilgili altsistem-subsytem ile bu altsisteme bağlı olan modüller sisteminizde yüklü bulunacaktır. kldstat(8) komutunu ilgili seçenekler ile çalıştırırsanız yukarıdaki çıktının daha ayrıntılı bir dökümünü elde edebilirsiniz. Bu dökümde çekirdek içerisinde yer alan modüllerin tam bir dökümü ile yüklü olan diğer modüllerin ayrıntılı bir listesini de elde edebilirsiniz.

Modüllerin Yüklenmesi ve Kaldırılması

freeBSD'de bir çekirdek modülünü yüklemek için kldload(8) ve kaldırmak için de kldunload(8) kullanılır. Bir modülü yüklemek veya kaldırmak için modülün bulunduğu dizini de belirten tam yolunu yazmanıza gerek yoktur. Modülün adını yazmanız yeterlidir.

```
# kldload modül_adı.ko
```

benzer biçimde kaldırmak içinde

```
# kldunload modül_adı.ko
```

Eğer modülün tam adını biliyorsanız modül adını yazarak kullanmak en basit yoldur. Aksi durumda ise modüle ait tam yolu yazarak kabuğun komut tamamlama özelliğini kullanabilirsiniz. Aşağıda virtualbox için gereken vboxdrv.ko modülü yükleniyor

```
tardis# kldload vboxdrv.ko
tardis# kldstat
Id Refs Address            Size     Name
 1   23 0xffffffff80100000 d188c0   kernel
 2    1 0xffffffff80e19000 23ed0    snd_hda.ko
 3    2 0xffffffff80e3d000 75708    sound.ko
 4    1 0xffffffff80eb3000 52d0     atapicam.ko
```

```
 6    1 0xffffffff81022000 a3c      pflog.ko
 7    1 0xffffffff81023000 2bbc1    pf.ko
 8    1 0xffffffff8104f000 5ad4a    radeon.ko
 9    1 0xffffffff810aa000 11454    drm.ko
10    1 0xffffffff810bc000 23754    vboxdrv.ko
tardis#
```

Modülü kaldırdıktan sonra

```
tardis# kldunload vboxdrv.ko
tardis# kldstat
Id Refs Address            Size     Name
 1   21 0xffffffff80100000 d188c0   kernel
 2    1 0xffffffff80e19000 23ed0    snd_hda.ko
 3    2 0xffffffff80e3d000 75708    sound.ko
 4    1 0xffffffff80eb3000 52d0     atapicam.ko
 6    1 0xffffffff81022000 a3c      pflog.ko
 7    1 0xffffffff81023000 2bbc1    pf.ko
 8    1 0xffffffff8104f000 5ad4a    radeon.ko
 9    1 0xffffffff810aa000 11454    drm.ko
tardis#
```

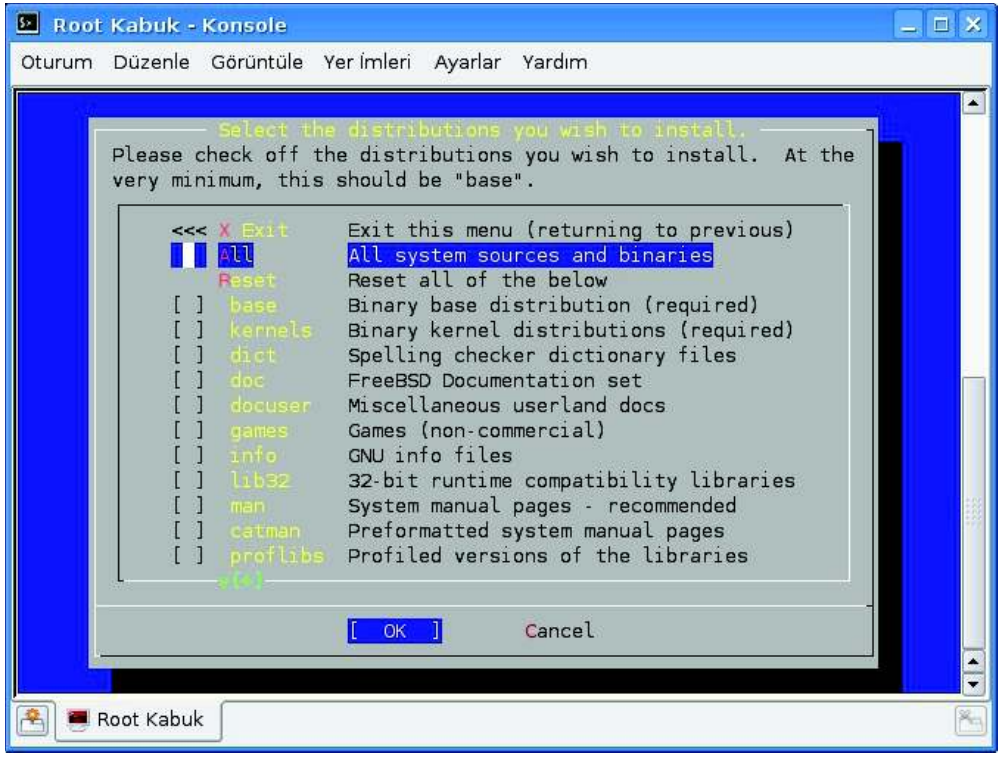
Çekirdek Modüllerinin Otomatik Yüklenmesi

Bir veya daha fazla çekirdek modülünün yüklenmesi veya kaldırılması işlemlerini sık sık yapmak durumunda kalıyorsanız gereksinim duyduğunuz çekirdek modüllerinin sistemin başlatılması durumunda loader(5) tarafından yüklenmesini sağlayabilirsiniz. Bunun için yapılması gereken, yüklenmesini istediğiniz modülün adını aşağıdaki gibi loader.conf(5) dosyasına eklemenizdir.

```
tardis# cat loader.conf
snd_hda_load="YES"
atapicam_load="YES"
cd9660_incov_load="YES"
vboxdrv_load="YES"
tardis#
```

Yüklenmesi gereken modülün ".ko" uzantısını kaldırıp sonuna "_load="YES"" eklemeniz durumunda söz konusu modül açılışta yüklenecektir.

Bu yapılandırmada önemli olan nokta, yüklenmesini istediğiniz çekirdek modülüne karar vermektir. Genellikle tercih edilenler GENERIC çekirdek tarafından desteklenmeyen donanımlara ait olan sürücülerdir. Böylelikle çekirdeğin yeniden derlenmesine gerek kalmadan söz konusu donanım için desteği ilgili sürücü modülünü açılışta yükleyerek sağlayabilirsiniz. Bir modülü yüklemeden önce kılavuz sayfasından ayrıntılı bilgi alabilirsiniz. Evet FreeBSD'de kullandığınız araçlar dışında, yapılandırma dosyaları, çekirdek ve modülleri içinde bir kılavuz sayfası bulunur.



Resim 1 - FreeBSD kurulumunda tüm sistem kaynakları ve ikili dosyaların seçimi

Çekirdek Derlemek

Çekirdek parametrelerinin bir bölümü düzenlenebilir. İstenilen çekirdek modülleri de yüklenebilip kaldırılabiliyorken çekirdeğin yeniden derlenmesine neden gerek duyulur? Bu sorunun yanıtı GENERIC çekirdeğin desteklemediği donanım veya FreeBSD sisteminizi kuracağınız platforma özel yapılandırma gereksinimlerinde yatmaktadır. GENERIC adı verilen çekirdek FreeBSD CD/DVD ile gelir ve geniş bir donanım yelpazesini destekleyecek şekilde derlenmiştir. Bu yapılandırma üzerinde bulunduğunuz donanımın gerektirdiği bazı modüller çekirdek içerisinde yer almayabilir veya modül olarak bırakılmıştır. GENERIC çekirdek içerisinde yer alan bileşenlerin birçoğu da sizin için gerekli değildir. Bu durumda çekirdeği donanımınıza veya platforma uygun olarak yeniden yapılandırıp derlemek uygun olacaktır. Burada unutmamanız gereken önemli nokta çekirdek derlemenin her sorununuzun çözümü olmadığıdır. Donanımdan kaynaklı bir sorun çekirdek derlenerek çözülmez! Desteklenmeyecek bir donanım için çekirdeği yeniden derlemek de sorunu çözmez! FreeBSD, NetBSD, OpenBSD ve diğerlerinin geliştiricileri donanım desteği konusunda donanım üreticilerinin tavrına ve kendi kaynaklarını da dikkate alarak karar verirler ve geliştirme süreci de bu kararlara göre sürer. Bu nedenle Geliştiricilerin kesin olarak belirttikleri destekleyecekleri veya desteklemeyecekleri donanımları çalıştırmak için çekirdeğin yeniden derlenmesinin bir yarar sağlamayacağı açıktır. Bu konu aslında BSD geliştirme süreci ile ilgili olduğu için bunu ayrı bir yazıda ele almak gerekecektir.

Gereksinimler

FreeBSD çekirdeğinizi derlemek istiyorsanız sistem kaynak kodlarının kurulu olması gereklidir. Sistem kaynak kodlarını kurulum sırasında kurmadıysanız sysinstall ile kurabilirsiniz. Sysinstall'u çalıştırıp configure seçeneğinde Distributions'a gelerek kaynak kodlar kurulmadıysa kurabilirsiniz. Eğer freeBSD sisteminizi freebsd-update ile güncellediyseniz, elinizdeki CD/DVD seti ile sisteminiz uyumlu değildir. Sysinstall CD/DVD setindeki sistem ile kurulu sistem arasında -örneğin bendeki 8.0-RELEASE seti ile kurulu olan 8.0-RELEASE-p2 uyumlu değildir- uyumsuzluğa işaret edip işlemi yapmayacaktır. Bunu aşmak için CD/DVD bağlayıp içerisindeki kaynak kodların bulunduğu dizine geçerek bu dizinde yer alan kabuk programını root olarak çalıştırmanız gerekecektir. Program kaynak kodları ilgili dizine kop-yalayacaktır. Elinizdeki kaynak kodlar halen sisteminiz ile uyumlu değildir. Uyumlu hale getirmek için yeniden freebsd-update'i root olarak çalıştırın. Bu işlem ile kaynak kodlar son yamaları da içeren güncel kararlı sürüme terfi edecektir. Bu işlemi cvs kullanarak da yapabilirsiniz. Ancak elinizde hazır CD/DVD seti varsa freebsd-update ile bunları güncellemek hem bant genişliği hem de zaman ka-zandıracaktır.

Çekirdek kaynak kodu /usr/src/sys dizini altında bulunmaktadır. /sys dizini aslında bu dizine işaret eden bir sembolik bağdır. Doğrudan sys dizinine geçtiğinizde /usr/src/sys dizinine geçmiş olursunuz. Bu dizindeki kaynak kodlar ile çalışmaya başlamadan önce platformunuzu, donanımızı kesin ve tam olarak bilmeniz gereklidir. Bu özellikle de donanım üreticilerinin belirli bir ürün ailesi için aynı adı kullanmalarından kaynaklanan belirsizliklerin neden olacağı hataları önlemek için zorunludur. Örneğin D-Link ağ kartınızın aslında yonga setinin Broadcom olmasından dolayı, birçok kaynakta D-Link donanımlarının BSD, GNU/Linux uyumlu olduğu ileri sürülse de, elinizdekinin neden çalışmadığını açıklayacaktır. Benzeri birçok yanıltıcı markalar yanında bazı donanımlarınızın kutularında belirtildiği mimari ve yonga seti ile geldiğinden emin olabilirsiniz. Örneğin Atheros ağ kartlarının gerçekten Atheros yongasına sahip oldukları gibi FreeBSD tarafından Intel olarak tanınan donanımın gerçekten Intel olduğudan emin olabilirsiniz.

Donanım hakkında kesin bilgi edinmek için birçok yöntem söz konusudur. Sistemin başlatılması sırasında bulunan donanımlara

ilişkin bilgi /var/run/dmesg.boot dosyasında yer alır. Bu dosyadaki herbir satır ya donanıma ya da ilgili çekirdek bileşenine ait bilgilerdir. Bu dosyayı eğer yeni bir çekirdek derleyecekseniz yedekleyip bir kopyasını referans olarak kullanabilirsiniz.

dmesg.boot dosyası sisteme ilişkin kesin bir bilgi verir. Bu bilgiyi doğru yorumlamak için dmesg dosyasındaki mesajların birbiri ile olan ilişkilerinin doğru anlaşılması gerekir.

```
Copyright (c) 1992-2009 The FreeBSD Project.
Copyright (c) 1979, 1980, 1983, 1986, 1988, 1989, 1991, 1992,
1993, 1994
      The Regents of the University of California. All rights
reserved.
FreeBSD is a registered trademark of The FreeBSD Foundation.
FreeBSD 8.0-RELEASE-p2 #0: Tue Jan  5 21:11:58 UTC 2010
    root@amd64-
builder.daemonology.net:/usr/obj/usr/src/sys/GENERIC
Timecounter "i8254" frequency 1193182 Hz quality 0
CPU: AMD Athlon(tm) Dual Core Processor 5000B (2611.90-MHz K8-
class CPU)
  Origin = "AuthenticAMD"  Id = 0x60fb2  Stepping = 2

Features=0x178bfbff<FPU,VME,DE,PSE,TSC,MSR,PAE,MCE,CX8,APIC,SE
P,MTRR,PGE,MCA,CMOV,PAT,PSE36,CLFLUSH,MMX,FXSR,SSE,SSE2,HTT>
  Features2=0x2001<SSE3,CX16>
  AMD
Features=0xea500800<SYSCALL,NX,MMX+,FFXSR,RDTSCP,LM,3DNow!+,3D
Now!>
  AMD Features2=0x11f<LAHF,CMP,SVM,ExtAPIC,CR8,Prefetch>
  TSC: P-state invariant
real memory  = 4294967296 (4096 MB)
avail memory = 4097519616 (3907 MB)
ACPI APIC Table: <HPQOEM SLIC-BPC>
FreeBSD/SMP: Multiprocessor System Detected: 2 CPUs
FreeBSD/SMP: 1 package(s) x 2 core(s)
 cpu0 (BSP): APIC ID:  0
 cpu1 (AP): APIC ID:  1
ioapic0: Changing APIC ID to 4
ioapic0 <Version 1.1> irqs 0-23 on motherboard
kbd1 at kbdmux0
acpi0: <HPQOEM SLIC-BPC> on motherboard
acpi0: [ITHREAD]
acpi0: Power Button (fixed)
```

Yukarıdaki mesajlara baktığımızda işlemci ile ilgili bilgileri okuyoruz. İşlemcinin özellikleri ve ardından sistem belleği vb bilgiler geliyor. Bunların ardından ise sistemdeki ilk donanım ACPI geliyor. Sonra klavye ve anakart görülüyor. dmesg.boot dosyasındaki kayıtları ayrıntılı bir şekilde incelediğinizde donanım ve ilişkili çekirdek bileşenleri hakkında ayrıntılı bilgi edinebilirsiniz. Bu bilgiler çekirdeğinizi yapılandırırken gerekecek olan tüm modülleri vb belirlemenize olanak verecektir. Eğer edindiğiniz bilgileri doğrulamak isterseniz pciconf(8) kullanabilirsiniz. pciconf(8) kullanırken -lv seçenekleri ile kullanmanızda yarar vardır.

```
tardis# pciconf -l
none0@pci0:0:0:0:       class=0x050000 card=0x2a72103c
chip=0x03ea10de rev=0xa1 hdr=0x00
isab0@pci0:0:1:0:       class=0x060100 card=0x2a72103c
chip=0x03e010de rev=0xa2 hdr=0x00
none1@pci0:0:1:1:       class=0x0c0500 card=0x2a72103c
chip=0x03eb10de rev=0xa2 hdr=0x00
none2@pci0:0:1:2:       class=0x050000 card=0x2a72103c
chip=0x03f510de rev=0xa2 hdr=0x00
ohci0@pci0:0:2:0:       class=0x0c0310 card=0x2a72103c
chip=0x03f110de rev=0xa3 hdr=0x00
ehci0@pci0:0:2:1:       class=0x0c0320 card=0x2a72103c
chip=0x03f210de rev=0xa3 hdr=0x00
pcib1@pci0:0:4:0:       class=0x060401 card=0x2a72103c
chip=0x03f310de rev=0xa1 hdr=0x01
hdac0@pci0:0:5:0:       class=0x040300 card=0x2a72103c
chip=0x03f010de rev=0xa2 hdr=0x00
nfe0@pci0:0:7:0:        class=0x068000 card=0x2a72103c
chip=0x03ef10de rev=0xa2 hdr=0x00
atapci0@pci0:0:8:0:     class=0x010185 card=0x2a72103c
chip=0x03f610de rev=0xa2 hdr=0x00
atapci1@pci0:0:8:1:     class=0x010185 card=0x2a72103c
```

```
chip=0x03f610de rev=0xa2 hdr=0x00
pcib2@pci0:0:9:0:       class=0x060400 card=0x2a72103c
chip=0x03e810de rev=0xa2 hdr=0x01
pcib3@pci0:0:11:0:      class=0x060400 card=0x2a72103c
chip=0x03e910de rev=0xa2 hdr=0x01
pcib4@pci0:0:12:0:      class=0x060400 card=0x2a72103c
chip=0x03e910de rev=0xa2 hdr=0x01
hostb0@pci0:0:24:0:     class=0x060000 card=0x00000000
chip=0x11001022 rev=0x00 hdr=0x00
hostb1@pci0:0:24:1:     class=0x060000 card=0x00000000
chip=0x11011022 rev=0x00 hdr=0x00
hostb2@pci0:0:24:2:     class=0x060000 card=0x00000000
chip=0x11021022 rev=0x00 hdr=0x00
hostb3@pci0:0:24:3:     class=0x060000 card=0x00000000
chip=0x11031022 rev=0x00 hdr=0x00
vgapci0@pci0:2:0:0:     class=0x030000 card=0x16001462
chip=0x94981002 rev=0x00 hdr=0x00
hdac1@pci0:2:0:1:       class=0x040300 card=0xaa381462
chip=0xaa381002 rev=0x00 hdr=0x00
tardis# pciconf -lv
none0@pci0:0:0:0:       class=0x050000 card=0x2a72103c
chip=0x03ea10de rev=0xa1 hdr=0x00
    vendor     = 'Nvidia Corp'
    device     = 'nForce 430 (MCP61) Memory Controller'
    class      = memory
    subclass   = RAM
isab0@pci0:0:1:0:       class=0x060100 card=0x2a72103c
chip=0x03e010de rev=0xa2 hdr=0x00
    vendor     = 'Nvidia Corp'
    device     = 'PCI standard ISA bridge (nForce 430)'
    class      = bridge
    subclass   = PCI-ISA
none1@pci0:0:1:1:       class=0x0c0500 card=0x2a72103c
chip=0x03eb10de rev=0xa2 hdr=0x00
    vendor     = 'Nvidia Corp'
    device     = 'nForce 430 (MCP61) SMBus'
    class      = serial bus
    subclass   = SMBus
none2@pci0:0:1:2:       class=0x050000 card=0x2a72103c
chip=0x03f510de rev=0xa2 hdr=0x00
    vendor     = 'Nvidia Corp'
    device     = 'nForce 430 (MCP61) Memory Controller'
    class      = memory
    subclass   = RAM
ohci0@pci0:0:2:0:       class=0x0c0310 card=0x2a72103c
chip=0x03f110de rev=0xa3 hdr=0x00
    vendor     = 'Nvidia Corp'
    device     = 'nForce 430 (MCP61) USB Controller'
    class      = serial bus
    subclass   = USB
ehci0@pci0:0:2:1:       class=0x0c0320 card=0x2a72103c
chip=0x03f210de rev=0xa3 hdr=0x00
    vendor     = 'Nvidia Corp'
    device     = 'nForce 430 (MCP61) USB Controller'
    class      = serial bus
    subclass   = USB
pcib1@pci0:0:4:0:       class=0x060401 card=0x2a72103c
chip=0x03f310de rev=0xa1 hdr=0x01
    vendor     = 'Nvidia Corp'
    device     = 'nForce 430 (MCP61) PCI bridge'
    class      = bridge
    subclass   = PCI-PCI
hdac0@pci0:0:5:0:       class=0x040300 card=0x2a72103c
chip=0x03f010de rev=0xa2 hdr=0x00
    vendor     = 'Nvidia Corp'
    device     = 'nForce 430 (MCP61) High Definition Audio'
    class      = multimedia
    subclass   = HDA
nfe0@pci0:0:7:0:        class=0x068000 card=0x2a72103c
chip=0x03ef10de rev=0xa2 hdr=0x00
    vendor     = 'Nvidia Corp'
    device     = 'Nvidia Networking Card (nForce 405)'
    class      = bridge
atapci0@pci0:0:8:0:     class=0x010185 card=0x2a72103c
chip=0x03f610de rev=0xa2 hdr=0x00
    vendor     = 'Nvidia Corp'
    device     = 'nForce 430 (MCP61) SATA Controller'
    class      = mass storage
    subclass   = ATA
atapci1@pci0:0:8:1:     class=0x010185 card=0x2a72103c
chip=0x03f610de rev=0xa2 hdr=0x00
    vendor     = 'Nvidia Corp'
```

```
    device     = 'nForce 430 (MCP61) SATA Controller'
    class      = mass storage
    subclass   = ATA
pcib2@pci0:0:9:0:       class=0x060400 card=0x2a72103c
chip=0x03e810de rev=0xa2 hdr=0x01
    vendor     = 'Nvidia Corp'
    device     = 'nForce 430 (MCP61) PCIe bridge'
    class      = bridge
    subclass   = PCI-PCI
pcib3@pci0:0:11:0:      class=0x060400 card=0x2a72103c
chip=0x03e910de rev=0xa2 hdr=0x01
    vendor     = 'Nvidia Corp'
    device     = 'nForce 430 (MCP61) PCIe bridge'
    class      = bridge
    subclass   = PCI-PCI
pcib4@pci0:0:12:0:      class=0x060400 card=0x2a72103c
chip=0x03e910de rev=0xa2 hdr=0x01
    vendor     = 'Nvidia Corp'
    device     = 'nForce 430 (MCP61) PCIe bridge'
    class      = bridge
    subclass   = PCI-PCI
hostb0@pci0:0:24:0:     class=0x060000 card=0x00000000
chip=0x11001022 rev=0x00 hdr=0x00
    vendor     = 'Advanced Micro Devices (AMD)'
    device     = 'Athlon64/Opteron/Sempron (K8 Family)
HyperTransport Technology Configuration'
    class      = bridge
    subclass   = HOST-PCI
hostb1@pci0:0:24:1:     class=0x060000 card=0x00000000
chip=0x11011022 rev=0x00 hdr=0x00
    vendor     = 'Advanced Micro Devices (AMD)'
    device     = 'Athlon64/Opteron/Sempron (K8 Family) Address
Map'
    class      = bridge
    subclass   = HOST-PCI
hostb2@pci0:0:24:2:     class=0x060000 card=0x00000000
chip=0x11021022 rev=0x00 hdr=0x00
    vendor     = 'Advanced Micro Devices (AMD)'
    device     = 'Athlon64/Opteron/Sempron (K8 Family) DRAM
Controller'
    class      = bridge
    subclass   = HOST-PCI
hostb3@pci0:0:24:3:     class=0x060000 card=0x00000000
chip=0x11031022 rev=0x00 hdr=0x00
    vendor     = 'Advanced Micro Devices (AMD)'
    device     = 'Athlon64/Opteron/Sempron (K8 Family)
Miscellaneous Control'
    class      = bridge
    subclass   = HOST-PCI
vgapci0@pci0:2:0:0:     class=0x030000 card=0x16001462
chip=0x94981002 rev=0x00 hdr=0x00
    vendor     = 'ATI Technologies Inc. / Advanced Micro
Devices, Inc.'
    class      = display
    subclass   = VGA
hdac1@pci0:2:0:1:       class=0x040300 card=0xaa381462
chip=0xaa381002 rev=0x00 hdr=0x00
    vendor     = 'ATI Technologies Inc. / Advanced Micro
Devices, Inc.'
    class      = multimedia
    subclass   = HDA
tardis#
```

Yedekleme

Yeni çekirdeği derlemeden önce var olan çalışır durumdaki çekirdeğinizin bir yedeğini almanızda yarar var. Derlediğiniz çekirdek veya derleme sırasında olabilecek bir olumsuzluğun sisteminizi açılmaz hale getirmesi olasıdır. Bu olasılığın ortadan kaldırılması için var olan çekirdeğin sağlam bir yerde bulunmasında yarar var. Çalışır durumdaki çekirdeğin yedeğini /boot/kernel.generic adlı dizine veya sizin belirleyeceğiniz bir isimle /boot dizini altındaki bir başka dizine kopyalayın.

```
tardis# cp -Rp /boot/kernel /boot/kernel.generic
tardis#
```

Bu işlemin ardından halen çalışır durumda olan çekirdeğiniz ile kopyası /boot dizini altında yer alacaktır. Bu işlemin çok sayıda

çekirdek derlemesi yapmanız durumunda disk alanınızı tüketeceğinden endişelenmenize gerek yok. Derlediğiniz çekirdekleri harici bir depolama birimine kopyalayabilir ve gerektiğinde yeniden /boot dizinine kopyalayıp kullanabilirsiniz.

