Empowered by Innovation NEC

LAPP/SELinux

A secure web application stack powered by SELinux

KaiGai Kohei <kaigai@ak.jp.nec.com>
NEC OSS Promotion Center

Self Introduction

KaiGai Kohei

has worked at NEC for 7 years,

and contributed to SELinux development, such as

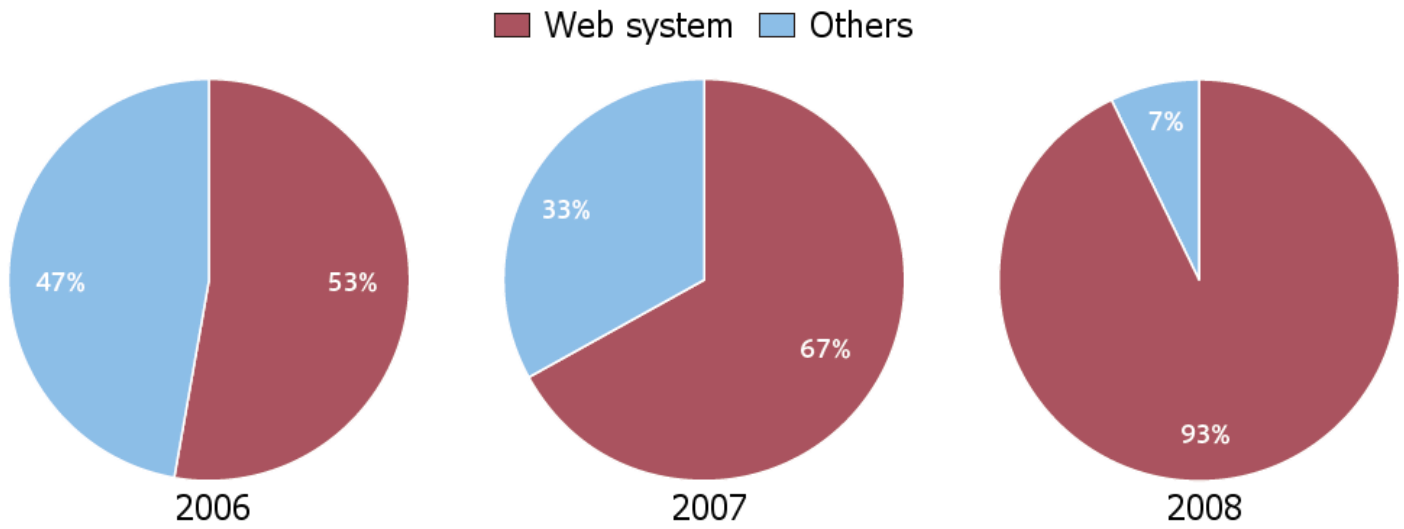
- SMP scalability improvement
- A series of efforts to port embedded platform
- Development of SE-PostgreSQL
- Per thread security context setting support

Recently, I have focused on web-system's security.

Empowered by Innovation NEC

1. Background
2. SE-PostgreSQL
3. Apache/SELinux Plus
4. LAPP/SELinux

Security nightmare in Web systems



Targets of significant security incidents

(Reference: JSOC analysis report of the incursion trend, vol.12, LAC)

- Attacks to web systems have been rapidly increasing.
- It is estimated the cause is growth of web-based commerce.
- Existing security features are really attractive?

LAPP - A typical web application stack

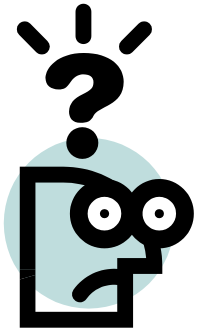
LAPP