Çekirdek Yapılandırma Dosyası

FreeBSD çekirdeğinin yapılandırması bir metin dosyasının düzenlenmesi ile gerçekleştirilir. Grafik veya metin tabanlı menülerden oluşan bir yapılandırma aracı kullanılmaz. 4.4 BSD sürümündeki yapılandırma aynen korunmuştur. Çekirdek derlemek için kullanılan yapılandırma dosyası aşağıda görülen GENERIC çekirdek yapılandırma dosyası gibidir.

```
options     SCHED_ULE           # ULE scheduler
options     PREEMPTION          # Enable kernel thread
preemption
options     INET                # InterNETworking
options     INET6               # IPv6 communications protocols
options     SCTP                # Stream Control Transmission
Protocol
options     FFS                 # Berkeley Fast Filesystem
```

Çekirdek yapılandırması yukarıda gördüğünüz gibi satırlardan oluşmaktadır. Her bir satır bir çekirdek modülü ve açıklamasından oluşmaktadır. Yapılandırma dosyasına baktığınızda dosyanın sadece bu satırlardan oluşmadığını göreceksiniz. Dosyada cpu, ident, makeoptions, options ve devices ile başlayan satırlar bulunmaktadır. Bu satırlardan bazılarının # ile yorum satırı olarak işaretlendiği ve yapılandırmadan çıkarıldığını görebilirsiniz. X86 mimarisi için hazırlanmış olan çekirdeklerde farklı işlemci sınıfları için verilen desteği 386, 486, 586 ve 686 gibi görmeniz olanaklıdır. Bir çekirdek yapılandırmasında bir tek işlemci için destek bulunmak durumunda değildir. Birden çok işlemci mimarisini x86 örneğindeki gibi destekleyecek şekilde yapılandırabilirsiniz. Ancak aynı çekirdek ile hem SPARC, hem AMD64 hem de x86 işlemcilerde çalıştıramazsınız. Bu nedenle işlemci mimarinize uygun olan seçimleri yapmanız zorunludur.

ident alanı çekirdeğe verilen ismi tanımlar. FreeBSD CD/DVD seti ile kurulum yaptığınızda çekirdeğin adı GENERIC'tir. Bu tanımlama ident bölümünde yer alır.

makeoptions bölümü çekirdeğin derlenmesi sırasında kullanılacak parametreleri tanımlar. Varsayılan değeri DEBUG = -g olarak tanımlıdır. Bu seçenek derleyicinin çekirdeği derlerken hata ayıklama kipinde derlemesini belirtir. Böylelikle hata takibi yapılabilir ve sorunlar için çözümler geliştirilmesi kolaylaşır.

options ile başlayan satırlar ilgili çekirdek fonksiyonları protokoller, dosya sistemleri vb tanımlar.

devices satırları donanım sürücülerini belirtir. Sistemin desteklemesini istediğiniz donanımlara ilişkin sürücülerini bu alanda tanımlarsınız. Devices satırlarında göreceğiniz bazı satırlarda pseudo devices olarak tanımlı olan bölümler bulunur. Bunlar belirli bir donanıma ait değildir. Genel sürücü desteği sunarlar. Aşağıdaki satırda bir pseudo device girdisi görülüyor.

```
# Pseudo devices.
device              loop        # Network loopback
```

Pseudo devices bölümü donanım olarak bulunmayan ama sistem bileşenlerinin kendi aralarında iletişim kurmalarını sağlayan örneğin ağ kartınızın veriyi tarayıcınıza iletmesi için kullanılan donanımları tanımlar. Pseudo devices bölümünün çekirdek yapılandırmasında bulunmasında yarar var.

Yapılandırma dosyasını oluşturmak için en başından başlamanıza ve yazmanıza gerek yoktur. Bunun yerine var olan yapılandırma dosyalarının kopyalarını alıp bu dosyalar üzerinde düzenleme yapmak

daha kolaydır. GENERIC çekirdek yapılandırması donanımınız için gereken tüm bileşenleri sunar. GENERIC yapılandırması /usr/src/sys dizininde yer alır. Bu dizinde sadece sisteminize ait olan mimari dışında FreeBSD tarafından desteklenen tüm mimarileri görebilirsiniz. Çekirdek derleme için gerekli olan yapılandırma dosyası ile /usr/src/sys/mimari/conf dizininde yer alır. /usr/src/sys dizinine bakacak olursanız bu dizinde FreeBSD tarafından desteklenen diğer mimarilerin de yer aldığını görebilirsiniz. İlgili dizinlere bakarak farklı mimariler için hazırlanmış GENERIC yapılandırmalarını da inceleyebilirsiniz.

```
[goksin@tardis /usr/src/sys/amd64/conf]$ ls -l
total 42
-rw-r--r--  1 root  wheel    479 25 Eki  2009 DEFAULTS
-rw-r--r--  1 root  wheel  12645 10 Kas  2009 GENERIC
-rw-r--r--  1 root  wheel    703 25 Eki  2009 GENERIC.hints
-rw-r--r--  1 root  wheel    136 25 Eki  2009 Makefile
-rw-r--r--  1 root  wheel  14527 25 Eki  2009 NOTES
-rw-r--r--  1 root  wheel   5841 25 Eki  2009 XENHVM
[goksin@tardis /usr/src/sys/amd64/conf]$
```

GENERIC yapılandırmasının bulunduğu dizinde yukarıda gördüğünüz gibi başka dosyalar da bulunmaktadır. Bu dosyalar çekirdek derlerken kullanacağınız yapılandırma için kritik öneme sahiptir.

DEFAULTS dosyası, söz konusu mimari için yapılandırmada yer alan standart donanım listesidir. Bu dosyadaki donanımlar genel olarak söz konusu mimari için gerekli olan donanımların bir listesidir. Eğer olabildiğince küçük bir çekirdek hazırlamak istiyorsanız bu dosyayı esas alabilirsiniz.

GENERIC dosyası söz konu mimari için kullanılan standart yapılandırmadır. Sisteminizde kurulu olan çekirdeğin yapılandırmasıdır.

GENERIC.hints dosyası kurulum sırasında /boot/device.hints dosyası olarak kopyalanan dosyadır. Eski donanımlara ait yapılandırma parametrelerini barındırır.

NOTES çekirdek yapılandırmanızda kullanabileceğiniz mimariye özel yapılandırma seçeneklerinin yer aldığı dosyadır. Bu dosyadaki yapılandırma seçeneklerini kendi yapılandırmanızı hazırlarken kullanabilirsiniz. Bu dosya ile aynı isimde olan ama platform bağımsız yapılandırma seçeneklerinin yer aldığı bir NOTES dosyası da bulunur. Bu dosya ise /usr/src/sys/conf dizininde aynı isimle bulunur. Bu dosyaları çekirdek yapılandırmanızı hazırlamadan önce incelemenizde yarar var.

Çekirdek derlemede kullanılacak olan yapılandırma dosyanızı hazırlarken GENERIC dosyasından yararlanabilirsiniz ancak doğrudan bu dosya üzerinde işlem yapmamanız gereklidir. Dosyayı başka bir isimle kopyasını oluşturup bu kopya üzerinde işlem yapmanız uygun olur. Örneğin GENERIC dosyasının TASLAK adlı bir kopyasını alıp bu dosya üzerinde işlem yapın. Kendi sistemimde RAID donanım bulunmamaktadır. Dolayısıyla da çekirdekte RAID, teyp yedekleme ve disket sürücü desteğine gereksinim duymuyorum. Bu nedenle /var/run/dmeg.boot dosyasını esas alırsam aşağıda gördüğünü ATA yapılandırması bölümünü

```
# ATA and ATAPI devices
device             ata
device             atadisk           # ATA disk drives
device             ataraid           # ATA RAID drives
device             atapicd           # ATAPI CDROM drives
device             atapifd           # ATAPI floppy drives
device             atapist           # ATAPI tape drives
options     ATA_STATIC_ID     # Static device numbering
```

yeniden düzenleyerek aşağıdaki gibi yapabilirim.

```
# ATA and ATAPI devices
device             ata
device             atadisk           # ATA disk drives
```

```
device              atapicd             # ATAPI CDROM drives
options     ATA_STATIC_ID     # Static device numbering
```

Bu yapılandırmada ATA desteğini sağlayan satır olan ilk satır korunurken sadece sistemde olmayan donanımların desteğini kaldırdım. Bu donanımların bulunduğu sistemde yeni derleyeceğim çekirdeği kullanmam durumumda çekirdek bu donanımları desteklemeyecektir. Benzer biçimde çekirdek içerisinde sisteminizde olmayan donanımları kaldırarak çekirdeğin boyutunu küçültebilirsiniz. Bu küçültme işlemi – yani çekirdekten bugün için gerek duymadığınız donanımların desteğinin kaldırılıp boyut olarak azaltılması – zorunlu olarak yapılması gereken bir işlem değildir. Özellikle de güncel sistemlerin yapısı dikkate alındığında belleğin boyutunun küçültülmesinin ciddi bir yarar sağlamayacağı görülebilir. Öte yandan sonradan eklenecek olan RAID kartı gibi donanımların çekirdek tarafından desteklenmemesi durumunda yapılması gereken çekirdeğin yeni baştan derlenmesidir. Her donanım eklendiğinde çekirdeği yeniden derlemek tavsiye edilen bir işlem değildir. Öte yandan x86 mimarisi için bazı avantajları olacaktır.

x86 için hazırlanmış olan GENERIC çekirdek yapılandırmasında aşağıdaki bölümü görebilirsiniz.

```
cpu         I486_CPU
cpu         I586_CPU
cpu         I686_CPU
ident       GENERIC
```

Bu yapılandırma yerine doğrudan işlemcinizin özelliklerine uygun mimariyi seçip diğerlerini kaldırabilirsiniz. İşlemcinizin özelliklerine göz atmak için /var/run/dmesg.boot dosyasının ilk satırlarına göz atmanız yeterlidir. Bu satırlarda işlemcinin özellikleri ve ilgili bilgiler yer alır. Bu bilgileri kullanarak i486 ve i586 işlemcilerini kaldırıp sadece i686 satırını bırakabiliriz. Bu durumda çekirdek sadece i686 sınıfı işlemcileri destekleyecek bir diğer deyişle sadece bu işlemciler üzerinde çalışacaktır.

Sisteminizi hangi amaçla kullandığınız da çekirdek yapılandırmasında belirleyici olacaktır. Örneğin sunucu iş istasyonu vb sistemlerde ağ desteği çekirdeğin çoklu görev -multitasking- farklı dosya sistemlerinin desteklenmesi vb özelliklere sahip olması zorunludur. Bu nedenle çekirdek yapılandırmasında genelde aşağıdaki yapılandırma seçeneklerinin eklenmesi gereklidir.

```
options     SCHED_ULE           # ULE scheduler
options     PREEMPTION          # Enable kernel thread
preemption
options     INET                # InterNETworking
options     INET6               # IPv6 communications protocols
options     SCTP                # Stream Control Transmission
Protocol
options     FFS                 # Berkeley Fast Filesystem
options     SOFTUPDATES         # Enable FFS soft updates
support
options     UFS_ACL                   # Support for access
control lists
options     UFS_DIRHASH         # Improve performance on big
directories
options     UFS_GJOURNAL              # Enable gjournal-based
UFS journaling
options     MD_ROOT                   # MD is a potential root
device
options     NFSCLIENT           # Network Filesystem Client
options     NFSSERVER           # Network Filesystem Server
options     NFSLOCKD            # Network Lock Manager
options     NFS_ROOT            # NFS usable as /, requires
NFSCLIENT
options     MSDOSFS                   # MSDOS Filesystem
options     CD9660                    # ISO 9660 Filesystem
options     PROCFS                    # Process filesystem
(requires PSEUDOFS)
options     PSEUDOFS            # Pseudo-filesystem framework
```

Bu seçenekler içerisinde INET, IPv4 desteği sağlarken INET6, IPv6 desteği sağlamaktadır. Dosya sistemi destekleri olarak UFS,

SOFTUPDATES ve UFS dosya sistemine özel araçların kullanılabilmesi için gerekir. UFS_ACL dosya sistemindeki kontrol özellikleri için gereklidir. UFS_DIRHASH ise çok sayıda dosyanın/dizinin bulunduğu dosya sistemlerinde performans için zorunlu olan bir bileşendir. MD_ROOT bileşeni sisteminizin standart olarak UFS formatlı bir diskten başka bir donanımı / olarak kullanmanızı sağlar. NFSCLIENT ve NFSSERVER seçenekleri dosyalarını NFS ile paylaştırmanızı ve ağ üzerindeki sistemlerdeki NFS paylaşımlarına erişmenizi sağlar. MSDOSFS ve CD9660 adından anlaşılıyor. PROCFS için PSEUDOFS zorunludur. PROCFS dosya sistemini kullanmak ve gerektiğinde /proc altına bağlamak için gereklidir. Aynı zamanda PSEUDOFS sisteme giriş yaptığınız terminaller için de gereklidir. Bunları çekirdek modülü olarak da kullanabilirsiniz ama modülleri eklemek ve kaldırmak ile uğraşmak yerine çekirdekte yer almaları daha uygun olacaktır.

Eğer FreeBSD eski sürümleri için hazırlanmış olan yazılımları da kullanmak durumunda iseniz bu durumda aşağıdaki satırlarında yapılandırmaya eklenmesinde yarar var. Bu satırlar geriye dönük uyumluluk desteğini sağlayacaktır. Aşağıdaki yapılandırma AMD64 için hazırlandığından 32 bit uyumluluk kipi de eklenmiştir.

```
options     COMPAT_43TTY            # BSD 4.3 TTY compat
(sgtty)
options     COMPAT_IA32       # Compatible with i386 binaries
options     COMPAT_FREEBSD4         # Compatible with FreeBSD4
options     COMPAT_FREEBSD5         # Compatible with FreeBSD5
options     COMPAT_FREEBSD6         # Compatible with FreeBSD6
options     COMPAT_FREEBSD7         # Compatible with FreeBSD7
```

Sisteminizde veritabanı sunucuları yer alacak ise aşağıdaki seçeneklerin de yer alması gerekecektir. Bazı veritabanı yazılımları bunlara gereksinim duymaktadır.

```
options     SYSVSHM                 # SYSV-style shared memory
options     SYSVMSG                 # SYSV-style message
queues
options     SYSVSEM                 # SYSV-style semaphores
```

Yapılandırma dosyasındaki seçenekler ilgili oldukları donanım için gereklidir. Sisteminizde bulunan donanıma göre yapılandırmanızı düzenleyebilirsiniz. x86 mimarisi için hazırlanan bir yapılandırmada aşağıdaki bölüme SMP desteği için gereksinim duyulur.

```
# To make an SMP kernel, the next two lines are needed
options     SMP             # Symmetric MultiProcessor
Kernel
device            apic              # I/O APIC
```

Aynı bölüm ise AMD64 için aşağıdaki gibi eklenmelidir.

```
# Make an SMP-capable kernel by default
options     SMP             # Symmetric MultiProcessor
Kernel
```

Burada SMP desteği için bir noktayı aydınlatmakta yarar var. FreeBSD i386 SMP yapılandırması sadece Intel SMP spec uyan işlemciler ile uyumludur. 486 işlemciler ile uyumlu değildir. Benzer olarak SPARC64 işlemciler ile de uyumlu değildir. Bu uyumluluk söz konusu çekirdeğin bu mimarilerde sadece çalışacağı anlamına gelir. SMP desteği çekirdeğe eklenmiş olsa da kullanılmayacaktır.

Seçeneklerin yer altığı options bölümünden sonra sürücülerin bulunduğu kısım yer almaktadır. Bu kısımda zorunlu olmadıkça değişiklik yapmayın. ATA ve ATAPI drives bölümüne dek olan kısmı olduğu gibi bırakın. Bunlar içerisinden sadece disket sürücü kısmını disket sürücünüz yoksa kaldırabilirsiniz.

Bundan sonraki kısım ise ATA ve ATAPI drives bölümü ile RAID ve SCSI donanım sürücülerinin yer aldığı bölümlerdir. Bu bölümlerde RAID ve SCSI donanımlardan sisteminizde bulunmayanları kaldırabilirsiniz. Ancak sonradan gereksinim olacağını düşünerek ileride

RAID kullanmayı düşünüyorsanız RAID ve SCSI bölümlerini korumanızda yarar var. Sonrasında gelen ağ donanım bölümünde ise kullanmadığınız veya kullanmayacağınız ağ kartlarına ait sürücüleri kaldırabilirsiniz. PSEUDEO DEVICES bölümü zorunlu olan bir bölümdür. Bu alanı korumanız gereklidir. USB ve firewire desteğini de gerek duymuyorsanız kaldırabilirsiniz. PF'in kalması önerilir. Sisteminize erişimi kontrol etmek için önemli bir bileşendir. Güvenlik konusunu ele aldığımız yazılarda anımsayacaksınız. Bu işlemlerden sonra çekirdek yapılandırma dosyanız sisteminizdeki donanımları destekleyecek şekilde küçültülmüş olacaktır.

Derleme İşlemi

Sisteminizi destekleyecek büyüklükte bir çekirdek yapılandırması hazırladıktan sonra çekirdeği derleyerek sistemi bu yeni çekirdek ile başlatarak deneyebilirsiniz. Derleme işlemini yapmadan önce çekirdeğinizi tanımlayacak olan bir isim belirtmeniz gereklidir. Bunu /etc/make.conf dosyasında tanımlamanız veya derleme işlemi sırasında belirtmeniz gerekir.

```
# cd /usr/src
# make KERNCONF=<isim> kernel
```

Bu komut sizin belirttiğiniz <isim> ile anılan ve yukarıda hazırladığınız yapılandırmaya göre derlenecek olan çekirdeğin derleme işlemini başlatır. İlk olarak config(8) çalışacak ve yapılandırmada hatalı girdi olup olmadığını kontrol edecektir. Eğer config(8) yapılandırmada bir hata bulursa işlemi sonlandırıp hata mesajı döndürecektir. Örneğin UFS desteğini dahil etmeden UFS disklerden açılış yapılmasını etkinleştirdiyseniz bu durumda config(8) hata döndürecektir. Döndürülen hata mesajı söz konusu hatanın kaynağını açık şekilde belirttiği için ilgili satırı yapılandırma dosyasına ekleyerek bu hatayı giderebilirsiniz. Bazen hata mesajları imla hatalarına işaret eder. Örneğin Ipv6 desteğini belirten INET6 yerine INT6 yazdıysanız

```
<isim>: unknown option "INT6"
```

hata mesajı döndürülecektir. Çekirdek yapılandırma dosyasındaki seçenekleri iyi biliyorsanız bu mesajın işaret ettiği hatayı kolaylıkla çözebilirsiniz.

config(8) yapılandırmadaki hataları taradıktan ve hatasız bir yapılandırma dosyası elde edildikten sonra derleme işlemini başlatacaktır. Güncel donanımlarda bu işlem yaklaşık olarak bir saat kadar veya daha az sürmektedir. Derleme işlemini terminalden izleyebilirsiniz. Derleme işlemi tamamlandığında kullanılmakta olan çekirdek yani /boot/kernel dizini yeni derlenmiş çekirdeğin dizini olacağı için eski çekirdek /boot/kernel.old dizinine taşınacaktır. Sisteminizi yeniden başlattığınızda sistem bu /boot/kernel dizininde yer alan çekirdek ile yani derlediğiniz çekirdek ile açılacaktır. Açılış sırasında kendi derlediğiniz çekirdeğin adını ekranda görebilirsiniz.

Derleme Sırasında Hata Mesajı Döndürülürse

Derleme işlemleri sırasında hata mesajları döndürülebilir ve derleme işlemi sonlandırılabilir. Son yapılan işlemde en son satır hata mesajı değildir. Tersine hata mesajı birkaç satır yukarıda yer alır. Hata mesajını gördüğünüz satırı incelerseniz hataya ilişkin bilgi edinebilirsiniz. Ayrıntılı bilgi edinmek için ise hata mesajını kopyalayıp internette arama yaparsanız daha fazla bilgiye ulaşabilirsiniz. Böylelikle problemi daha kolay çözersiniz.

Derleme işleminin tamamlanmaması sisteminizde yarım kalmış bir çekirdek kurulduğu anlamına gelmez. FreeBSD sisteminiz çekirdeğin tamamının -modüller de dahil- derlenmeden kurulmasına izin vermez. Ancak ve ancak tüm derleme işlemi tamamlandıktan sonra çekirdek ve ilgili modülleri kurulur. Dolayısıyla da sisteminiz halen eski çalışan çekirdek ile açılacak demektir. Hatayı giderdikten sonra /usr/src dizinini make clean komutu ile temizleyip baştan derleme işlemine yeniden başlayabilirsiniz.

Eski Çekirdek İle Sistemi başlatmak

Derleme işlemi başarılı bir şekilde sonuçlanmış olsa da yapılandırmadaki eksik bileşenlerden ötürü çekirdeğiniz bazı donanımlarınızı desteklemiyor olabilir. Bu durumda çalışır durumdaki eski çekirdeği kullanarak sistemi yeniden başlatabilirsiniz. Eski çekirdek /boot dizini altında yer aldığı için sisteminizi yeniden başlatıp açılış menüsünde "Loader promt" seçeneğini seçerek sistemi başlatın.

Yeni derlenmiş çekirdek yüklendiği için öncelikle onu bellekten kaldırmamız gerekir. Bunun için

```
ok unload
```

komutunu verdiğinizde açılış sırasında yüklenen çekirdek bellekten kaldırılır. Yedeğini aldığımı çalışır durumdaki çekirdeği ve kullanmak istediğini modülleri tanımlayarak belleğe yükleyebilirsiniz. Çalışır durumdaki çekirdeğin GENERIC olduğunu ve yukarıda anlatıldığı gibi yedeğini aldıysanız yedeklediğini çekirdek ile veya /boot/kernel.old ile sistemi başlatabilirsiniz. Kernel.generic ile sistemi başlatalım:

```
ok load /boot/kernel.generic/kernel
ok load /boot/kernel.good/acpi.ko
ok boot
```

Sisteminiz eski çekirdek yani /boot/kernel.generic ile açılacaktır. Son derlediğiniz çekirdekteki hataları giderip yeniden derleyip sistemi yeni çekirdek ile yeniden başlatabilir veya eski çekirdeğinizi kullanmaya devam edebilirsiniz.

GENERIC Çekirdeğe Ekleme/Çıkarma Yapmak

Çekirdek derlemek için GENERIC yapılandırmasını değiştirmek dışında bu yapılandırmaya eklemeler ve çıkarmalar yaparak da kendi yapılandırmanızı oluşturabilirsiniz. Bunun için NOTES dosyasında yer alan diğer yapılandırma seçeneklerini kullanabilirsiniz. Bu dosyada GENERIC yapılandırmasında yer alamayan ama FreeBSD çekirdeğinin tüm işlevlerinin yapılandırma dosyasına nasıl konulacağına ilişkin bilgi verilmektedir. Aslında NOTES dosyası adından da anlaşılacağı üzere notlar barındıran bir dosyadır.

Bu dosyada yer alan notlar özel donanımlara veya mimarilere ilişkin seçenekler bulunur. NOTES dosyası FreeBSD kaynak kodunun aktarılabildiği her mimari için ayrı ayrı hazırlanmıştır. Bu mimarileri /usr/src altında ilgili dizin adı ile görebilirsiniz. Mimariden bağımsız olan çekirdek özellikleri de /usr/src/sys/conf/NOTES dosyasında bulabilirsiniz. Bu dosyadaki seçeneklerden bazıları GENERIC yapılandırmasında yer alır. Bunlardan GENERIC veya kendi yapılandırmanızda yer almayan çekirdek fonksiyonlarından eklemek istedikleriniz bulunuyorsa yapılandırma dosyasında yer vermeniz yeterlidir.

Eklemek istediğiniz çekirdek özelliklerini GENERIC çekirdek yapılandırmasına eklemek için GENERIC dosyasının kopyasını alıp üzerinde çalışmanıza gerek yoktur. FreeBSD çekirdek geliştiricileri bu işlemi kolaylaştırmak için include özelliğini geliştirmişlerdir. Eklemek istediğiniz özellikleri çekirdek yapılandırma dosyasının en sonuna yazıp önceki satırda ise

```
include GENERIC
```

yazmanız ve dosyanızı kayıt etmeniz yeterlidir. Yapılandırma dosyasınız oldukça kısaldı. Bundan sonra derlemeye başlayabilirsiniz. Aşağıdaki örnek dosyada REISERFS dosya sistemini eklediğimiz bir yapılandırma görülüyor.

```
ident       RESISER+GENERIC
include     GENERIC
options     REISERFS
```

Bu yapılandırmayı bu şekilde yazmak yerine doğrudan GENERIC dosyasının içeriğini yazarak da yapabilirdik. Bu düşünce ilk bakıştı doğrudur ama FreeBSD aktif olarak geliştirildiği için herhangi iki sürüm arasında GENERIC yapılandırmasında önemli değişiklikler olacaktır. Bu değişiklikler sistemin kaynak koddan derlenerek bir üst sürüme güncellendiği durumlarda sorunların ortaya çıkmasına neden olacaktır. Bu nedenle include GENERIC ile değişikliklerin neden olacağı hataların ve zaman kayıplarının önüne geçilebilir.

GENERİC çekirdekten bir çekirdek özelliğini çıkartmak isterseniz benzer şekilde çıkarılacak olan seçeneği veya seçenekleri tanımlayan bir nooption satırı gerekeni yapacaktır. Nooption, include tersine çıkarmak istediğiniz özelliklerin çekirdekten çıkarılmasını sağlar.

```
ident       GENERIC-gerekyok
include     GENERIC
nooption    <bileşen-adı>
```

Derlenen Çekirdeğin Diğer Sistemlere Kurulumu

Eğer elinizde aynı donanıma sahip birden çok sayıda sistem bulunuyorsa bu durumda bu sistemlerden birisinde derlediğiniz çekirdeği diğer sistemlerde de kullanabilirsiniz. /boot/kernel dizinini bir tar dosyası haline getirip bunu diğer sistemlere kopyalayıp tar dosyasını açtığınızda çekirdek diğer sistemlere de aktarılmış olur. Bu sistemleri yeniden başlattığınızda derlediğiniz çekirdek çalışacaktır.

```
nextboot(8) İle Çekirdek Testi
```

FreeBSD sisteminiz ister elinizin altında ister uzaktaki bir sistem odasında olsun, nextboot(8) ile derlediğiniz çekirdekleri bir sonraki sistem başlangıcında çalıştırabilirsiniz. Bu araç özellikle uzak sistemleri yönetmek durumunda iseniz ve sistemde doğrudan fiziksel erişim olanağınız yok ise çok işe yarayacaktır.

nextboot(8) ile yeni derlediğiniz ve denemek istediğiniz çekirdeği bir sonraki açılışta denemek için en iyi yol denenecek olan çekirdeğin/boot/kernel dışında bir dizine kopyalanması veya derleme işlemi sırasında yeni çekirdeğin hangi dizine kopyalanacağının belirtilmesidir.

İlk durum olan çekirdeğin /boot/kernel dizinine kurulmasından önce çalışır durumdaki çekirdeğin yedeğinin bulundurulması ve yeni derlenen çekirdeğin de farklı bir dizine kopyalanması gereklidir. Elimizde GENERIC çekirdeğin olduğunu varsaydığımızda derlenen çekirdeğin de kernel.test dizinine kopyalanması ilk adımlardır.

```
# mv /boot/kernel /boot/kernel.test
# mkdir /boot/kernel
# cp /boot/kernel.generic/* /boot/kernel/
```

Diğer seçenek ise çekirdek derleme işlemi sırasında INSTKERNAME parametresine yeni çekirdeğin kopyalanacağı dizinin belirtilerek make(1)'in bu dizine kopyalama yapmasını sağlamaktır.

```
# cd /usr/src
# make KERNCONF=<isim> INSTKERNAME=test kernel
```

Çekirdek <isim> ile belirtilen isme sahip olsa da kopyalanacağı dizin /boot/kernel.test olacaktır. Bu dizindeki çekirdeğin sizin derlediğiniz ve denenecek olan çekirdek olduğunu unutmayın. Derleme işleminin ardından /boot/kernel.test dizinine kopyalanacaktır. Bu aşamadan sonra nextboot(8) ile çalıştırılacak olan çekirdeğin bulunduğu dizinin tanımlanması gereklidir.

```
# nextboot -k kernel.test
```

Bu komutu verdikten sonra sistemi yeniden başlatabilirsiniz. loader(5) /boot/kernel7 dizinindeki çekirdeği değil /boot/kernel.test/ dizinindeki çekirdek ile sistemi başlatacaktır. Denemeleriniz olumlu ise ve yeni

çekirdeğin uygun olduğuna kara verdiyseniz asıl çekirdeği yedekleyip yeni çekirdeği /boot/kernel/ dizinine kopyalamanız yeterlidir.

```
# mv /boot/kernel /boot/kernel.generic
# mv /boot/kernel.test /boot/kernel
```

İşlem tamam. Artık sistemi yeniden başlattığınızda yeni derlenen çekirdek ile sistem başlatılacaktır.

Çekirdek İle İlgili Birkaç Anımsatma

* Çekirdek derlemek problem(ler)i çözmez!
* Çekirdek derlemek demek sizin bu işi iyi bildiğiniz anlamına gelmez! Sadece derleme nasıl yapılır biliyorsunuz demektir.
* GENERIC çekirdek yeterlidir. Çekirdek derlemek zorunlu de,ğildir ve uç sistemler için gerekebilir!
* İster GENERIC ister kendi yapılandırmanızı kullanın çekirdek hata mesajlarını asla göz ardı etmeyin!

Hata mesajları içerisinde dikkat etmeniz gerekenler ACPI, SMP ve Lock Order reversal mesajları ile x86 sistemlerdeki PAE mesajlarıdır.

ACPI hata mesajları genelde Microsoft ürünleri için tasarlanmış donanımlardaki güç yönetimi uyumsuzluklarından kaynaklanır. DESIGNED for Windows veya $windows CAPABLE etiketli sistemlerde ACPI hataları ile karşılaşabilirsiniz. Hatalı ACPI yapılandırmasına sahip bu sistemlerde farklı işletim sistemleri bir şekilde uyumlu hale getirilmiştir ve sorunsuz olarak çalışabilirler ama hatalı uygulamalar hatayı gidermez. Intel x86 ailesinin 4 Gb'tan daha büyük bellekleri adresleyebilmesi için geliştirilen PAE - Physical Address Extension bazı eski donanımlarda sorun çıkarmaya adaydır. Bu sistemlerde PAE için gereken donanım bulunmadığı halde güncel FreeBSD sürümlerindeki GENERIC çekirdek içerisinde bu destek bulunur ama gerekli olan donanımların desteği yer almaz. Kendi derlediğiniz çekirdekte bu donanımları eklediyseniz ve eski bir sistemde çalışıyorsanız sorun çıkması kuvvetle olasıdır.

SMP hataları ise teoride işletim sisteminin üzerindeki yükü işlemcilere eşit olarak paylaştırdığını öngörür. Uygulamada SMP desteğinin sorunsuz olarak çalışması için yapılandırmada SMP ile birlikte APIC seçeneklerinin de bulunması gereklidir. Bunlara rağmen sorun oluyorsa sysctl ile SMP ve ACPI kapatılarak sistemi çalıştırabilir ve sorunları gidermeyi deneyebilirsiniz. Son önemli hata ise Lock Order Reversals adı verilen SMP çekirdek yapılandırmasının önemli bir bileşeninde ortaya çıkabilir. Lock Order Reversal SMP işlemcilerde işlemcilerin sis-temdeki farklı süreçler tarafından paylaşılması için gereklidir. LOR olarak kısa adı ile anılır. LOR hataları genelde RELEASE ve STABLE sürümlerinde ender olarak görülür veya hiç görmezsiniz. CURRENT kullanıyorsanız LOR hata mesajlarını görmeniz daha olasıdır. Çekirdek hata ayıklama özelliğini aktif kıldıysanız bu mesajları görebilirsiniz. LOR hatası çekirdek geliştiricilerin SMP kodunda bir şeylerin beklendiği gibi gitmediğini belirtir. Bu hatayı aldıysanız durumu FreeBSD geliştiricilerine aktarmakta tereddüt etmeyin. Aktarmadan önce ise hatanın daha önceden bildirilip bildirilmediğini bir kontrol edin. [1] Hatayı aktardığınızda FreeBSD topluluğu size eğer hatayı kendi derlediğiniz çekirdek ile alıyorsanız bu durumda çekirdeği derlemek yerine GENERIC kullanmanız veya derleyecekseniz de işi doğru yapmanızı kibar bir dille belirtecektir. Alınganlık göstermeyin. Bu hepimizin başına en az bir defa gelmiştir.

[1] http://sources.zabbadoz.net/freebsd/lor.html

BSD - XIX

NetBSD ve OpenBSD'de çekirdek/kernel derlemek

Gökşin Akdeniz
goksin@enixma.org

NetBSD ve OpenBSD'de Çekirdek/Kernel Derlemek

NetBSD, OpenBSD ve diğer BSD sistemlerde çekirdek derlemek zor bir işlem değildir. FreeBSD için anlatılan çekirdek derleme süreci diğer NetBSD, OpenBSD, vb için de bazı noktalarda farklı olmakla birlikte hemen hemen aynıdır. Çekirdeği derlemek için her zaman kaynak kodlara gereksinim vardır. Eğer sisteminizi kurarken kaynak kodları da diske kopyaladıysanız çekirdek derlemeye başlayabilirsiniz.

NetBSD'de Çekirdek Derlemek

NetBSD iso dosyalarında kaynak kodlar bulunmaz. Kurulum için kullandığınız iso dosyası sadece derlenmiş ikili dosyaları veya diğer adı ile “sets”i barındırır. Bunların içinde ise kaynak kodlar bulunmaz. Kaynak kodlar NetBSD ftp sunucuları ile yansılarında bulunur. Kaynak kodları üç farklı şekilde edinebilirsiniz.

İlk seçenek CVS (Concurrent Versions System) kullanılarak NetBSD kaynak kodlarını indirmektir. cvs(1) ile kaynak kodları edinmeden önce kullandığınız kabuk ortamında bazı hazırlıklar yapmanız gereklidir. C Kabuğu vb için ortam değişkenlerinin aşağıdaki belirtildiği gibi tanımlanması gerekir.

```
# setenv CVS_RSH ssh
# setenv CVSROOT anoncvs@anoncvs.NetBSD.org:/cvsroot
```

Bourne kabuğunu vb kullanıyorsanız

```
# export CVS_RSH="ssh"
# export CVSROOT="anoncvs@anoncvs.NetBSD.org:/cvsroot"
```

Gerekli olan değişken tanımlamaları yapıldıktan sonra ise kullanmakta olduğunuz sürüme ait olan kaynak kodu edinmek için cvs ile kullandığınız sürümü belirterek aşağıdaki komutu vermeniz yeterlidir

```
# cd /usr
# cvs checkout -r <KULLANILAN_SÜRÜM> -P src
```

Kullanılan sürüm olarak kullanmakta olduğunuz sürümü -kendi kullandığım NetBSD 5.0.2 olduğu için netbsd-X-Y-Z-RELEASE olarak belirtin- Örneğin 5.0.2 sürümü için netbsd-5-0-2-

RELEASE olarak belirtmek zorunludur. Farklı bir sürümü tanımlarsanız hatalı bir işlem yapmış olur ve siteminizi derlerken “downgrade” yapma riski de söz konusudur. Eğer X sunucusuna ait olan kaynak kodları da indirmek istiyorsanız yukarıdaki komutu

```
# cvs checkout -r <KULLANILAN_SÜRÜM> -P xsrc
```

olarak tanımlamanız gereklidir.

cvs(1) kullanmadan ftp sunucusundan kullandığınız sürüme ait olan kaynak kodları tgz dosyaları olarak indirebilirsiniz. Kaynak kodlar netBSD ftp sunucularında

```
/pub/NetBSD/NeBSD-<SÜRÜM>/source/sets/
```

dizininde bulunur. İndirmek için

```
# ftp -i ftp://ftp.NetBSD.org/pub/NetBSD/NetBSD-
5.0.2/source/sets/
```

komutunu vermeniz yeterlidir. Bu komutun işlenmesinin ardından ftp komutlarını kullanabilirsiniz.

```
ftp> mget *.tgz
local: gnusrc.tgz remote: gnusrc.tgz
229 Entering Extended Passive Mode (|||60299|)
150 Opening BINARY mode data connection for 'gnusrc.tgz'
(8614045 bytes).
```

Dosyalar indirildiğinde elinizde 5 adet dosya bulunacaktır.

```
# ls *.tgz
gnusrc.tgz        sharesrc.tgz           src.tgz
syssrc.tgz        xsrc.tgz
```

Bu dosyaları /usr/src altına açtığınızda kaynak kodlar edinilmiş olur.

NetBSD kaynak kodları edinmenin son yolu ise kaynak kodları barındıran ISO dosyalarını indirip bunları bir CD'ye aktarıp /usr/src altına kopyalamanızdır. Kaynak kodları barındıran iso dosyasına ftp sunucusundan erişebilirsiniz.[1]

NetBSD'nin desteklediği tüm platformlara ait çekirdek kaynak kodları /usr/src/sys/arch dizini altında yer alır. Bu dizinin altında yer alan altdizinlerin altında conf dizini yer alır. Buraya göz attığınızda söz konusu platform için NetBSD geliştirme ekibi tarafından hazırlanmış olan örnek çekirdek yapılandırmasını görebilirsiniz. GENERIC yapılandırması standart olarak kullandığınız çekirdek yapılandırmasıdır. Genel kural olarak /var/run/dmesg.boot dosyası donanım ile çekirdek yapılandırmasının eşleşen bileşenlerini verir. Bu eşleşmeyi başlangıç olarak kullanarak kendi çekirdek yapılandırmanızı oluşturabilirsiniz. NetBSD /var/run/dmesg.boot dosyasının içeriği FreeBSD için farklı olarak aşağıdaki yapıdadır:

```
XXX at YYY
```

Bu satırları dosyadan çekerek donanımınıza uygun bir çekirdek yapılandırması için gereken bilgiyi edinebilirsiniz.

Başlangıç olarak GENERIC yapılandırmasını başka bir isimle kopyalayarak işleme başlayabilirsiniz. Yapılandırmaya göz attığınızda seçenekler hakkında bilgi veren yorum satırlarını görebilirsiniz. Bu satırlar bildiğimiz gibi # ile başlamaktadır. Eğer seçenekler hakında daha ayrıntılı bilgiye gereksinim duyuyorsanız options(4) kılavuz sayfasına başvurabilirsiniz. NetBSD çekirdeğini derlemek için

```
# cp GENERIC TEST
# vi TEST
```

vi ile TEST yapılandırmasını düzenlerken GENERIC çekirdek içerisinde yer alan ama sizin sisteminizde bulunmayan donanımlara ilişkin desteği kaldırabilirsiniz. Bunlara ek olarak bazı çekirdek servislerini

örneğin NFS istemci desteği, Linux uyumluluk kipi vb ek bileşenleri kullanmak için dahil edebilir veya kaldırabilirsiniz. Bunlara ek olarak çekirdek parametrelerini de düzenleyebilirsiniz. Salt kullandığınız donanımı dikkate alırsanız GENERIC kernel ile kıyaslandığında daha küçük bir çekirdek elde etmeniz olanaklıdır.

Çekirdek yapılandırmanız tamamladıktan sonra çekirdeği derleyebilirsiniz. Bu işlemi elle veya NetBSD için hazırlanmış olan ve sistemi kaynak koddan derlemek ve inşa etmek için kullanılan build.sh ile yapabilirsiniz. Build.sh uygulaması başlı başına bir konu olduğu için burada yer vermiyoruz. Çekirdeği doğrudan kaynak koddan make ile derleme yöntemi ile devam edeceğiz.

Yapılandırma dosyanızda bir sorun olup olmadığını görmek için en kolay yol yapılandırmanızın config(8) ile denenmesidir. Eğer bir hata yok ise aşağıdaki komutun çıktısı size derleme yapacağınız dosyaların bulunduğu dizini döndürecek ve bir sonraki işleme geçmenizi isteyecektir. Aksi durumda çekirdek yapılandırmanızı hatasız hale getirinceye dek bu işlemi tekrarlamanız gerekecektir.

```
# config TEST
Build directory is ../copmpile/TEST
Dont!'forget to run “make depend”
#
```

config(8) döndürdüğü dizinde derleme işlemi için gerekli olan tüm dosyalar yer almaktadır. Bunlar içerisinde header dosyaları ile çekirdeğe ait olan Makefile dosyası da bulunmaktadır. Derleme işlemine geçmeden önce sırasıyla aşağıdaki komutları çalıştırmamız gereklidir.

```
# cd ../compile/MYKERNEL
# make depend
# make
```

Bu aşamalarda make(1) hata vererek derleme işlemini sonlandırabilir. Bu ender de olsa gerçekleşen bir durumdur. config(8) tarafından bu-

```
# $NetBSD: GENERIC,v 1.231.4.2 2009/02/19 20:23:46 snj Exp $
#
# GENERIC machine description file
#
# This machine description file is used to generate the default NetBSD
# kernel.  The generic kernel does not include all options, subsystems
# and device drivers, but should be useful for most applications.
#
# The machine description file can be customised for your specific
# machine to reduce the kernel size and improve its performance.
#
# For further information on compiling NetBSD kernels, see the config(8)
# man page.
#
# For further information on hardware support for this architecture, see
# the intro(4) man page.  For further information about kernel options
# for this architecture, see the options(4) man page.  For an explanation
# of each device driver in this file see the section 4 man page for the
# device.

include "arch/amd64/conf/std.amd64"

options         INCLUDE_CONFIG_FILE     # embed config file in kernel binary
```

Resim 1 - generic yapılandırma

lunamamış olan hatalar veya bazen de donanımdan kaynaklı sorunlar, derleme işlemi donanım üzerinde diğer uygulamalardan daha fazla yük oluşturduğu için RAM de olabilecek hatalar make(1) işleminin sonlandırmasına neden olabilmektedir. Bazı durumlarda da bazı seçeneklerin diğer seçeneklere bağımlı olması nedeni ile derleme işleminde eksik bir dosyadan kaynaklanan hatalar ortaya çıkıp işlem yarıda kesilebilmektedir. Bazen de derleme işlemi donanıma bağlı olarak uzayabilir. Sistemin askıda kaldığını düşünüp bilgisayarı yeniden başlatmaya çalışmayın.

Başarılı bir derleme işleminin ardından netbsd adlı bir dosya compile dizininde oluşacaktır. Bu derlenmiş olan NetBSD çekirdeğidir. Eski çekirdeğin yedeğini alıp yeni çekirdeği / dizinine kopyaladıktan sonra sistemi yeni çekirdek ile başlatabilirsiniz.

```
# mv /netbsd /netbsd.generic
# mv netbsd /
```

Sistemi yeniden başlatarak yeni çekirdeği kullanabilirsiniz.

```
# shutdown -r now
```

İşler Ters Giderse...

Derleme işleminin başarıyla sonuçlanması derlediğiniz çekirdeğin işe yarayacağı anlamına gelmez. Yapılandımanız doğru olsa bile kaldırmış olduğunuz bir bileşen veya tanımladığınız çekirdek parametrelerinin hatalı olması ve başka nedenlerden dolayı yeni çekirdek ile sistemi başlatamayabilirsiniz. Bu durumda sistemi tek kullanıcı -single user- kipinde başlatıp eski çekirdeği seçerek sistemi başlatabilirsiniz.

```
> boot netbsd.old -s
```

Bu işlemin ardından aşağıdaki komutları kullanarak eski çekirdeği geri yükleyip sisteminizi açabilirsiniz.

```
# fsck /
# mount /
# mv netbsd.old netbsd
# reboot
```

Eski çekirdek ile sistemi başlatıp kendi derlediğiniz çekirdek ile karşılaştırarak sorunu belirleyebilir ve çekirdek yapılandırmanızı yeniden düzenleyerek tekrar deneyebilirsiniz.

OpenBSD'de Çekirdek Derlemek

OpenBSD geliştiricileri ve kullanıcıları çekirdeğin derlenmesine veya bütün sistemin sıfırdan kaynak kodundan derlenmesine pek sıcak bakmazlar. OpenBSD geliştirme ve inşa sürecinin diğer işletim sistemlerine göre farklı olması nedeniyle birçok kullanıcı OpenBSD'de çekirdek derleme ve/veya sistemi kaynak koddan derleme sürecini alışkanlıkları doğrultusunda yaptıklarında ellerindeki sistemin beklemedikleri gibi olmadığını görüp diğer OpenBSD topluluğuna başvurmaktadır. Bu başvuruya alınan yanıt pek de hoş olmamaktadır. OpenBSD geliştiricileri ve topluluğu sisteminizin hem sistem hem de kullanıcı tarafının kaynak koddan derlenmesine sıcak bakmadıklarını nazik bir dille dile getirecektir. Sık Sorulan Sorular – FAQ bakacak olursanız gerekçelerinini görebilirsiniz.

OpenBSD çekirdeğini yeniden derleme konusunda kararlıysanız sisteminizde kaynak kodlar bulunmalıdır. Kaynak kodlar OpenBSD'de iki dosya olarak sunulur. Birinci dosya çekirdek sistem- core system- ve ikinci dosya da kernel -çekirdek- olarak adlandırılır. Çekirdek sistem dosyası 128 Mb ve çekirdek dosyası da 20 Mb büyüklüğündedir. Çekirdek sistem ve çekirdek dosyalarını OpenBSD ftp sunucusundan [2],[3] edinebilirsiniz. Eğer X sunucusu -Xenocara (103 MB)- ve port ağacını (17 Mb) da isterseniz yine aynı şekilde bunları da ftp sunucusundan indirebilirsiniz. [4], [5]

İndirme işleminin ardından kaynak kodları barındıran dosyaların sırasıyla /usr/src dizinine açılması gerekir. Src.tar.gz ve sys.tar.gaz dosyaları /usr/src ve xenocara.tar.gz ile ports.tar.gz dosyaları da /usr dizinine açılmalıdır.

```
# mv src.tar.gz /usr/src
# cd /usr/src
# tar -xvzf src.tar.gz
# mv sys.tar.gz /usr/src
# cd /usr/src
# tar -xvzf sys.tar.gz
# mv ports.tar.gz /usr
# cd /usr
# tar -xvzf ports.tar.gz
# mv xenocara.tar.gz /usr
# cd /usr
# tar -xvzf xenocara.tar.gz
```

Dosyaları açtıktan sonra OpenBSD geliştirme ekibi tarafından yayınlanan yamaları indirip kaynak kodlara bu yamaların yapılması gereklidir. OpenBSD için hazırlanan yamaları ister cvs(1) isterseniz

tar.gz dosyası olarak indirebilirsiniz. Yamalar OpenBSD sunucularından [6] edinilebilir. Yamalar ile ilgili bilgiyi de OpenBSD sitesinden edinebilirsiniz. [7]

Yama dosyasını indirdiğinizde /usr/src dizini altında patches adlı bir dizin oluşturup dosyayı buraya kopyalayıp ardından da sırayla yamaları çekirdek kaynak koduna uygulamanız gerekir. Eğer daha önceden çekirdeğe yaması yaptıysanız bu durumda sadece son yamayı veya yamaları uygulamanız yeterli olacaktır.

```
# mkdir /usr/src/patches
# mv 4.7.tar.gz /usr/src/patches
# cd /usr/src/patches
# tar -xvzf 4.7.tar.gz
```

Dizinin içeriğine göz atacak olursanız OpenBSD'nin aktarıldığı platformlara ait dizinler görebilirsiniz. Bu dizinlerin altında her hangi bir dosya bulunmaz. Yamalar toplu halde common dizininin altında bulunmaktadır. Yamaları uygulamadan önce yamaların çekirdek veya diğer bileşenlerden birisine ait olup olmadıklarını kontrol etmeniz gereklidir. Örneğin OpenBSD 4.7 için yayınlanan yamalardan sadece 002_mpi.patch, 004_pfsync.patch ile 005_pfsync.patch çekirdek yamalarıdır. Diğerleri OpenBSD ile gelen çeşitli kütüphanelere ait olan yamalardır.

Yamaları kontrol etmek için head kullanabilirsiniz. Döndürülecek olan mesaj hem yama hem de yamanın nasıl uygulanacağı hakkında bilgi vermektedir.

```
# head -7 004_pfsync.patch
This is the 2nd revision of this patch.

Apply by doing:
        cd /usr/src
        patch -p0 < 004_pfsync.patch
```

```
-rw-r--r--  1 root  wheel   86155045 Feb  6 19:28 gnusrc.tgz
-rw-r--r--  1 root  wheel    7266679 Feb  6 19:28 sharesrc.tgz
-rw-r--r--  1 root  wheel   60638706 Feb  6 19:26 src.tgz
-rw-r--r--  1 root  wheel   37340637 Feb  6 19:28 syssrc.tgz
-rw-r--r--  1 root  wheel  139243356 Feb  6 19:31 xsrc.tgz
# pwd
/root/indirme
# cd ../
# ls
.cshrc     .lesshst   .profile  MD5        indirme    test
.klogin    .login     .shrc      SHA512     md5sum     usr
# cd ./test/
/root/test
# ls
gnusrc.tgz   sharesrc.tgz src.tgz      syssrc.tgz   xsrc.tgz
# ls =l
ls: =l: No such file or directory
# ls -l
total 323104
-rw-r--r--  1 root  wheel   86155045 Feb  6 19:28 gnusrc.tgz
-rw-r--r--  1 root  wheel    7266679 Feb  6 19:28 sharesrc.tgz
-rw-r--r--  1 root  wheel   60638706 Feb  6 19:26 src.tgz
-rw-r--r--  1 root  wheel   37340637 Feb  6 19:28 syssrc.tgz
-rw-r--r--  1 root  wheel  139243356 Feb  6 19:31 xsrc.tgz
#
```

Resim 2 - /usr/src dizini

Çekirdeği derlemeden önce tüm çekirdek yamalarının uygulanması zorunludur. Ancak yamaların uygulanmasından sonra çekirdeği derleyebilirsiniz. Derleme işlemine başlamadan önce çalışır durumdaki çekirdeğin yedeğini almanız gereklidir.

```
# cp /bsd /bsd.old
```

Çekirdeğin bir yedeğini aldıktan sonra çekirdek yapılandırmasını düzenlemeniz gerekli. Eğer çekirdek yapılandırmasını değiştirmeyecek iseniz ve standart yapılandırma ile çekirdeğinizi derleyecek iseniz aşağıdaki işlemleri uygulamak yeterli olacaktır. OpenBSD geliştiricileri çekirdeğin fazlasıyla donanıma bağlı olan yapısı gereği farklı mimariler için standart çekirdek yapılandırma dosyaları hazırlamışlardır. Bunlar içinden kullanılan mimariye uygun olanını seçerseniz derleme işleminin sorunsuz ve desteklenen bir çekirdek yapılandırması ile sorunsuz olarak tamamlanmasını garantilemiş olursunuz.

```
# cd /usr/src/sys/arch/<mimari>/conf/
```

Kullandığım sistem amd64 olduğu için

```
# cd /usr/src/sys/arch/amd64/conf
# config GENERIC
```

config GENERIC komutunun tamamlanmasının ardından yeni çekirdeğin derlenmesi aşamasına geçilebilir.

```
# cd ../compile/GENERIC
# make
# make depend
# make install
```

Bu işlemler tamamlandığında yeni çekirdek kurularak sistemi yeniden başlattığınızda kullanılabilir. Eğer yeni derlediğiniz çekirdek ile sistemi başlatamazsanız eski çekirdeği kullanarak sistemi yeniden başlatabilirsiniz.

```
Using drive 0, partition 3.
Loading...
probing : pc0 com0 apm mem[634K 319M a20=on]
disk: fd0 hd0+
>> OpenBSD/i386 BOOT 3.01
boot>
```

Yeni çekirdek yerine eski çekirdek ile sistemi başlatmak için bsd.old yazmanız yeterlidir.

```
Using drive 0, partition 3.
Loading...
probing : pc0 com0 apm mem[634K 319M a20=on]
disk: fd0 hd0+
>> OpenBSD/i386 BOOT 3.01
boot> bsd.old
```

Bu durumda eski çekirdek ile sistemi açabilir ve yeni çekirdeğin derlenmesi sırasında oluşah hatayı gidermek için işlemi yeniden gerçekleştirebilirsiniz.

[1] ftp://ftp.netbsd.org/pub/NetBSD/iso/<sürüm>/sourcecd.iso
[2] ftp://ftp.openbsd.org/pub/OpenBSD/4.7/src.tar.gz
[3] ftp://ftp.openbsd.org/pub/OpenBSD/4.7/sys.tar.gz
[4] ftp://ftp.openbsd.org/pub/OpenBSD/4.7/ports.tar.gz
[5] ftp://ftp.openbsd.org/pub/OpenBSD/4.7/xenocara.tar.gz
[6] ftp://ftp.openbsd.org/pub/OpenBSD/patches/
[7] http://www.openbsd.org/errata.html

SUDO

Hak ve yetkilerin farklı yönetimi

Gökşin Akdeniz
goksin@enixma.org

Sistem yöneticisi hesabının ancak ve ancak zorunlu durumlarda kullanılması sistem yöneticilerinin temel prensiplerindendir. Bu nedenle yönetimsel faaliyetler kullanıcı hesapları veya grupları tarafından gerçekleştirilecek şekilde yapılandırılır. Böylelikle sistem yöneticisi hesabının şifresinin diğer kullanıcılar ile paylaşılmasının önüne geçilmiş olur.

Kullanıcı hesapları ve gruplarının yönetilmesi dışında bir diğer seçenek ise sudo kullanmaktır. Sudo benzer olarak belirli kullanıcı ve gruplara root yetkileri gerektiren bazı eylemleri normal kullanıcı hesapları ile sisteme erişip gerçekleştirmelerine olanak verir. Bunu yaparken kullanıcılara sadece belirtilen komutları root yetkisi ile belirli bir süre için çalıştırma izni verilir. Buna ek olarak gerçekleştirilen her işlem syslogd(8) -system log daemon- ile kayıt altına alınabilir. Kullanıcıların verdikleri komutlar sonradan kontrol edilebilir. Bunlara ek olarak sudo birçok platformda kullanılabilir. Hatta merkezi bir yapılandırma dosyası kullanarak farklı sistemlerdeki kullanıcılar kolaylıkla yönetilebilir.

Sudo Tarihçesi

Sudo ilk olarak 1980'lerde Bob Coggeshall ve Cliff Spencer tarafından geliştirilip kullanılmıştır. İlk uygulama üzerinde 4.1 BSD çalışan bir VAX-11/750'de gerçekleştirilmiştir. Bunu izleyen geliştirme sürecinde bir sonraki sürüm Phil Betchel, Cliff Spencer, Gretchen Phillips, John LoVerso ve Don Gworek tarafından geliştirilip USENET üzerinde 1985 yılının Aralık ayında duyurulup dağıtılmıştır. 1986 yılı yazında Garth Snyder iyileştirmeler yaparak daha gelişmiş özelliklere sahip olan yeni bir sürüm hazırlamıştır. Bu sürüm Bob Coggeshall, Bob Manchek ve Trent Hein'ın da katkıları ile izleyen yıllarda geliştirilmiştir. 1991 yılında Dave Hieb ve Jeff Nieusma, sudo yapılandırma dosyasının bu günkü halini aldığı yeni bir sürümünü "The Root Group" adlı bir danışmanlık firmasının siparişi üzerine geliştirmiştir. Bu sürüm daha sonra Genel kamu Lisansı ile dağıtılmaya başlanmıştır.

1994 yılına dek Bob Coggeshall, Bob Manchek ve Trent Hein tarafından geliştirilmeye devam edilen sudo, Todd Miller tarafından geliştirilen ve The Root Group için geliştirilen sürümden ayrılması için "CU sudo" adı verilen 1.3 sürümü ile devam etmiştir. Bu sürüm farklı işletim sistemlerini desteklerken önceki sürümlerde bulunan hatalar için geliştirilen yamaları da kapsayan sürümdür.

1995 yılında sudoers dosyası için kullanılan ayrıştırıcının yeni sürümü Chris Jepay tarafından geliştirilmiştir. Böylelikle sudoers dosyası sudo ve visudo tarafından doğru olarak yorumlanabilmiştir.

1999 yılında ise The Root Group tarafından 1991 yılından o güne dek geliştirmedikleri için "CU-sudo"da yer alan "CU" ön eki kaldırılmıştır. 1.6 sürümünden itibaren ise "The Root group"a ait kodlar kullanılmamıştır. Bu sürümden itibaren "ISC-tarzı" bir lisans ile dağıtılmaya başlanmıştır.

2001 yılında ise sudo ve ilgili e-posta listeleri, web sitesi vb bugün kullanılmakta olan sudo.ws sitesine taşınmıştır. Sudo.org'un neden kullanılmadığını merak edenler için hemen belirtelim; sudo.org alan adının bir başkası tarafından alınmış olması nedeni kullanılmamaktadır.

2005 yılında suoders ayrıştırıcıs Todd Miller tarafından yeniden yazılarak bugünkü halini almıştur. Todd Miller halen aktif olarak OpenBSD ve sudo geliştiricisidir. Kendisine Todd.Miller@courtesan .com e-posta adresinden ulaşılabilir.

Neden Sudo?

UNIX sistemlerde farklı servislerin kullanılması ve yönetimi söz konusu olduğunda iki kullanıcı bu servisler üzerinde kontrol sahibi olabilir. Birincisi root ve diğeri de söz konusu servise ait olan kullanıcı hesabı ve grubudur. BSD sistemleri ele aldığım yazı dizisinde kullanıcı hesapları ve gruplarının yapılandırılmasını ele almış ve farklı servisleri kullanmak ve yönetmek için kullanıcı hesaplarının nasıl yapılandırılacağını ve kullanılacağını göstermiştim. Bunun dışında bir diğer yöntem de 1980'lerde Bob Coggeshall ve Cliff Spencer tarafından geliştirilip kullanılmaya başlanan sudo kullanmaktır. İlk uygulama üzerinde 4.1 BSD çalışan bir VAX-11/750 üzerinde geliştirilen bugüne hem ticari hem de ticari olmayan UNIX sistemler ile POSIX uyumlu UNIX türevlerinde kullanılan sudo kullanıcı hesapları ve grupları ile gerçekleştirilebilen yönetsel işlemlerin komut, sunucu ve kullanıcı bazında tanımlanarak kullanıcıların geçici olarak root yetkileri ile söz konusu işlemleri gerçekleştirmelerini veya komutları çalıştırmalarını sağlar.

Sistemdeki kullanıcı hesaplarını ve grupları doğru biçimde yapılandırdığınızda normal kullanıcı hesapları kullanılarak birçok sistem servisi yönetilebilir ve root hesabının kullanılmasına gerek kalmaz. Ancak bazı durumlarda birden çok sistemi yönetmek gerekebilir. Bu durumda ise farklı sistemlerdeki kullanıcı hesapları ve gruplarının yönetimi işletim sistemlerinin farklı olması durumunda karmaşık bir hal alacaktır. Bu karmaşıklık Windows ve diğer sistemler olarak düşünülmemelidir. UNIX ve türevi sistemlerde de kullanıcı grupları ve hesapları aynen oluşturulmuş olsa da uid ve gid modelinin işleyişinin farklılıklar göstermesinden kaynaklanmaktadır. Bu durumda bir sistemde yaptığınız yapılandırma diğer sistemde farklılıklar gösterecek sistem yöneticisinin farklı çözümler aramasını ve bunları belgelendirmesini gerektirecektir. Bu karmaşıklığın önüne geçebilmek ve uid ve gid modelinin işleyişinin farklılıklarının neden olduğu sorunların aşılması için sudo kullanılabilir. sudo farklı UNIX ve türevi sistemlerde kullanılabilir. Sistemlerin yapısının farklı olmasına rağmen kullanıcıların gereken yetkileri gerektiği kadar kullanabilmeleri sağlar.

Sudo, root veya farklı bir kullanıcı olarak çalıştırılması gereken bir veya daha çok sayıdaki komutun, programların söz konusu kullanıcı tarafından çağrılarak çalıştırılmasını sağlar. Bunu yaparken de setuid mekanizmasını kullanır.

Aşağıda FreeBSD sistemdeki sudo görülüyor.

```
[goksin@tardis /usr/home/goksin]$ ls -l /usr/local/bin/sudo
---s--x--x  2 root  wheel  143824  6 Haz 21:57
/usr/local/bin/sudo
[goksin@tardis /usr/home/goksin]$
```

Aşağıdaki dosya özelliklerinde ise Debian dağıtımındaki sudo yapılandırmasını görüyoruz:

```
goksin@debian:~$ ls -l /usr/bin/sudo
-rwsr-xr-x 2 root root 127432 Haz 11 19:06 /usr/bin/sudo
goksin@debian:~$
```

Sudo kurulum için Linux dağıtımlarında depolarda yer almaktadır ve paket yönetim aracını kullanarak kurulabilir. BSD sistemlerde ports veya pkgsrc ile kurulumu gerçekleşecektir. Eğer kullanmakta olduğunuz sistem için hazırlanmış olan bir ikili dosyası bulunmuyorsa kaynak kodu indirilip derlenerek kullanılabilir. Ancak yapılandırma seçenekleri derleme işlemine başlanmadan önce kontrol edilmeli ve gereksinimler dikkate alınarak yapılandırılıp sonra derlenerek kurulmalıdır. Sudo kılavuzu iki parçalıdır. Birincisi sudo(8)'nun temel işleyişini tanımlar ve ikincisi de sudo yapılandırma dosyası olan sudoers(5) kılavuz sayfasıdır.

Sudo kurulduktan sonra sistem yöneticisi herhangi bir kullanıcının sistemde tanımlı bir başka kullanıcı hesabını kullanarak izin verdiği komutları ve programları kullanıcıların çalıştırmasına izin verebilir. Sudo'nun bu esnek yapısından dolayı yeni kullanıcı ve sistem yöneticileri tarafından kılavuzları ve belgeleri kavranması güç, anlaşılmaz olarak algılanmaktadır, ama tersine olarak basit, esnek ve son derece güçlüdür. Sudo doğru biçimde yapılandırıldığında diğer kullanıcılar ile root şifresinin paylaşılmasına gerek yoktur. Burada doğru yapılandırmadan kasıt gerekli yetkilerin gerektiği kadarının gereken kullanıcıya tanınmasıdır. Ayrıca sudo yapılandırma dosyası bir defa oluşturulduktan sonra aynı yapılandırmaya sahip birden çok sistemde kullanılarak merkezi bir yetkilendirme sistematiği oluşturulup merkezi olarak kullanılmasını ve denetlenmesini sağlar. Root şifresini bilen kullanıcı sayısının bir veya iki kişiyle sınırlanması da sistem yönetimi konusunda sistem yöneticilerinin üzerindeki yükü önemli ölçüde hafifletir. Sudonun esnekliği ve ince ayar yapılabilirliği dışında bir diğer önemli özelliği de çalıştırılan her komutun kaydının tutulmasıdır. Böylelikle sonradan her bir sudo komutunun kim tarafından, ne zaman verildiği ve ne yapıldığı kolaylıkla incelenebilir kılmasıdır.

Sudonun bu olumlu yanlarına karşın olumsuz tarafı kullanıcılar ve genç(!) sistem yöneticilerinin burun kıvırması, beğenmemeleridir. Genel olarak sistemde root yetkilerine sahip kullanıcılar/sistem yöneticilerinin elinden root şifresini alır ve root yetkilerini kullanmalarını önlerseniz, bu onların ellerindekini kaybettikleri şeklinde algılanmaktadır. Aslında halen günlük olarak yaptıkları işlemleri sudo ile yapmaya devem edecekleri için herhangi bir kayıpları söz konusu olmamaktadır. Kullanıcılar ve sistem yöneticileri root şifresini ve hesabını kullanmakta ısrar edebilirler. Bu durumda sorumlulukların kimin tarafından yerine getirildiğini kesin bir dille belirtmeniz ve bu ısrarların yersiz olduğunu kesin olarak belirtmeniz zorunlu olacaktır. Bu davranışlar kullanıcıların veya sistem yöneticilerinin sorumlulukları arasında olamayan işleri yerine getirip size sorun çıkarmaktan başka bir işe yaramayacaktır. Bunu kesin olarak belirtip çizgiyi çekmenizde yarar var.

Sudo birçok açıdan root hesabının kullanılmasını asgariye indirip güvenlik açısından önemli yararlar sağlamakta olsa da bazı durumlarda sistem yöneticisi tarafından hazırlanan yapılandırma dosyasındaki hatalar nedeni ile tersine olarak ciddi güvenlik riskleri yaratabilir. Bu risk bir kullanıcının veya sistem yöneticisinin hatalı yapılandırma sonucu root olarak her komutu çalıştırabilir duruma gelmesi demek olabilir. Bu nedenle sudo yapılandırmasını hazırlarken güvenlik ve sistem yönetimi politikalarınızı göz önünde bulundurmalısınız.

sudo(8), sudoers(5) ve visudo(8)

Sudo üç farklı bileşenden oluşur. Birincisi asıl kullandığınız sudo(8) komutu, sudo yapılandırma dosyası olan sudoers(5) dosyası ile sudoers dosyasını düzenlemekte kullandığınız visudo'dur. Sudo(8) aslında setuid mekanizmasını kullanan ve root veya diğer bir kullanıcı hesabını geçici olarak kullanmanıza olanak veren uygulamadır.

sudoers(5) dosyası ise sudo'nun nasıl davranacağını ve kullanıcıların hangi komutları, programları hangi farklı kullanıcı olarak çağırabileceğini belirten yapılandırma dosyasıdır. visudo(8) ise sudoers dosyasını düzenlemek için kullanacağınız ve yapılandırmanızı olası hatalara karşı kontrol etmenizi sağlayan görsel sudoers düzenleyicisi, bir diğer deyişle tercih ettiğiniz kabuk ortamında çalışan metin düzenleyicisidir. Visudo ile sudoers dosyasınızı düzenlemek ve kontrol etmek son derece kolay olsa da, visudo kullanmak için sudo'ya başvuruyorsanız bu durumda önemli bir risk ile karşı karşıya kalabilirsiniz. sudoers dosyasında yapacağınız bir hata elinizi kolunuzu bağlayacak ve visudo kullanamayacağınız gibi sudo da beklendiği gibi çalışmayacaktır. Bu riskli durumu önlemek için sudo kendi güvenlik mekanizmaları barındırır.

vipw(8) gibi visudo(8) da dosyayı diğer bir kullanıcının erişimine karşı kilitler ve dosyayı sadece bir kişi düzenleyebilir. visdudo(8) için varsayılan metin düzenleyicisi vi(1)'dır ama $EDITOR değişkeninde tanımlanan editörü çalıştıracaktır. BSD sistemlerde aksine bir düzenleme yapmadıysanız vi(1) çalışacaktır. sudoers(5) dosyasında yapacağınız düzenlemeleri bitirdiğinizde visudo(8) dosyadaki hataları kontrol eder. Bir hata bulunursa sizi uyarır. Eğer hata almadıysanız bu, dosyayı gereksinmelerinize, sistem yönetimi ile güvenlik politikalarınıza göre gerektiği gibi düzenlediğiniz anlamına gelmez; sadece imla hatası yapmadığınız anlamına gelir. Hata bulunması durumunda visudo hatanın bulunduğu satırı belirten bir mesaj döndürerek sizi uyaracaktır.

```
tardis# visudo
>>> sudoers file: syntax error, line 11 <<<
What now?
```

Burada visudo(8) dosyada hata bulduğu için dosyanızı kayıt etmeyecektir. Bu durumda "e" tuluna basarak geri dönüp hatalı olan satırı düzeltmeye çalışabilirsiniz. "x" tuşuna basarak hatalı olan yapılandırmayı göz ardı ederek yaptığınız değişikliklerin silinmesini ve asıl dosyasının olduğu gibi kalmasını tercih edebilirsiniz. Bu durumda çalışır durumdaki eski dosyanız geçerlidir ve kullanılır durumdadır. Son seçeneğiniz olarak ise her şeyi göze alıp "Q" tuşuna basar ve hatlı yapılandırmanız kayıt edilerek kullanılmasını ve geçerli kılınmasını sağlayabilirsiniz. Bu durumda sudo(8) doğru biçimde çalışmayacaktır. Hatayı ancak root olarak sisteme giriş yaparak düzeltebilirsiniz. Bunu asla yapmayın!

sudoers(5) dosyası OpenBSD'de ve Linux dağıtımlarında /etc /sudoers, FreeBSD'de ise /usr/local/etc/sudoers olarak yer alır. Sudoers dosyasını düzenlemek için herhangi bir metin düzenleyici kullanmayın. Bunun yerine visudo(8) kullananın. Çünkü sudo(8) kullanılan yetkilendirme tanımlama söz dizimi, sudo ile daha önce çalışmamış birisi için zor gelecektir. Sudo(8) işleyişini kavradığınızda söz dizimi de daha anlaşılır gelecektir. Ancak her ne kadar iyi kavramış olsanız da visudo(8) kullanmak hataları - sadece söz dizimi açısından - sıfıra indirgeyecektir.

İnternette birçok sitede örnek sudoers(5) dosyası bulabilirsiniz. Bu dosyalar genelde sudo sitesindeki yapılandırma dosyasının kopyalarıdır veya belirli bir sistem için hazırlanmış olan yapılandırmaya aittir. Bazıları ise sudo(8) ile yapabileceği her şeyi göstermek için hazırlanmış olan dosyalardır. Eğer sudo(8) ile ilk defa ayrıntılı olarak ilgileniyorsanız bunların size bir yararı olmayacaktır.

Sudoers(5) dosyasında kullanılan söz dizimi aşağıdaki yapıyı kullanmaktadır.

```
kullanıcı_adı           sistemin_adı = komut
```

kullanıcı_adı, komutu çalıştıracak olan kişinin sistemde tanımlı olan kullanıcı adıdır. sistemin_adı ise sisteminizin adı yani sisteminizi tanımlayan /etc/hosts dosyasındaki girdidir. sudo(8) tasarlanırken bu durum dikkate alındığı için tek bir sudo dosyası yazarak bunu farklı sistemlerde kullanabilirsiniz.

Komut kısmında ise tanımlanan sistemdeki tanımlı kullanıcının çalıştırabileceği komut belirtilir. Komutu tanımlarken komutun tam yolunu ve seçeneklerini kesin olarak tanımlamanız gereklidir. Aksi durumda işe yaramayacak ve komut çalışmayacaktır. Bunu geliştiricilerin hatası olarak düşünebilirsiniz, ama kullanıcıların $PATH değişkenini kendi istedikleri gibi değiştirip istedikleri komutları ve programları çalıştırmalarını önleyecektir.

sudo(8) yapılandırmasında varsayılan eylem herhangi bir komutun çalıştırılmamasıdır. Bir kullanıcının bir komutu çalıştırmasına izin vereceksenız söz konusu kullanıcının söz konusu sistemde söz konusu komutu çalıştırabileceğini belirtmeniz gerekir. Bu sudoers(5) dosyasında belirtilen yapılandırmadır. Eğer sudoers(5) dosyasında tanımlanan kullanıcı, sunucu ve komuttan herhangi biri eşleşmez ise işlem gerçekleşemeyecektir.

ALL terimini kullanarak tüm sistemlerde ve girilen her komutun çalıştırılmasını tanımlayabilirsiniz. Örneğin kendi sistemimde goksin kullanıcısının root olarak herhangi bir komutu çalıştırmasını istersem ALL kullanarak bunu yapabilirim.

```
goksin            ALL = ALL
```

Birden çok sistemi yönetmek durumunda iseniz bu durumda kullanıcının tüm sistemlerde aynı görevi yerine getirmekte olduğunu varsayalım. Bu durumda kullanıcının tüm sistemlerde -ALL- örneğin web sunucusunu yönetmesi için gereken grup ve kullanıcı izinlerini yapılandırmak dışında web sunucusunun başaltılması, durdurulması vb işlemler için root yetkilerine gereksinim olacaktır. Söz konusu Apache web sunucu servislerini yöneten kullanıcı hesabının sudoers(5) dosyasındaki girdisi aşağıdaki gibi olacaktır.

```
goksin            ALL = /usr/local/sbin/apachectl start, \
                  /usr/local/sbin/apachectl stop, \
                  /usr/local/sbin/apachectl restart, \
                  /usr/local/sbin/apachectl graceful
```

sudoers(5) dosyasında çalıştırılacak olan komutları teker teker tanımlamak yerine her birini bir girdide de tanımlayabiliriz. Benzer olarak tüm sistemleri ALL ile tanımlamak yerine isimlerini kullanarak bu sistemlerden bazılarında geçerli olmasını sağlayabiliriz. Örneğin sadece web sisteminde komutlarının çalıştırılmasını izin verecek olsam bu durumda ALL yerine web yazmak yeterli olacaktır. Aynı zamanda aynı kullanıcının farklı bir sistemde tüm yetkileri kullanmasına izin veriyorsam bu durumda aynı sudoers(5) dosyasına diğer sistemlerin adlarını ve kullanıcıya verilen yetkileri tanımlamak yeterli olacaktır.

```
goksin            dns = /usr/sbin/ndc
goksin            web = /usr/local/sbin/apachectl start, \
                  /usr/local/sbin/apachectl stop, \
                  /usr/local/sbin/apachectl restart, \
                  /usr/local/sbin/apachectl graceful
goksin            tardis = ALL
```

suoders(5) dosyasının sadece kullanıldığı sisteme özel hazırlanmış olması gerekmez. Örnekte de görüldüğü gibi aynı dosya farklı sistemlerde farklı komutlar ve farklı kullanıcılar için tek bir dosya olarak hazırlanıp tüm sistemlerde aynı dosya kullanılabilir. Örneğin root olarak değil de farklı bir kullanıcı olarak farklı bir sistemde sudo ile farklı bir komut çalıştırmak için aşağıdaki örnekteki gibi kullanıcı adı ve komut tanımlanabilir.

```
goksin            droideka = (viceroy) pfctl -e, \
                             pfctl -d, \
                             pfctl -F all -f /etc/pf.conf
                             pfctl -s [ rules | nat | state ]
                             pfctl -vnf /etc/pf.conf
```

Bu örnekte goksin kullanıcısı droideka adlı sistemde viceroy adlı kullanıcı olarak pfctl komutlarını çalıştırmaktadır.

Sudoers(5) dosyasının yapısı girdilerin tek satır olarak yazılmasını öngörmektedir. Yukarıdaki örneklerde ise komutlar tek satıra sığmadığı için "\" kullanılarak alttaki satırdan devam edilmiştir. "\" kullanırsanız komut alttaki satırdan devam ediyor demektir. Bu suoders(5) dosyasında en sık yapılan hatadır. Komut alttaki satırdan devam ederken bu satıra bazen yeni bir girdi yazılabilir. Bu durumda visudo(5) hata mesajı döndürecektir.

Sudo(8) Kullanımı

sudo(8) kullanmak için sudoers(5) dosyasının yapılandırılması yeterlidir. Kullanıcı hesabınızz gereken yetkileri visudo(8) ile tanıdığınızda sudo kullanmaya başlayabilirsiniz. İlk defa kullandığınızda sudo sizden şifre girmenizi isteyecektir. Girmeniz gereken şifre root şifresi değil sudo çalıştıran kullanıcının şifresidir. Şifreyi üç defa yanlış girerseniz sudo derlerken ALL INSULTS seçeneğini aktif kıldıysanız sudo komutu veren kullanıcı ve ataları hakkında yorum yapacak ve işlemi sonlandıracaktır. Verdiğiniz sudo komutunu yeniden girmeniz gerekecektir.

Şifrenizi doğru girdiyseniz komut çalıştırılacaktır. sudo(8) bu işlemi zaman kaydını da belirterek syslog dosyasına kayıt edecektir. Eğer sudo(8) en son işleminizin üzerinden 5 dk geçtikten sonra yeniden çalıştıracak olursanız sizden yeniden şifrenizi girmenizi isteyecektir. Şifre girildikten sonra sudo(8) aksi belirtilmedikçe sadece 5 dk için şifre girmenizi istemeyecektir. Bu süre geçtikten sonra yeniden şifrenizi girmeniz gerekecektir.

Bir sistemde normel kullanıcı hesabınız ile işlerinizi yapıyorsanız sudo ile hangi komutları çalıştıracağını öğrenmek istersiniz. Genelde sistem yöneticileri sudo(8) ile yetkilendirdikleri kullanıcıları ve çalıştırabilecekleri komutları belgelendirip kullanıcıya bildirirler. Bazı sistem yöneticileri bunu yapmaz. Ancak sisteme normal kullanıcı hesabı ekleyerek buna gerekli olan grup ve erişim izinleri verip sudo(8) ile bazı servisleri yönetmelerine izin verdiklerinde sudo(8) yapılandırmasını kontrol etmek isteyebilir. Her iki durumda da sudo(8) ile kullanıcı hesabının hangi komutları çalıştırabileceğini öğrenebilirsiniz. 'sema' kullanıcısı sudo(8) ile herhangi bir komut çalıştıramaz. 'goksin' kullanıcısı wheel grubundadır ve root olarak sudo(8) ile komut çalıştırabilir.

```
[sema@tardis /usr/home/sema]$ sudo -l
Password:
Sorry, user sema may not run sudo on tardis.
[sema@tardis /usr/home/sema]$
```

```
[goksin@tardis /usr/home/goksin]$ sudo -l
Password:
User goksin may run the following commands on this host:
    (ALL) ALL
[goksin@tardis /usr/home/goksin]$
```

sudo ile bir komut çalıştırmak için çalıştırılacak olan komutun başına sudo eklenmelidir. Örneğin sudo kullanarak root olmak için aşağıdaki komut girilmelidir.

```
[goksin@tardis /usr/home/goksin]$ sudo su
Password:
tardis#
```

Bu şekilde root hesabına girişe izin verilmesi durumunda yetkilendirilmiş kullanıcı root şifresini değiştirebilir. Bu da sorunlara yol açacaktır. Bu nedenle her türlü komutu çalıştırma yetkisi vereceğiniz kullanıcılar gerçekten güveneceğiniz kişiler olmalıdır.

Sudo (8) ile uzun ve karmaşık komutları kolaylıkla kullanabilirsiniz. Örneğin root hesabını kullanmaya gerek kalmadan log dosyalarını tail -f /log/dosyası olarak tanımlayabilir ve eş zamanlı olarak olan olayları log dosyası üzerinden izleyebilirsiniz.

sudo ile yapabileceğiniz bir diğer önemli işlem ise farklı bir kullanıcı

hesabını tanımlayarak o hesaba ait yetkileri kullanabilmenizdir. Bunun için komutu aşağıdaki gibi vermeniz gereklidir.

```
sudo -u <kullanıcı_adı> <komut>
```

Örneğin operator hesabının sistem yedeği almak için dump kullanması işlemi sudo ile aşağıda gördüğünüz gibi yapılacaktır.

```
[goksin@tardis /usr/home/goksin]$ sudo -u operator dump
/dev/ad4s1a
```

Sudoer(5) Yapılandırmasında Alias Kullanımı

sudo(8) yapılandırmasında birden çok komut ve birden çok kullanıcıyı tek bir dosyada tanımladığınızda bu kullanıcıların eklenmesi, çıkarılması veya çalıştırabilecekleri komutlarda gerçekleştirilecek olan değişikliklerin yapılması sudo yapılandırmasından sorumlu olan kişi veya kişiler için zor hale gelebilir. Bu nedenle sudo(8) geliştiricileri kabuk ortamında tanımlanan ve bir çok uzun komutun tek bir isimle tanımlanarak çalıştırılmasına olanak veren alias özelliğini de eklemişlerdir. Alias tanımlamaları hem sudo(8) yapılandırmasını ve yönetimini çok basitleştirmektedir.

Sudo(8) için alias, kullanıcılar, sistemler ve komutların bir grup altında toplanmasıdır. Alias özellikle de bir veya birkaç kullanıcının yetkilerinin, görevlerinin değiştirilmesi, yeni sistemlerin eklenmesi veya var olan sistemlerden bir veya birkaçının kaldırılması vb durumunda sudoers(5) dosyasında her bir kullanıcı, sistem ve komut için yapılacak olan düzenlemeleri kolaylaştırarak sudo(8) yapılandıran sistem yöneticilerinin işlerini çok kolaylaştırır.

sudoers(5) dosyasında bir alias tanımlanmadan önce içeriğinin ne olacağına karar vermeniz gereklidir. sudoers.(5) dosyasında alias tanımlamaları dosyanın baş tarafında yer alır. sudoers(5) dosyasında alias tanımlamaları kullanıcı – User_Alias, komut – Cmnd_Alias, sistem – Host_Alias ve farklı bir kullanıcı olarak komut çalıştırmaya olanak veren – Runas_Alias olarak dört adettir.

User_Alias tanımlaması kullanıcıları görevlerine göre gruplandırmak için kullanılır. Aşağıdaki örnekte web sunucusundan sorumlu sistem yöneticileri WEBADMIN grubunda sistemlerdeki tanımlı kullanıcı hesapları ile tanımlanmışlardır.

```
User_Alias  WEBADMIN = cem, can, cemil, cahit
```

Bu kullanıcıların sadece web sunucuları üzerinde çalışacaklarını bildiğimiz için yetki oldukları sistemleri Host_Alias ile tanımlayabiliriz.

```
Host_Alias  WEBSUNUCU = WEB1, WEB 2, WEB3
```

Host_Alias kısımında sistemleri tanımlarken sistem adları dışında bu sistemlerin ip adreslerini de kullanabiliriz. WEB1, WEB2 ve WEB3 adları yerine ip adresleri kullanılarak yazarsak yukarıdaki kısım şu şekilde yazılabilir:

```
Host_Alias  WEBSUNUCU = 192.168.1.8, 192.168.1.9, 192.168.1.10
```

Bu yapılandırma şekli ssistem ip adreslerini teker teker yazmak yerine blok halinde tanımlamak da olanaklıdır. Örneğin kurumsal bir ağ üzerindeki tüm bilgisyarları teker teker adreslemek yerine bunları tek bir blok olarak tanımlayabiliriz.

```
Host_Alias  KURUMSAL = 192.168.1.0/16
```

Host_Alias, tanımlamalarında sistem isimlerinin kullanmak yerine ip adreslerini kullanmanız daha uygun olur. Çünkü DNS sorunları sudoers(5) dosyasındaki alias tanımlamalarının özellikle de sistem isimlerinin çözümlenememesi durumunda işe yaramayabilir. Ancak sisteminin hosts dosyasında sistem adı ile ip adresinin birlikte kullanılması çözümlemenin yerelden yapılmasını sağlayarak sorunu ortadan kaldırabilir. Ancak muhafazakar davranarak ip adreslerini

kullanmak sizi olası sorunlardan uzak tutacaktır.

Bu sistemlerde çalıştırabilecekleri komutları da Cmnd_Alias içerisinde benzer olarak tanımlayabilirsiniz. Örneğin bu sistemlerde yedek alma ve apache web sunucusunun yönetilmesi vb işlemleri aşağıdaki gibi tanımlayabiliriz.

```
Cmnd_Alias  YEDEKLE = /sbin/dump,/sbin/restore
Cmnd_Alias  WEBKOMUT = /usr/local/sbin/apachectl start, \
                       /usr/local/sbin/apachectl stop, \
                       /usr/local/sbin/apachectl restart, \
                       /usr/local/sbin/apachectl graceful
```

Bazı durumlarda ise çalıştırılması istenen komutlar, programlar özel bir kullanıcıyı gerektirir. Örneğin veritabanı uygulamaları kendilerine özel bir kullanıcı hesabını mysql, postgresql gibi sistemde tanımlı olmasını gerektirir. Sistemin yedeklemekten sorumlu olan bir kişi bu işlemleri operator kullanıcısı olarak gerçekleştirebilir. Bunun için de Runas_Alias kullanılır.

```
Runas_Alias OP = operator, root
```

Alias kullanabilmek için tanımlamalarını yaptıktan sonra kullanıcıların ve komutların listelendiği bölümlerde ilgili alias'ın kullanıldığı kuralların tanımlanması yeterlidir. Sudoers(5) dosyası sudo(8) tarafından okunurken önce alias kısmı okunduğu için sözkonusu kuralda hangi komutun, kullanıcının ve sistemin belirtildiği bellidir. Örneğin web sunucularını yöneten kullanıcı hesaplarını web sunucuları üzerinde belirtilen apache komutlarını çalıştırmak üzere aşağıdaki kuralı kulanabiliriz.

```
WEBADMIN    WEBSUNUCU = WEBKOMUT
```

Web sunucu yönetiminden sorumlu olan cem kullanıcısının yedekleme yetkilerine de sahip olmasını istersek ayrıca bir kural yazılmalıdır. Her ne kadar cem WEBADMIN grunbunda tanımlnamış olsa da yedekleme yetkisi yoktur.

```
cem         WEBSUNUCU = (OP) YEDEKLE
```

Bu satır ile cem kullanıcısı web sunucusu üzerinde yedekleme komutlarını kullanabilecektir.

suoder(5)'da Komut Kullanımını Yasaklamak

Bir kullanıcının bazı komutları kullanmasını önlemek isteyebilirsiniz. Zira bazen bir komuta izin verdiğinizde bir diğer komutun da çalıştırılmasına istemeden izin verebilirsiniz. Bunu önlemek için sudoers(5) dosyasında engellemek istediğiniz komutun başına ! Yazmanız gerekir ama yeterli olmayacaktır. Bunun nedeni internette bulabileceğiniz çeşitli suoders(59 dosyalarında Cmnd_Alias kısmında bazı komutların örneğin kabukların aşağıda görüldüğü gibi fahrettin kullanıcı hesabı ile kullanılmasının engellenmiş olduğunu görebilirsiniz.

```
Cmnd_Alias        SHELLS =/bin/sh,\
                          /bin/csh,\
                          /usr/local/bin/tcsh
Cmnd_Alias        SU = /usr/bin/su
fahrettin         ALL = ALL, !SHELLS, !SU
```

fahrettin kullanıcı hesabı su(1) komutunu çalıştırabilmektedir. Çalıştırdığı sistemde eğer ev dizini bulunuyorsa tercih ettiği bir kabuğu sisteme kurabilecei gibi kurulu olanı da kendi ev dizinine taşıyarak oradan çalıştırabilir.

```
tayfun# sudo sh
passwd:
Sorry, user fahrettin is not allowed to execute '/bin/sh' as
root on tayfun.
#
```

sudo(5) bir komutu çalıştırmak için tam yolu kullanır. Yukarıdaki

örnekte fahrettin kullanıcısı /bin/sh farklı bir konuma kopyalayarak veya bulunduğu dizini değiştirerek çalıştırabilir. Kopyalamak en kolay yol olduğu için fahrettin kullanıcısı /bin/sh dizinini kendi bulunduğu dizine kopyalayarak çalıştırabilir.

```
# id
uid=1000(goksin) gid=1000(goksin) groups=1000(goksin),
0(wheel)
# cp /bin/sh .
# sudo ./sh
# id
uid=0(root) gid=0(wheel) groups=0(wheel), 2(kmem), 3(sys),
4(tty), 5(operator), 20(staff), 31(guest)
#
```

Bu zayıflık sudo'nın nasıl işlediğini bilen herhangi bir kullanıcı tarafından kolaylıkla istismar edilebilir. Sudo kılavuz sayfasında bu duruma dikkat çekilmiş ve NOEXEC vb çözüm önerileri getirilmiştir. Ancak tercih edilmesi gereken kullanıcının gerekli yetkileri gerektiği kadar kullanmasının suoders.(5) dosyasının gerektiği gibi düzenlenerek sağlanmasıdır.

syslog(3) İle sudo(8) Kayıtları

sudo su komutu her kullanıcının root hesabını kullanmasına olanak verir. Bu durumda sudo ile her komutu çalıştırma yetkisi verdiğiniz kullanıcılar sudo kullanmak yerine sudo su kullanıp doğrudan root yetkilerini kullanacaktır. Bu durum sistemin işleyişi üzerinde ciddi sorunlara neden olacaktır. Bu nedenle sudo(8) çalıştırılan her komutu sudoers85) dosyasında belirtildiği gibi kaydını tutacaktır. sudoers(5) dosyasında aksi belirtilmediyse sudo(8) kayıtlarını, syslog(3) üzerinden, syslog.conf(5) dosyasında belirtilen LOCAL2 dosyasında saklayacaktır. Bu dosyaya bakacak olursanız tüm başarılı ve başarısız sudo eylemlerinin kaydını ve yapılan işlemleri tarih kayıtları ile görebilirsiniz.

SYSLOG

Sistemin kayıt tutucusu

Gökşin Akdeniz
goksin@enixma.org

Sistemin işleyişine ilişkin kayıtların tutulması sistem güvenliği, sistem yönetimi ve sorunların çözülmesi için kritik öneme sahiptir. Birçok durumda bu kayıtlar sorunların büyümeden önlem alınarak çözülmesini de sağlar. Hatta sistemdeki bazı problemlerin ne zaman ve nasıl başladığının belirlenebilmesini de sağlarlar.

Tüm UNIX ve türevi sistemler bir kayıt tutma mekanizmasına sahiptir. Solaris/ OpenSOLARIS'te, ticari UNIXler'de AIX, HP-UX vb ile Linux dağıtımlarında ilk olarak 4.3 BSD'de geliştirilen syslog(3) – system logger – programı kullanılmaktadır. syslog(3) yaygın olarak kullanılmasının ardında yatan neden, syslog(3)'un esnek ve güçlü yapısıdır. BSD işletim sistemi ailesi ile SOLARIS/OpenSOLARIS istemlerde syslog(3) standart olarak yer alırken Linux dağıtımlarında ise syslog yerine rsyslog tercih edilmektedir. Linux kullanıcıları isterlerse rsyslog yerine syslog'u depolardan kurup kullanabilirler. Kurulum sırasında syslog otomatik olarak yapılandırılmakta ve bazı dağıtımlarda ise syslog.conf dosyası da hazır olarak gelmektedir. Debian kullanıyorsanız rsyslog'un paket yönetim aracı tarafından broken olarak tanımlandığını ve yerine syslog kurduğunuzda tamamen hazır olarak geldiğini görebilirsiniz. BSD ile SOLARIS/OpenSOLARIS sistemlerde ise syslog.conf dosyası büyük ölçüde hazır olarak olarak gelmekte ve sistem yöneticisi tarafından gereken düzenlemeler yapılarak kolayca kullanılabilmektedir.

BSD syslog programı ilk olarak 4.3 BSD ile kullanılmıştır. İlk tasarlandığında tüm sisteme ait kayıtların tutulacağı merkezi bir kayıt uygulaması olarak tasarlanmıştır. Her ne kadar birçok uygulama kendi kayıt dosyalarını oluşturup bu dosyaları kullansa da hemen hemen bütün sistem için merkezi kayıt sistemi olarak bugün iş görmektedir.

Merkezi bir kayıt uygulaması olarak syslog hem çalıştığı hem de uzak sistemler de dahil olmak üzere syslog destekleyen donanımlara ait olan kayıtları da tutabilmektedir. Bunlar içerisinde Cisco router/yönlendiriciler ile switch vb donanımlar da bulunmaktadır.

Merkezi bir kayıt sisteminin yararı tüm sisteme ait bileşenler ve kayıt dosyalarının tek bir sistemde ve hatta tek bir dosyada tutulmasını sağlar. Bu kayıt işlemi bir ağ üzerinde yer alan bir çok bilgisayarın kayıtlarının ağ üzerinden merkezi bir sunucuya aktarılmasını sağlar. Syslog daemon – syslogd(8) ağ üzerinden uzak bir kayıt sunucusuna kayıtları aktarmak için gerekli olan alt yapıyı sunar. Böylelikle ağ üzerindeki sistemlerden birisinden veya bir kaçından kaynaklı sorunun bulunabilmesi için sistem yöneticisinin merkezi syslogd(8) sunucusunda ilgili dosyasını inceleyerek sorunun doğrudan kaynağını belirlemesi kolaylaşır.

Ağ üzerinden merkezi bir sunucuda kayıt tutmak ilk bakışta zor bir işlem gibi gelebilir. Özellikle de farklı sistemlerde çalışan süreçlerin aynı sistem kaynaklarını paylaşması, kayıtların uzak sisteme aktarılması ve uzak sistemde oluşacak yükün neden olacağı yükün kayıtların zaman damgalarında gecikmelere neden olacağı ve bunun da hassas olmayacağı, hatta yazma sırasında hatalar oluşabileceği düşüncesi sık olarak karşılaştığım bir durumdur. Aslında syslogd(8) bu sorunların üstesinden gelmek üzere tasarlanmıştır. Gelen kayıt isteklerini kayıt dosyasına aktarmak için gerektiğinde doğrudan ve gerektiğinde ara bellek olarak çalışarak dosyaya yazma işlemini sırayla gerçekleştirir. Böylelikle olabilecek olan hataların önüne geçilir. Ayrıca syslog(3) bu işlemleri gerçekleştirirken uzak ve yerel sistemlerdeki damon'ların kayıtlarını zaman damgası ile aldığı için kayıtların gerçek zamanlı olması da sağlanır. Böylelikle syslog(3) mükemmel bir sistem kayıt aracı olarak görevini yerine getirir.

4.3 BSD'den bugüne geçen zaman içerisinde syslog(3) tasarım olarak esnek ve ölçeklenebilir olduğu birçok defa kanıtlanmıştır. Geçen zaman dilimi dikkate alındığında syslog(3) gerek kod gerek işleyiş ve hatta yapılandırma olarak geçirdiği değişiklikler yok denecek kadar azdır.

syslog.conf(5) Yapılandırması

Syslog çalışması basittir. Programlar system logging daemon – syslogd(8) kayıt edilecek olan bilgiyi iletirler. syslogd(8), syslog.conf(5) yapılandırmasında belirtilen girdiler ile bu gelen istekleri karşılaştırır. Eğer gelen kayıt isteği ile yapılandırma eşleşiyorsa syslog(3) yapılandırmada belirtilen işlemi gerçekleştirir.

Yapılandırma dosyası olan /etc/syslog.conf'a bakıldığında iki sütundan oluştuğu görülür. İlk sütunda kayıt isteklerinin geldiği program veya sistem yer alır. İkinci sütunda ise gelen kayıt isteğinin nasıl, nereye kayıt edileceğine dair tanımlanmış olan kural belirtilir.

syslog.conf(5) dosyasındaki girdilerin düzenlenmesinde en sık karşılaşılan sorun hangi programın nasıl kaydının tutulacağıdır. Standart olarak kaynak ve derecelendirme yöntemi kullanılır. Kaynak kayıt isteğinde bulunan program veya sistem servisidir. Aşağıda syslog ile kullanılan kaynaklar görülmektedir.

auth	Kullanıcıların yetkilendirilmesi/tanınması işlemleri. Örneğin login ve su mesajları gibi
authpriv	auth ile aynı şekilde çalışır ancak yetkilendirilmiş kullanıcılar bu kayıtları okuyabilir.
console	Sistem konsoluna döndürülen mesajları belirtir.
cron	cron kayıtlarını belirtir.
daemon	Sistemde çalışır durumdaki tüm daemon mesajlarını belirtir.
ftp	FTP daemon gerçekleştirdiği tüm işlemlere ait mesajları belirtir.
kern	Kernel mesajlarını belirtir.
lpr	Yazdırma sisteminin mesajlarını belirtir.
mail	Mail sisteminin mesajlarını tanımlar.
mark	Bu aslında gerçek bir servis değildir. Sadece her 20 dk bir log dosyasına bir kayıt düşer. Birden çok log dosyası tutuyorsanız yararlı olacaktır.
news	Internet News daemons mesajlarını belirtir.

ntp	Network Time Protocol mesajlarını tanımlar.
security	Çeşitli güvenlik servislerinin örneğin, ipfw gibi mesajlarını kapsar.
syslog	syslog servisi de kendini kapsar. Bunu yapmayın, zira kayıt sayısının çokluğu sizi yıldıracaktır.
user	Bu aksi belirtilmediyse tüm mesajları kayıt eder. Herhangi bir kayıt dosyası veya kaynak tanımlamadığınızda bu kullanılır.
uucp	Unix-toUnix Copy protocol mesajlarını belirtir. Nostaljik değeri vardır.
local0 – 7	Bu değerler sistem yöneticisi tarafından kullanılmak üzere ayrılmıştır. Bazı uygulamalar bunları kullanacak şekilde hazırlanmışlardır. Örneğin sudo gibi

Birçok sistem syslog'a yollanan mesajların tamamını kayıt etmez. Sadece belirli önem derecelerine sahip olan mesajlar kayıt edilip diğerleri göz ardı edilir. Bu ayrım derecelendirme ile gerçekleşir. Bu derecelendirme önem sırasına göre aşağıdaki gibidir:

emerg	Acil. Tüm terminallerde yayınlanan mesajlar bu önem derecesindedir. Sistemin kararlılığını kaybettiği, çökmenin eşiğine geldiği durumlardır.
alert	Uyarı. Acil kadar şiddetli değildir, sistem çalışmaya devam edebilir ama acilen sistem yöneticisinin ilgilenmesi gereklidir.
crit	Kritik. Donanım ile bazı ciddi yazılım sorunları bu sınıfta yer alır. Diskinizde problemli bloklar bulunuyorsa bu mesajı almanız kesindir.
err	Çeşitli hatalar. Sisteme zarar vermez ama düzeltilmeleri gereklidir.
warning	Çeşitli uyarı mesajları
notice	Tarafınızdan ilgilenilmesi gerekmeyen ama kayıt edilmesinde yarar olan uyarı mesajlarıdır.
info	Genel olarak sisteme ilişkin bilgilendirme mesajlarıdır
debug	Bu düzey genel olarak geliştiriciler ve programcılar hata ayıklamak için kullanırlar. Bazen de sistem yöneticileri bazı uygulamaların çalışma sürecini izlemek için kullanırlar. Bu seviyedeki mesajlar içerisinde geliştiricinin önemli olarak düşündüğü her türlü bilgi yer alır. Bazen kullanıcılara ait bilgiler de yer alabilir.
none	Herhangi bir bilgi kayıt edilmez.

/etc/syslog.conf dosyasında ilk sütunda kayıt kaynağı ve derecelendirme aralarında bir nokta konularak tanımlanır ve sağ taraftaki sütunda da kayıt kaynağı belirtilir. Aşağıdaki örnek satır mail sisteminin info ve daha üst seviyedeki önem derecesine sahip mesajları /var/log/maillog dosyasına kayıt etmesini tanımlamaktadır.

```
mail.info          /var/log/maillog
```

Mail sisteminden gelen tüm mesajların kayıt edilmesi için ise * kullanılarak kural yeniden tanımlanmalıdır.

```
mail.*              /var/log/maillog
```

Sistemden gelen tüm mesajların tek bir dosyaya yazılmasını istiyorsanız aşağıdaki kuralı kullanabilirsiniz. Bu durumda dosyanın içeriği ile büyüklüğü doğru orantılı olarak artarak kontrol edilemez duruma gelebilir. Bu durumda düzenli ifadeler ile desteklenmiş grep komutları yazarak aradığınızı bulmak tek çıkar yoldur ve tek bir log dosyası kullanmak bu nedenle tavsiye edilmez.

```
*.*                 /var/log/all.log
```

syslog.conf(5) kılavuz sayfasına baktığımızda birden çok servisin, uygulamanın noktalı virgül ile ayrılarak tek bir kayıt dosyasında tutulabileceğini görürüz. Tüm kayıtların tek bir dosyaya toplanması durumunda authpriv kullanmadan kayıt edilmesi için aşağıdaki kural tanımlanabilir.

```
*.*; authpriv.none      /var/log/all.log
```

Ayrıca farklı servislerin, uygulamaların farklı önem derecesine sahip mesajlarını tek bir dosyada tutmak yerine bunları önem derecelerine göre ayırarak saklanması sağlanabilir. Bunun için <,> ve 0 operatörleri kullanılır. Önem derecesine göre mesajları bu operatörler ile ayrı dosyalara kayıt edebiliriz.

```
mail.info           /var/log/maillog
mail.=debug         /var/log/maillog.debug
```

Böylelikle dosya içerisindeki kayıtları incelmek için cat, grep, srot vb. araçları kullanmak durumunda kalmazsınız.

sudo standart olarak ve aksi belirtilmedikçe local2'ye kayıt tutmaktadır. Bu kayıtların tutulması için de aşağıdaki girdiyi kullanarak sudo kayılarını tutabilirsiniz.

```
local2.*            /var/log/sudo.log
```

syslog.conf(5) dosyasında yukarıda belirtilen kaynaklar dışında doğrudan uygulamanın kayıtlarının da tutulmasını sağlayabilirsiniz. Bu durum uygulamanın syslogd(8) destekli olmaması durumunda söz konusudur ancak syslog.conf(5) içinde yapılacak bir düzenleme ile bunu çözebilirsiniz.

syslog.conf(5) dosyasında bu durumu gösteren örnek bir satır bulunmaktadır. PPP kayıtlarını tutumak için hazırlanmış olan satır aşağıda görülmektedir.

```
!ppp
*.*                 /var/log/ppp.log
!*
```

Programın syslogd tarafından izlenebilmesi için programın adı ünlem işareti ile tanımlanır. İzleyen satırda ise kaynak.derece yapısında kural tanımlanıp kayıt dosyası belirtilir. Girdiyi sonlandırmak için de !* kullanılır. Böylelikle söz konusu programın syslogd tarafından kaydı tutulabilecektir.

syslog(3) UNIX sistemlerin istemci sunucu mimarisini kullandığı için yerel olarak kayıt yapabildiği gibi aynı zamanda ağ üzerindeki bir sistemi de syslog kayıtlarının tutulduğu bir syslogd sunucusu olarak kullanabiliriz. Bu ayrı bir yazının konusu ancak syslog.conf859 dosyasına göz attığınızda uzak sistemlerin @sistem_adı ile tanımlanabildiğini görebilirsiniz.

Yukarıdaki örneklerden yola çıkarak

```
mail.info           @lucrehulk
mail.=debug         @lucrehulk
```

Bu girdiler söz konusu olan mail kaynağı kapsamında olan tüm mesajları lucrehulk adlı sisteme yönlendirecektir.

UID/GID VE CHMOD

Kullanıcı ve grup numaraları

Gökşin Akdeniz
goksin@enixma.org

UNIX sistemlerde erişim kontrolü temel olarak kullanıcı ID ile bu uid ile bağlantılı olan süreçler ile ilişkilendirilmiştir. Bu yaklaşımda her süreç bir veya daha çok sayıda uid ve gid değerlerine sahiptir. Bu değerler sistem kaynaklarının hangilerinin kullanılacağını veya kullanılamayacağını -örneğin dosyalar veya bir ağ bağlantısının kullandığı port vb- belirlemek için kullanılır. Bu gruplar ve kullanıcılar içerisinde bazıları -örneğin root kullanıcısı- diğer süreçler ve sistem kaynakları üzerinde daha geniş yetkilere sahiptir. Bir kullanıcı uygulaması bazen sahip olduğundan fazla erişim yetkisine, örneğin bir dosyayı okuyup yazabilmesi gibi, gereksinim duyabilir. UNIX tasarım prensipleri gereği bir uygulama her zaman için asgari yetkilerle çalışmalıdır. Bu nedenle gereksinim duyulan yetki geçici olarak okuma/yazma yetkilerini alarak işlemi tamamlamalı ve ardından geçici olarak edindiği yetkilerden arındırılmalıdır. Bu süreç setuid(2) olarak adlandırılır. ve herhangi bir sürecin geçici olarak edindiği yetkiler nedeni ile çalıştırmakta olduğu kodun neden olabileceği zararın önlenmesini sağlar. UNIX sistemlerde bu geçici yetkilerin edinilip bırakılması süreci sistem çağrıları ile gerçekleştirilir. Bu sistem çağrılarına uid-setting system calls – uid-düzenleme sistem çağrıları adı verilir.

UID ve GID Modeli

UNIX sistemlerde kullanıcılar kimlik ile tanımlanır ve tanınır. Her kimliğin gerçekleştirebileceği eylemler sınırlıdır. Bu süreçte /etc/passwd dosyası kritik öneme sahiptir. Bu dosyada tanımlanan "user id/user identifier" ve "group id/group identifier" yani "kullanıcı tanımlayıcısı" ve "grup tanımlayıcısı" ile birlikte dosya erişim izinleri birlikte kullanılarak kullanıcının gerçekleştirebileceği eylemler sınırlanır. Bu durumu görmek için /etc/passwd dosyasına bakmak yeterlidir.

```
[goksin@tardis /usr/home/goksin]$ ls -l /etc/passwd
-rw-r--r--  1 root  wheel  1840 19 Nis 12:57 /etc/passwd
[goksin@tardis /usr/home/goksin]$
```

Dosya sadece root tarafından yazılabilir ve okunabilir iken, diğer kullanıcılar ve gruplar sadece okuyabilir. Buna ek olarak bazı uygulamalarda farklılıklar gözleyebilirsiniz.

Gid ve uid değerlerini /etc/passwd ve /etc/groups dosyasında görebilirsiniz. Kullanıcılar kendilerini tanımlayan bir kullanıcı adı ve numaraya sahiptir. Benzer olarak gruplar da aynı şekilde onları diğerlerinden ayırt eden birer grup adı ve numarasına sahiptir. uid ve gid numaralarına pozitif tamsayı değerleri atanır. Atanacak olan tam sayı değerleri eski

sistemlerde 15 bit olarak sınırlıdır. Bu da 0 ile 32767 arasındaki pozitif tamsayılara karşılık gelir. Güncel işletim sistemlerinde ise 32 bit sınırı geçerlidir. Bu durumda grup ve kullanıcı numaralarının üst sınırı 4294967295 olmaktadır.

Geleneksel olarak bir kullanıcının ve/vaya grubun alabileceği değerler UNIX ve türevi sistemler arasında küçük farklılıklar göstermekle birlikte önemli ölçüde aynıdır. Örneğin root kullanıcısı için kullanıcı ve grup numarası her zaman için sıfırdır. Buna karşılık nobody kullanıcısı her aman için 16 bitlik sayı aralığının sonundaki değerlerden birisini alacaktır. Nobody kullanıcısının grup ve kimlik numarası 65534 veya 65537 arasındaki değerlerden herhangi birisi olarak tanımlıdır. Bunlara ek olarak sistemde yer alan diğer kullanıcı ve gruplar - örneğin cups, hal daemon vs - grup ve kullanıcı numarası aynıdır. UNIX sistemler ise Linux dağıtımları arasında bir karşılaştırma yapacak olursak tam sayı aralığının aynı olduğu halde atanan numaralarda bazı değişiklikler gözlenecektir. Bu farklılıklar Linux dağıtımlarının kendi içerisinde gözlenir. Linux dağıtımlarının bazıları 0-500 arasını bazısı ise 0-1000 arasını sisteme ayırırken 1000 ve üzerini kullanıcılara ayırmaktadır. UNIX sistemlerde benzer bir durum olmakla birlikte 0-1000 arası sistem ve 1000 ve üzeri ise kullanıcı hesaplarına ayrılmıştır. Kullanılan yazılımlar ve diğer araçlar hemen hemen aynı olmakla birlikte atanan numaralar da farklılıklar gözlenebilir.

Bir UNIX sistemde süreçler yukarıda belirtilen uid değerlerine sahiptir. Bu değerler süreci başlatan kullanıcının uid değerleridir ve üç adettir; real user ID – gerçek kullanıcı ID(uid veya ruid), effective user ID – efektif kullanıcı ID (euid veya efektif uid), saved user ID – kayıt edilmiş ID(saved uid veya suid). Gerçek kullanıcı ID değeri süreci başlatan kullanıcının kullanıcı ID değeri ile aynıdır. Efektif kullanıcı ID ise sıklıkla erişim izinleri için kullanılır. Kayıt edilmiş ID değeri ise sürecin geçici olarak farklı bir kullanıcının yetkileri ile çalıştırılıp işlemin sonlanmasının ardından asıl kullanıcı yetkilerine geri dönmek için kullanılır.

Benzer olarak bir süreçler aynı şekilde üç tane grup ID – gid değerine sahiptir: Real group ID – gerçek grup ID, effective group id – efektif grup id ile saved group id – kayıt edilmiş grup id değerleridir. UNIX ve türevi sistemlerde grup id değerleri genel olarak ait oldukları kullanıcı id değerleri ile aynıdır.
setgid(2) değerleri kullanıcı ID ile yanı işlevi gruplar için yerine getirir.

Linux dağıtımlarında ise bunlara ek olarak dosya sistemine erişimi düzenleyen fsuid ve fsgid kullanıcı ve grup tanımlamalarına da sahiptir. Bu değerler POSIX standartlarında tanımlı değildir. Dolayısıyla da POSIX uyumlu setuid(2) programları yazılırken birçok programcı tarafından dikkate alınmaz, bu durumu kapsayan bir hata mesajı hazırlanabilir ya da bu değerleri dikkate alan özel bir kod bloğu yazılarak kullanılabilir.

chmod(2) İle setuid(2) ve setgid(2) Tanımlamak

setuid(2) ve setgid(2) değerleri bir dosya veya dizin için tanımlanır. Dosyalar için tanımlama işlemi ise chmod(2) ile gerçekleştirilir. Chmod(2), bir dosya veya dizin için erişim izinlerini düzenlemek için kullanılır ama aynı zamanda dosya veya dizin için setuid(2) ve setgid(2) değerlerini de düzenleyebilir. Bunu yapmak için chmod komutunu sahip, grup ve diğerleri için belirten değerlere ak olarak dördüncü basamak olarak setuid(2) için 4- ve setgid(2) için 2 değerleri, sticky(8) için ise 1 değeri verilir.

setgid(2) ve setuid(2) değerlerini bir dizin üzerinde chmod ile tanımlanması durumunda farklı işleyecektir. Dizin altında oluşturulacak olan her yeni dizin ve dosya söz konusu dizin için tanımlanmış olan gid değerlerini taşıyacaktır. Aslında burada beklenen dosyayı ve/veya dizini oluşturan kullanıcıya ait olan gid değerlerinin bu dizin ve/veya dosyalar için de geçerli olacağıdır. Kullanıcıya ait olan uid

değeri korunurken sadece gid değeri değişkenlik gösterecektir. Bu işlemi eğer daha önceden dosyalar ve alt dizinler barındıran bir dizinde yapacak olursak setgid(2) işlemi yapılmadan önce oluşturulmuş olan dosya ve dizinler etkilenmeyecek, ancak yeni oluşturulacak olan dosya ve dizinler için bu durum söz konusu olacaktır. Eğer var olan dosyalar ve dizinler için de setgid(2) işlemi yapılacak ise bu işlem ayrıca gerçekleştirilmelidir. Bu işlem ancak ve ancak dosya sistemi ve mount(8) suiddir destekliyorsa yapılabilir.

```
find /dizin_adı -type d -exec chmod g+s '{}' +
```

chmod ile yapılacak olan düzenlemelerde kullanılabilecek sayısal değerler şu şekildedir.

4000 (setuid bit) Çalıştırılabilen dosyalarda bu değerin atanması durumunda dosya efektif uid değeri dosya sahibinin uid değeri olarak tanımlanır. Dizinlerde ise yukarıda anlatıldığı gibi işler.

2000 (setgid bit) Çalıştırılabilen dosyalar çağrıldığında dosya sahibinin gid değeri efektif gid değerine atanır.

1000 (sticky bit) Dizine uygulandığında dizinin sahibi ve root kullanıcısı dosyaları değiştirebilir veya silebilir. Diğer kulanıcılar herhangi bir işlem yapamazlar.

0400 Sadece sahibi tarafından okunabilir.

0200 Sahibi tarafından yazılabilir.

0100 Dosyalarda kullanılırsa sahibi çalıştırabilir. Dizinlerde kullanılırsa diğer kullanıcılar bu dizinde arama yapabilir.

0040 Grup tarafından okunabilir.

0020 Grup yazabilir.

0010 Dosyalar için kullanılırsa grubu çalıştırabilir. Dizinlerde kullanıldığında grup tarafından dizinde arama yapılabilir.

0004 Diğerleri tarafından okunabilir.

0002 Diğerleri tarafından yazılabilir.

0001 Dosyalarda kullanılırsa diğerleri tarafından çalıştırılabilir. Dizinlerde kullanılırsa diğerleri tarafından dizinde arama yapılabilir.

OpenBSD Disk Bölümleme

Kurulumda disk bölümleme

Gökşin Akdeniz
goksin@enixma.org

OpenBSD'yi sorunsuz olarak kurabilmeniz için öncelikle sistemi ne amaçla kullanacağınıza karar vermeniz gereklidir. Bunun yanında kullanacağınız donanımın OpenBSD tarafından desteklenmesi de aynı derecede önemlidir. Yapılacak kurulumda sunucu, masaüstü veya bir ağ trafiğini kontrol etmek gibi farklı amaçlar farklı yapılandırmalara ve çalışma şekillerini gerektirecektir. Kurulum öncesinde buna karar vermeniz kurulumun süresini kısaltacağı gibi aynı zamanda sistem gereksinimi ile donanım seçiminden kaynaklı olabilecek sorunları ortadan kaldıracaktır. Birçok kullanıcının OpenBSD kurmaya niyetlenip sonra da kaldırmasının asıl nedeni eksik bilgi veya doğrudan deneme yanılma ile yaparım düşüncesi ile başladıkları kurulumlardır. OpenBSD kurulumu aslında ne yapacağını bildiğinizde son derece basittir ve güncel ve OpenBSD destekli bir donanım üzerinde on dakikadan daha uzun sürmeyecektir. OpenBSD kurmadan önce her BSD sistemde olduğu gibi kurulum notlarını kurulum öncesinde okumanız gereklidir. Bu notlar hem internet kaynaklarında hem de OpenBSD CD setlerinde yer alır.

OpenBSD kurulumuna geçmeden önce ilk adım donanımınızın desteklenip desteklenmediğinden emin olmanızdır. OpenBSD'yi birçok farklı donanım platformunda kullanabilirsiniz. Bunlardan yaygın olarak bilenenler: i386, AMD64, SPARC vb. Tam listesi için [1] OpenBSD sitesine veya yansılarına bakabilirsiniz. Bu sayfalarda her bir platform için verilen bağlantılarda o platformda desteklenen donanımların bir listesini bulabilirsiniz.

Kurulum yapacağınız sistemin güncel olması zorunlu değildir, eski bir sistemi de kullanabilirsiniz. Uzun bir süre OpenBSD kullanarak hazırlanmış bir yönlendirici/firewall olarak kullandığımız pentium 166 işlemcili bir sistem sorunsuz olarak çalıştı.

Kullanacağınız donanımın eski veya güncel olup olmamasından daha önemli olan kullanacağınız donanımın iyi durumda olmasının gerektiğidir. Arızalı bir RAM, problemli bir ağ kartı veya başka sorunlu olan bir donanım ya kurulum aşamasında ya da sonrasında problem olacaktır. Dolayısıyla da donanım seçimi kurulum öncesi yapılması gereken kritik bir aşama durumuna gelmektedir.

Donanım Seçerken:

Birçok donanım üreticisi ilk bilgisayarların ortaya çıkmasından bu zamana dek geçen yıllarda ürettikleri donanıma ilişkin bilgi vermek yerine bunları gizli tutmayı tercih ettiler. Böylelikle pazarda rekabet etmek durumunda oldukları rakiplerinin kendi donanımlarının benzerlerini

üretmelerinin önüne geçebileceklerini ve pazardaki konumlarının güvence altında olacağını düşündüler. İlk bakışta bu düşünce doğru gibi görünse ve işletme fakültelerinde bu düşünce farklı bir şekilde rekabet edebilmenin ana ögelerinden birisi ticari sırdır olarak öğretilse de bilgisayar endüstrisi bunun o kadar da iyi bir fikir olmadığını kanıtladı. Birçok üretici tarafından kolaylıkla üretilebilen benzer özelliklere sahip olan ve donanım spesifikasyonları hemen hemen tüm üreticiler tarafında erişilebilen donanımlar hem çok sayıda üretici tarafından hemde çok büyük miktarlarda üretilince pazardaki dengeleri oldukça değiştirmiş oldu. Böylelikle sadece bir üretici tarafında üretilen donanımlar kısa zamanda pazardaki paylarını diğer donanımlara kaptırdı.

Bu durum öte yandan işletim sistemleri tarafında da önemli bir gelişme ile desteklendi .İşletim sistemi geliştiricileri donanım sürücüleri yazabilmek için donanıma ait bilgiye gereksinim duyuyorlardı. Donanım spesifikasyonlarının kolaylıkla erişilebilir olması geliştiricilerin bu donanımlar için sorunsuz çalışan sürücüler yazabilmelerine ve bu donanımların da işletim sistemi tarafından da desteklenmesini sağladı.

Pazarda sunulan tüm donanımlara ait olan spesifikasyonlar bulunmasa da bu donanımların yaygın kullanılması nedeni geliştiricilerin sıfırdan başlayarak elde bilgi olmadan sürücü yazmalarına da engel olmadı. Örneğin yaygın olarak bulunan Intel ağ kartlarının spesifikasyonları yayınlanmamış olsa da birçok geliştirici yaygın olarak kullanıldığı için bu donanım için sürücü yazmaya ve geliştirmeye devam etti. Bunun gibi bazı donanımlar için birçok işletim sisteminde sürücüler hazır olarak sunulur.

Bunun dışında yaygın olmayan veya spesifikasyonları olmayan birçok donanım ise sürücü yazılmasının zorluğu nedeni özellikle de yaygın olmayan donanımlar oldukları için -örneğin SUN Ultra SPAC III gibi- bazı işletim sistemlerince ve OpenBSD tarafından desteklenmez.

Eğer OpenBSD kurulumumu yapacağınız sistemde desteklenmeyen donanımlar bulunuyorsa bunları daha ucuz ve desteklenen eşdeğerleri ile değiştirmenizde yarar var.

İşlemci konusunda bir sıkıntınız olmayacaktır. INTEL, AMD ve IBM ve hatta Cyrix işlemciler ile de OpenBSD kullanabilirsiniz. İşlemcinizin sayısı ise kurulumda kurulum uygulaması tarafında kontrol edilerek ilgili çekirdek kurulacaktır. İşlemci seçimi sadece OpenBSD kullanırken bazı işlemlerin kısa veya uzun sürmesine etki edecektir.

Bellek miktarı konusunda bir sınırlama yok. Ancak bellek miktarınızı arttırmanız sizin yararınızadır. 16 MB ram'e sahip olan sistemlerde de OpenBSD kullanabilirsiniz ama 32 MB altına inmek pratikte “zaman” kayıplarına neden olacaktır. Öte yandan elinizde fazladan RAM olması arızalı bir RAM modülünün çıkarılıp sistemin sorunsuzca çalışmaya devam etmesi anlamına gelir.

Sabit disk olarak IDE, SCSI, SATA, PATA vs bir diski kullanabilirsiniz. Eski donanımları kullanacak iseniz IDE diskler yerine SCSI diskleri kullanmak veri transferinden kaynaklı darboğazların önüne geçecektir. Öte yandan SCSCI kartlar IDE disklere göre çok daha fazla sayıda sabit diski eş zamanlı kullanmanıza olanak vereceği için IDE disk yerine olanaklı ise SCSI disk tercih etmeniz daha iyi olacaktır. SATA, PATA diskler en yaygın olarak bulabileceğiniz disklerdir. Her ikisine de OpenBSD kurabilirsiniz. Kurulum için asgari 1 GB disk alanı yeterli olmaktadır. Bu durumda ise terfiler ve güncellemeler konusunda sıkıntı yaşayabilirsiniz. Kaynak kod gerekeceği için bu durumda düşük disk alanı kullanımı sıkıntı yaratacaktır. Bu nedenle disk seçiminde büyük diskleri tercih etmeniz yerinde olacaktır.

Kurulum Kaynakları

OpenBSD kurulumu için farklı kaynakları kullanabilirsiniz. OpenBSD projesinin devamını sağlamak için doğrudan 3 CD'den oluşan CD

setini sipariş edebilirsiniz. OpenBSD CD setleri OpenBSD sunucuları ile bant genişliği vb hizmetlerin devamlılığının sağlanması için kullanılmaktadır. Satın aldığınız her CD seti bir sonraki sürümün geliştirilmesinde ufak bir katkı sağlamaktadır. CD setini satın almak için sitesinden satış kanallarının bir listesine erişebilirsiniz.[1]

Satın aldığınız CD seti 3 CD olarak gelmektedir. Bu 3 CD'den her biri farklı bir veya birkaç platforma ait olan kurulum dosyaları ile depolardan alınmış bazı paketleri barındırmaktadır. Paketler her üç CD içerisinde dağınık olarak yer almaktadır. Ayrıca sistemin kaynak kodları yine bu CD seti ile gelmektedir. CD seti içerisinde ait olduğu sürümce desteklenen donanımların bilgisi ve kuruluma ilişkin bilgiler de yer almaktadır. Bunlara ek olarak da OpenBSD geliştiricileri tarafından çıkarılan sürüme ait olan bir şarkı CD içerisinde yer almaktadır. İyi eğlenceler...

OpenBSD kurulum kaynağı olarak 3 CD'den oluşan sete ait resmi ISO dosyaları nette bulunmaz, boşuna aramayın. Bazı p2p arama motorları ile internet sitelerinde CD1.iso vb dosyaları bulabilirsiniz ama bunlar resmi OpeNBSD ISO'ları değildir. Bulduğunuz iso dosyalarının bir meraklının kendi çabası ile oluşturduğu ISO dosyası olması kuvvetle muhtemeldir. Orijinal CD setine ait olan ISO dosyalarının telif hakları Theo De Raadt'a aittir. Bu ISO dosyaları OpenBSD kalite güvence ekibi tarafından kontrol edilerek hazırlanır ve ardından CD basılır. Bu CD setlerinin satışı OpenBSD geliştirme sürecinin masraflarının karşılanmasında kullanılır. Dolayısıyla bulacağınız ISO dosyaları eğer asıl kopyalardan oluşturulmuş ise bu Teho De Raadt'ın telif haklarının yasal olarak ihlal edilmesi anlamına gelir aynı zamanda da OpenBSD topluluğu tarafından da bu yapılan en basit anlamıyla “terbiyesizlik” olarak değerlendirilir. Topluluğun bazı üyelerinin tepkileri bundan çok daha sert olabilir.

Öte yandan indireceğiniz CD1.iso, CD2.iso ve CD3.iso dosyalarının da içeriğinde ne barındırdığını bilmeniz olanaklı değildir. Bu dosyaların içerisinde özel olarak hazırlanmış zararlı uygulamalar da barındırması da söz konusu olabilir. Bu nedenle herhangi bir siteden bulunacak olan bu iso dosyalarının gerçekten asıl OpenBSD sürümü olması söz konusu olmayacaktır. Bu durumda en güvenilir olanı OpenBSD CD setini sipariş etmek veya internet üzerindeki yansılarından erişebileceğiniz resmi kurulum iso dosyalarını veya ftp sunucularını kullanmak olacaktır.

OpenBSD'yi internetteki kaynakları kullanarak kurabilirsiniz. Ana FTP sunucusunda yer alan tüm dosyalar yansılarda bulunmaktadır. Dolayısıyla da coğrafi olarak size yakın olan bir yansıdan gereken tüm kaynakları edinebilirsiniz. Ana OpenBSD FTP sunucusu Kanada Alberta'da bulunmaktadır. Albetra Üniversitesi veri merkezinde yer alan sunucu, üniversiteye internet erişimini kullanmaktadır. Bu sunucunun diğer yansılar tarafından da kullanıldığını göz önüne alarak yansıların tercih edilmesi hız ve zaman kazandıracaktır.

Yansıların güncel bir listesi [2] OpenBSD sitesinde yer almaktadır. Tercih ettiğiniz protokole göre -ftp, http, afs ve rsync- var olan yansılardan bir tanesini seçin. Bu yöntem ile kurulum kaynaklarını edinmeden önce OpenBSD ana sunucusu ve yansıların dizin yapısı hakkında bilgi edinmeniz gerekli. OpenBSD sunucuları üzerindeki dizin yapısı ilk bakışta karmaşık gibi gelebilir aradığınızı bulmanız zor olabilir.

Sunuculardaki ana dizin /pub/OpenBSD dizinidir. Bu dizin altında tüm OpenBSD ftp arşivi ve kaynakları bulunur. Bu dizine girdiğinizde en son sürüme ait olan dizin -bu yazı yazıldığı sırada 4.7 idi- en üstte yer alır. Güncel sürüm olan 4.7 dizini sunucularda /pub/OpenBSD/4.7/ olarak yer alır ve içerisinde desteklenen platformlara ait olan alt dizinler yer alır. Edinmek istediğiniz kurulum kaynaklarına ulaşmak için bu dizinlerden ilgilendiğiniz platforma ait olan dizine girmeniz gereklidir.

```
[goksin@tardis /usr/home/goksin]$ ftp
ftp> open ftp.ulak.net.tr
Trying 193.140.100.10...
Connected to tulumba.ulakbim.gov.tr.
220 "ftp.ulak.net.tr  Microsoft FTP Service (Version 5.0).
Name (ftp.ulak.net.tr:goksin): anonymous
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> cd OpenBSD
250 Directory successfully changed.
ftp> pwd
Remote directory: /OpenBSD
ftp> ls
229 Entering Extended Passive Mode (|||53262|)
150 Here comes the directory listing.
drwxr-xr-x    9 1004     14            512 Jun 04  2001 2.0
drwxr-xr-x   11 1004     14            512 Jun 04  2001 2.1
drwxr-xr-x   12 1004     14            512 Jun 04  2001 2.2
drwxr-xr-x   15 1004     14            512 Jun 04  2001 2.3
drwxr-xr-x   13 1004     14            512 Jun 04  2001 2.4
drwxr-xr-x   14 1004     14            512 Jul 08  2000 2.5
drwxr-xr-x   12 1004     14            512 Jun 07  2001 2.6
drwxr-xr-x   13 1004     14            512 Oct 24  2003 2.7
drwxr-xr-x   15 1004     14            512 Oct 24  2003 2.8
drwxr-xr-x   15 1004     14            512 Oct 30  2002 2.9
drwxr-xr-x   15 1004     14            512 Oct 20  2005 3.0
drwxr-xr-x   15 1004     14            512 Oct 20  2005 3.1
drwxr-xr-x   15 1004     14            512 Oct 20  2005 3.2
drwxr-xr-x   14 1004     14            512 Nov 06  2004 3.3
drwxr-xr-x   16 1004     14            512 Nov 06  2004 3.4
drwxr-xr-x   18 1004     14           1024 Oct 20  2005 3.5
drwxr-xr-x   19 1004     14           1024 Nov 12  2004 3.6
drwxr-xr-x   21 1004     14           1024 Apr 01  2005 3.7
drwxr-xr-x   21 1004     14           1024 Sep 25  2005 3.8
drwxr-xr-x   21 1004     14           1024 May 01  2006 3.9
drwxr-xr-x   22 1004     14           1024 Dec 10  2006 4.0
drwxr-xr-x   22 1004     14           1024 Apr 30  2007 4.1
drwxr-xr-x   19 1004     14           1024 Oct 31  2007 4.2
drwxr-xr-x   21 1004     14           1024 Apr 30  2008 4.3
drwxr-xr-x   22 1004     14           1024 Sep 04  2008 4.4
drwxr-xr-x   22 1004     14           1024 Mar 25  2009 4.5
drwxr-xr-x   22 1004     14           1024 Oct 09  2009 4.6
drwxr-xr-x   22 1004     14           1024 Apr 04 22:23 4.7
drwxr-xr-x    2 1004     14            512 Nov 11  2009
OpenBGPD
drwxr-xr-x    3 1004     14            512 Nov 11  2009
OpenNTPD
drwxrwxr-x    4 1004     14           4608 Aug 23 11:18
OpenSSH
-rw-r--r--    1 1004     14           1508 Jan 04  2010 README
-rw-r--r--    1 1004     14           1508 Jul 07  2009
README~
drwxr-xr-x    8 1004     14            512 Aug 16  2001 cvs
drwxr-xr-x    2 1004     14           2560 Aug 20 01:46 doc
-rw-r--r--    1 1004     14           6162 Jul 05 17:24
ftplist
drwxr-xr-x   28 1004     14           1024 Jun 21 22:13
patches
drwxr-xr-x   21 1004     14            512 Aug 16 20:14
snapshots
drwxr-xr-x    2 1004     14           1024 Mar 18 19:27 songs
-rw-r--r--    1 1004     14             11 Aug 29 18:00
timestamp
drwxr-xr-x    2 1004     14            512 Jan 07  2005 tools
226 Directory send OK.
ftp> cd 4.7
250 Directory successfully changed.
ftp> ls
229 Entering Extended Passive Mode (|||53490|)
150 Here comes the directory listing.
-rw-r--r--    1 1004     14          30596 May 19 13:06
ANNOUNCEMENT
drwxr-xr-x    2 1004     14           1024 Jun 02 10:06
Changelogs
-rw-r--r--    1 1004     14           2280 Mar 24 01:24
HARDWARE
-rw-r--r--    1 1004     14           3488 Mar 24 01:24
PACKAGES
-rw-r--r--    1 1004     14           2375 Mar 24 01:24 PORTS
```

```
drwxr-xr-x    2 1004     14             512 Jun 02 10:06 alpha
drwxr-xr-x    2 1004     14             512 Jun 02 10:06 amd64
drwxr-xr-x    2 1004     14             512 Jun 02 10:06 armish
-rw-r--r--    1 1004     14           10608 Mar 05 20:23
ftplist
drwxrwxr-x    2 1004     14             512 Jun 02 10:06 hp300
drwxr-xr-x    2 1004     14             512 Jun 02 10:06 hppa
drwxr-xr-x    2 1004     14            1024 Jun 02 10:06 i386
drwxr-xr-x    2 1004     14             512 Jun 02 10:06
landisk
drwxr-xr-x    2 1004     14             512 Jun 02 10:06
loongson
drwxr-xr-x    2 1004     14             512 Jun 02 10:06 macppc
drwxr-xr-x    2 1004     14             512 Jun 02 10:06
mvme68k
drwxrwxr-x    2 1004     14             512 Jun 02 10:06
mvme88k
drwxr-xr-x   14 1004     14             512 Jun 02 10:06
packages
-rw-r--r--    1 1004     14        18002486 Mar 19 15:25
ports.tar.gz
-rw-r--r--    1 1004     14            4546 Mar 19 15:25
root.mail
drwxr-xr-x    2 1004     14            1024 Jun 02 10:06 sgi
drwxr-xr-x    2 1004     14             512 Jun 02 10:06 socppc
drwxr-xr-x    2 1004     14            1024 Jun 02 10:06 sparc
drwxr-xr-x    2 1004     14            1024 Jun 02 10:06
sparc64
-rw-r--r--    1 1004     14       131759003 Mar 21 19:17
src.tar.gz
-rw-r--r--    1 1004     14        20668814 Mar 21 19:17
sys.tar.gz
drwxr-xr-x    3 1004     14             512 Jun 02 10:06 tools
drwxr-xr-x    2 1004     14             512 Jun 02 10:06 vax
-rw-r--r--    1 1004     14       106253503 Mar 19 15:25
xenocara.tar.gz
drwxr-xr-x    2 1004     14             512 Jun 02 10:06 zaurus
226 Directory send OK.
ftp> cd amd64
250 Directory successfully changed.
ftp> ls
```

```
229 Entering Extended Passive Mode (|||63694|)
150 Here comes the directory listing.
-rw-r--r--    1 1004     14           86297 Mar 20 22:13
INSTALL.amd64
-rw-r--r--    1 1004     14            1807 Mar 20 22:13 SHA256
-rw-r--r--    1 1004     14        55089310 Mar 20 22:13
base47.tgz
-rwxr-xr-x    1 1004     14         8350595 Mar 20 22:12 bsd
-rwxr-xr-x    1 1004     14         8378251 Mar 20 22:12 bsd.mp
-rwxr-xr-x    1 1004     14         7299401 Mar 20 22:12 bsd.rd
-rw-r--r--    1 1004     14         7417856 Mar 20 22:12
cd47.iso
-r-xr-xr-x    1 1004     14           43168 Mar 20 22:12 cdboot
-r-xr-xr-x    1 1004     14            2048 Mar 20 22:12 cdbr
-rw-r--r--    1 1004     14       111000061 Mar 20 22:12
comp47.tgz
-rw-r--r--    1 1004     14          521741 Mar 20 22:12
etc47.tgz
-rw-r--r--    1 1004     14         1474560 Mar 20 22:12
floppy47.fs
-rw-r--r--    1 1004     14         2707181 Mar 20 22:12
game47.tgz
-rw-r--r--    1 1004     14            1401 Jun 02 10:06
index.txt
-rw-r--r--    1 1004     14       284704768 Mar 20 22:12
install47.iso
-rw-r--r--    1 1004     14         9462372 Mar 20 22:12
man47.tgz
-rw-r--r--    1 1004     14          364789 Mar 20 22:12
misc47.tgz
-r-xr-xr-x    1 1004     14           53492 Mar 20 22:13
pxeboot
-rw-r--r--    1 1004     14        17235032 Mar 20 22:13
xbase47.tgz
-rw-r--r--    1 1004     14           71698 Mar 20 22:13
xetc47.tgz
-rw-r--r--    1 1004     14        39688438 Mar 20 22:13
xfont47.tgz
-rw-r--r--    1 1004     14        21365491 Mar 20 22:13
xserv47.tgz
-rw-r--r--    1 1004     14         2946216 Mar 20 22:13
```

```
xshare47.tgz
226 Directory send OK.
ftp>
```

Kurulumu yapabilmeniz için yansılardan iki farklı şekilde yararlanabilirsiniz. Birincisi ağ üzerinden kurulum yapmak ikincisi ise yansılardan edineceğiniz resmi kurulum iso ve ilgili diğer dosyaları indirip bir cd'ye yazarak kurulumu gerçekleştirmek.

Ağ üzerinden kurulum yapacaksanız protokol olarak ftp'nin tercih edilmesi daha iyidir. FTP üzerinden kurulum yapılması http kullanılmasına göre hata düzeltmesi desteği sunduğu için kurulum işlemi sırasında ağ aktarımından kaynaklı problemlerin önüne geçilebilir. Ancak bulunduğunuz ağ üzerinde bir proxy sunucusu bulunuyorsa bu durumda ftp yerine http kullanmanız zorunlu olacaktır. Bu durumda diğer bir seçenek ise yerel bir kurulum sunucusu oluşturmak da olabilir. Bu yöntem yine ağ üzerinden kurulum yapmanıza olanak verirken yerel bir sunucu kurmanızı gerektirir. Yerel sunucu kurmak yerine aynı disk üzerinde birden çok işletim sistemi yer alıyorsa bu durumda OpenBSD'nin desteklediği dosya sistemlerinden birisine sahip olan disk bölümüne gereken dosyaları indirip bu dosyaları kullanarak da kurulum yapabilirsiniz.

Yerel ağ üzerinde bir sunucuyu ağdan kurulum için kullanacak iseniz bu durumda sadece kurulum yapacağınız platforma ait olan dizindeki örneğin i486 veya amd64 gibi dosyaları yerel sunucuya indirmeniz yeterli olacaktır. Bu aşamadan sonra ise yerel sunucuya erişim için gerekli olan bilgilere gereksiniminiz olacaktır.

Ağ üzerinden kurulum yapmak yerine resmi kurulum iso dosyalarını da kullanabilirsiniz. Bu durumda kurulum yapmak istediğiniz sistem veya sistemlerde bir optik sürücü bulunması zorunludur. OpenBSD resmi 3 CD seti dışında yansılarda ve ana sunucuda kurulum için kullanılabilen bir iso dosyası bulunur. Bu dosya sürüm numarasını da barındıran örneğin 4.7 sürümü için install47.iso dosyasıdır. Bu dosyayı kullanarak kurulum yapabilirsiniz.

install47.iso dosyası OpenBSD kurmak için gerekli olan bileşenleri içerir. Bu bileşenler ayrıca ftp üzerinden gerçekleştirilecek olan kurulumlar için de sunucudaki aynı dizinde yer alır. OpenBSD kurmadan önce kurulumda kullanan bu bileşenlerden hangilerini kuracağınıza karar vermeniz gerekir. İlk kurulumda bu bileşenlerin hepsini seçebilirsiniz. Sonraki kurulumlar ve güncellemelerde ise tercihlerinize göre seçim yaparak sayısını azaltabilirsiniz.

OpenBSD Bileşenleri

bsd – OpenBSD kernel'i. Kurulması zorunludur.Kurmamayı seçebilirsiniz ama kurulum sırasında bazı bileşenler kernel olmadan kurulmayacaktır. Kurulumu tamamlasanız bile sistem açılmayacaktır.

bsd.mp – Çoklu işlemciye sahip sistemlerde kullanılan OpenBSD kernel'idir. Bu bileşen sadece bazı platformlarda kullanılabilir.

bsd.rd – RAM DISK OpenBSD kernel'idir. Bu bileşen root dosya sistemidir. Sistemi bununla başlattığınızda temel bir OpenBSD sistemde olması gereken tüm araçlar elinizin altında olur. Hatta bsd.rd'nin yeni sürümünü edinip bununla sistemi başlatıp OpenBSD dahi kurabilirsiniz.

base47.tgz – Bu bileşen OpenBSD için gerekli olan tüm temel araçları barındırır. Bazı kaynaklarda bunun için OpenBSD'yi UNIX yapan öğedir denir. Diğer bir deyişle /bin, /sbin, /usr/bin ve /usr/sbin, kütüphaneler ve diğer uygulamalar bu bileşen içerisinde yer alır. Kurmanız zorunludur.

etc47.tgz - /etc dizininde yer alan tüm dosyalar bu bileşen içerisinde yer alır. Bunlara ek olarak da gerekli olan /var/log ve /root dizinleri de bunun içindedir. Kurmanız zorunludur.

comp47.tgz – Derleyiciler (C/C++, Fortran vb) ile gerekli olan diğer araçları barındırır. Aynı zamanda derleyicilere ait olan kılavuz sayfalarını barındırır. Eğer ports kullanmak istiyorsanız bu bileşenin kurulması zorunludur. Aynı zamanda OpenBSD tarafından duyurulan yamalar da kaynak kod olarak geldiği için ilerideki yamaları kurabilmek için gereksinim duyabilirsiniz. Bu yamalar güvenlik açıkları için değil kurulan bileşenlerdeki hataları örneğin X gibi gidermek için yayınlanır. Bir ağ geçidi veya firewall tarzı kurulumda buna gereksinim duymazsınız ve derleyici bileşenini kurmanıza gerek yoktur. Zira sisteme yetkisiz erişmeye çalışanlar için sistemde yer alan derleyici ve ilgili diğer araçları her zaman bulunmaz bir nimettir.

man47.tgz – base ve etc'de yer alan uygulamaların kılavuz sayfalarına gereksinim duyuyorsanız bu bileşeni kurmanızda yarar var. Diğer bileşenlerde yer alan araçların kılavuz sayfaları, uygulamalar ile birlikte gelmektedir.

misc47.tgz – Sözlükler, belgelendirme, kurulum ve yapılandırma vb belgeler burada yer alır. Masaüstü olarak kullanacaksanız işinize yarayacaktır ama sunucu olarak düşündüğünüz bir sistemde gerekli olmayacaktır.

game47.tgz – Orijinal BSD 4.4-Lite ile dağıtılan oyunları ve kılavuzlarını barındırır. Örneğin UNIX klasiği kabul edilen fortune(1) ve rogue(6) vb bulunur. Eski UNIX sistemlerdeki oyunları merak ediyorsanız kurabilirsiniz. Bu bileşende yer alan oyunların daha gelişmiş sürümleri ports üzerinde yer alır.

xbase47.tgz – X11 için gerekli olan temel dosyalar ve araçlar burada yer alır. Eğer X'e gereksinim duyacak iseniz bunu kurmanızda yarar var. Öte yandan sisteme bağlı olan bir monitör vs. olmasa da bazı uygulamalar için X11 kurulu olması gerekli olabilmektedir.

xetc47.tgz - /etc/X11 ve /etc/fonts yapılandırma dosyalarını barındırır. X11 kullanacaksanız kurmanız zorunludur.

xfont47.tgz – X11 font sunucusu ve fontları barındırır.

xserv47.tgz – X11 için gerekli olan X sunucusunu ve sürücülerini barındırır.

xshare47.tgz – X için gerekli olan yerel dosyaları ve kılavuz sayfaları vb bileşenleri barındırır.

Bileşenlerden hangilerini seçeceğinize karar vermeniz kullanım amacınıza göre değişecektir. Eğer OpenBSD öğrenmek istiyorsanız masaüstü olarak kurmanız başlangıç için uygun olacaktır. Bu durumda bsd.mp hariç olmak üzere diğer bileşenler bir masaüstü sistem için yeterlidir.

Disk Bölümleme

Bir BSD sistemi kurmanın belki de en zor olan yanı disk bölümlemesini nasıl yapacağınıza karar vermektir. Özellikle de diski bölümlemenin ne işe yaradığı veya ne demek olduğunu tam olarak bilmeyen birisi için hangi işletim sistemi olursa olsun disk bölümlemek eziyetten başka bir şey değildir. OpenBSD kurulum uygulaması diğer BSD sistemlerin kurulum uygulamaları gibi metin tabanlıdır ve sistemi kuran kişinin ne yaptığını bildiğini ve konuya tam olarak hakim olduğunu var sayar. Diğer bir deyişle diski bölümlerken gereken araçları kullanabildiğiniz kabul edilir. Ayrıca diskinizi tek bir işletim sistemi ve birden çok işletim sistemi kurarak kullanmak da farklı şekillerde disk bölümlemeyi gerektirir.

Burada şu soru akla gelebilir: “Diskini bölümlemesi zorunlu mudur?” Yanıt evet. Birçok işletim sistemi diski tek bir bölüm olarak kullanmanızı teşvik eder. Bu durumda disk üzerinde sadece tek bir işletim sistemi kurabilirsiniz. Diğerleri için yer kalmayacaktır. Windows işletim

sistemi ailesi farklı sürümleri tek bir disk bölümü üzerinde kullanabilmektedir ama farklı işletim sistemleri tek bir disk bölümünde kurulup kullanılmamaktadır. Bu durumda disk bölümlerini düzenleyen uygulamalar ile disk üzerinde boş alan yaratabilirsiniz ama her işletim sistemi bundan memnun olmamaktadır.

Bu nedenle tek bir işletim sistemini bir tane büyük disk alanını kullanacak şekilde sisteminize kurmak yerine diskinizi bölümleyerek kullanmanız yerinde olacaktır. OpenBSD'yi tek işletim sistemi olarak kullanmak isteseniz de diğer BSD işletim sistemleri gibi diskiniz gene bölümlenecektir. Bu garip gelebilir ama UNIX sistemler ile çalışıyorsanız bu olağan uygulamadır. Diskinizi bölümleniz için bir çok neden vardır:

Birincisi, sabit diski oluşturan disklerin açısal hızı aynıdır ama disk üzerindeki bir noktanın çizgisel hızı değişkendir; bazıları daha hızlı döner bazıları daha yavaş. Sık erişilen veriyi daha hızlı dönen disk alanı üzerinde tutarsanız sistemin genel olarak performansını iyileştirmiş olursunuz. Bunu yapabilmenin tek yolu da diskinizi bölümlemektir. İşletim sistemi bu disk bölümlerini gerektiği gibi kullanır. Her bir disk bölümü için ayrı kurallar tanımlayabilirsiniz ve bu disk bölümleri bu kurallara göre işletim sistemi tarafından gerektiği gibi kullanılır.

Diskteki root bölümü tüm donanımlara ilişkin olan bilgileri barındırır. Bu bölüm ayrı bir özen gerektirir. Kullanıcı verilerinin yer aldığı bölüm üzerinde herhangi bir setuid'ye sahip olan bir uygulama hatta daha da ileri gidersek herhangi bir program bulunmamalıdır. Sistem yapılandırmasını ayrı bir disk bölümünde tutmak ve bunu değiştirilemez yapmak yerinde olacaktır. Kullanıcılara verilen alan sınırlandırılmalı ve gereksiz dosyalar (mp3, film vb) ile diski doldurmalarının önüne geçilmelidir. Bunların dışında ayrıca disk bölümlerinden birisinin zarar görmesi durumunda diğer bölümler bundan etkilenmeyecektir. Doğru olarak bölümlenmiş bir disk aynı zamanda sistemin güvenliğine de olumlu katkıda bulunacaktır.

Bir diski bölümlemeden önce kullanım amacınıza göre gereksinim duyulabilecek olan disk bölümlerinin tahmini olarak boyutlarını belirlemeniz gerekecektir. OpenBSD tek sistem olarak kurmak ve diğer işletim sistemleri ile birlikte kullanmak için disk bölümleme işlemleri farklı olacaktır. OpenBSD kurulum için seçtiğiniz bileşenlere göre disk bölümlerini kendisi ayarlayabilmektedir. Bu işlem kurulum sırasında otomatik disk bölümleme ile yapılabilir. Ancak diski elle bölümleyecekseniz, bu durumda kullanım amacınıza göre şu disk bölümlerinin oluşturulması ileriye dönük olarak işinizi kolaylaştıracaktır: /, swap, /tmp, /usr/, /var ve /home.

/

Sistem yapılandırmanıza ait olan dosyaların tamamı ve sistem için gerekli olan temel araçların tümü / bölümünde yer alır. Kurtarma ve bakım durumlarında tek kullanıcı kipinde sistemin başlatılması durumunda bu araçların erişilebilir olması için tamamı / altında yer almalıdır. Bu bölümü birden çok diski olan bir bilgisayardaki kurulum sırasında birinci disk üzerinde oluşturmanız gereklidir. / bölümü için 500MB bir disk alanı yeterli olacaktır ve de 100 MB altına inmemeniz de önerilir.

/ disk bölününü oluştururken dikkat etmeniz gereken diğer nokta ise / bölümünün disk üzerindeki ilk 8GB alan üzerinde olması gerektiğidir. Bu sınırlama BIOS sınırlamaları nedeni ile dikkat edilmesi gereken bir durumdur. BIOS diskin ilk bölümündeki veriyi okuyabilir. Bu nedenle işletim sisteminin BIOS tarafından okunabilen bu alanda yer alması zorunludur. Aksi halde işletim sistemi başlatılamaz. Eski BIOS için bu sınır 504 MB ve güncel sistemler için ise 8GB 'tır. Bu sınır ancak tüm sistemler daha doğrusu platformlar için bu sınırlama geçerli olmamakla birlikte OpenBSD bu sınırlamaya uyar. Dolayısıyla / bölünün bu ilk 8GB alanda yer almasında yarar var. Ayrıca / bölümünü 500

MB'tan az yapmanız durumunda sisteminizi terfi edip yeniden başlattığınızda açılmama olasılığı da söz konusu olabilir.

Takas Alanı – SWAP

Sanal bellek sistemi tarafından kullanılacak olan disk alanı diğer önemli disk bölümüdür. İşler yolunda gittiği zaman bu alanı kullanmanıza gerek olmayacaktır ama bazen bu alanın da kullanılması gerekebilir. Bu alanın diskin hızlı bir bölümünde olması zorunludur. Takas alanının boyutu sıklıkla tartışma konusu olmaktadır. Genel olarak üzerinde birleşilen düşünce bu alanın sistemin fiziksel belleğinin en az iki katı olmasıdır. OpenBSD otomatik disk bölümleme sırasında bu alanı yaklaşık olarak 200 MB ile 1 GB arasında seçmektedir. Fiziksel belleğin tükenmesi durumunda takas alanı kullanılmaya başlayacaktır. Bazen de takas alanı tükeneceğinden bu durumda yeni bir takas alanı oluşturup kullanmaya başlayabilirsiniz. Bunun için ise yeni takas alanı oluşturmak üzere boş disk alanı gerekecektir. Bazı durumlarda da fiziksel belleği arttırmak takas alanı kullanımını ortadan kaldıracaktır. Bu gibi gereksinmeleri de düşünerek ileriye dönük olarak takas alanını fiziksel belleği arttıracağınızı düşünerek biraz büyük seçebilirsiniz.

Takas alanının büyüklüğünü belirlerken eğer birden çok diski kullanacaksanız bu durumda takas alanını birçok diske bölerek oluşturabilirsiniz. Bu tür yapılandırmalarda SCSI diskler diğerlerine göre üstün olmaktadır. Bu yapılandırmayı SATA ve PATA diskler ile deneme olanağım olmadığı için bir yorumda bulunamayacağım.

/tmp

/tmp bölümü sistem genelinde kullanılan geçici dizindir. Eğer bunu ayrı bir disk bölümü olarak oluşturmazsanız doğrudan / bölümüne eklenecektir ve / bölümü ile aynı bölümde olduğu için aynı kurallara tabi olacaktır. Özellikle de / bölümlerinin salt okunur olarak bağlandığını düşünürsek bu /tmp'nin işlevini yerine getirmesine olanak vermeyecektir. Bu dizini /home altında oluşturup kullanmayı tercih edebilirsiniz. Büyüklüğü ise size kalmış durumda. 200 MB'tan daha düşük seçmemek önerilir. Günümüzün büyük kapasiteli sabit disklerini düşünecek olursa bu durumda 500 MB yeterince büyük olacaktır.

/var

/var bölümü e-posta, log dosyaları ve sıklıkla değişen dosyalar ve web sunucusunun dosyalarının bulunduğu dizindir. Eğer sunucu amaçlı olarak kurulum yapıyorsanız bu durumda e-posta ve web sunucusu için yaklaşık olarak 1GB disk alanı yeterli olabilmektedir. Ancak veritabanı ve e-posta vb sunucu servislerinin disk gereksinimleri daha fazla olacağından daha fazla bir alan ayırmanız gerekebilir. Diskin kalan bölümünün %20 ile %70 sunucu amaçlı kurulumlar için /var disk bölümüne ayrılabilir.

/usr

/usr dizini işletim sisteminin kaynak kodlarını, sistem programlarını, derleyicileri ve diğer dosyaları barındırmaktadır. Sistemi bir üst sürüme terfi ederken /usr bölümü önem kazanmaktadır. Asgari boyut olarak 8 GB'tan daha az seçmemeye özen göstermeniz yerinde olacaktır.

/home

Kullanıcılara ait olan dosyaların barındığı disk bölümü için fazla alan ayırmanız uygundur. Genelde bu disk bölümü eğer aynı sistemde birden çok kullanıcı bulunuyorsa ve kota uygulamıyorsanız kısa zamanda dolabilmektedir. Bu disk bölümünün diskin en sonundaki alan olmasının da bir sakıncası yoktur. Aynı zamanda çok büyük bir alan ayırmanız da zorunlu değildir. Gereksinmelerinize göre boyutunu seçebilirsiniz.

OpenBSD Disk Bölümleme

Sisteminizde birden çok disk bulunuyorsa bu durumda disk bölümlerini farklı diskler üzerinde tutmak iyi olacaktır. Planlamasını kurulum öncesinde yaparsanız hem kurulum hem de sonrasındaki işletim aşamasında işler daha kolay olacaktır. Birden çok disk bulunuyorsa takas alanını tüm disklere yayabilirsiniz. Kurulumu bir web sunucusu için yapacaksanız ikinci diski /www veya /home olarak bölümleyebilirsiniz. E-posta sunucusu ise ikinci diski /var veya /var/mail olarak bölümlemek en iyisi olacaktır. Eğer merkezi bir kayıt sunucusu olarak kullanacaksanız bu durumda /var/log olarak ikinci diski bölümlemek ve kullanmak en iyisi olacaktır. Genel olarak işletim sistemi ile işlenecek olan veriyi birbirinden fiziksel olarak ayırmak yani birden çok diske dağıtmak kişisel görüşüm olarak en uygun çözümdür. Disk bölümleme açısından en ideal çözüm sistemin kullanım amacına, donanıma ve en önemlisi sistemi yapılandıran ve yöneten sistem yöneticisinin bilgi ve deneyimine dayalı olarak değişir. Ancak unutulmaması geren en önemli kural birinci disk diğerlerinden daha yavaş ise bu diski kullanmaktır. Tüm sistem yöneticileri bunda hem fikirdir. Ayrıca birden çok işletim sistemini kullanacaksanız bu durumda birden çok disk kullanmak yerinde olacaktır.

[1] http://www.openbsd.org/orders.html
[2] http://www.openbsd.org/ftp.html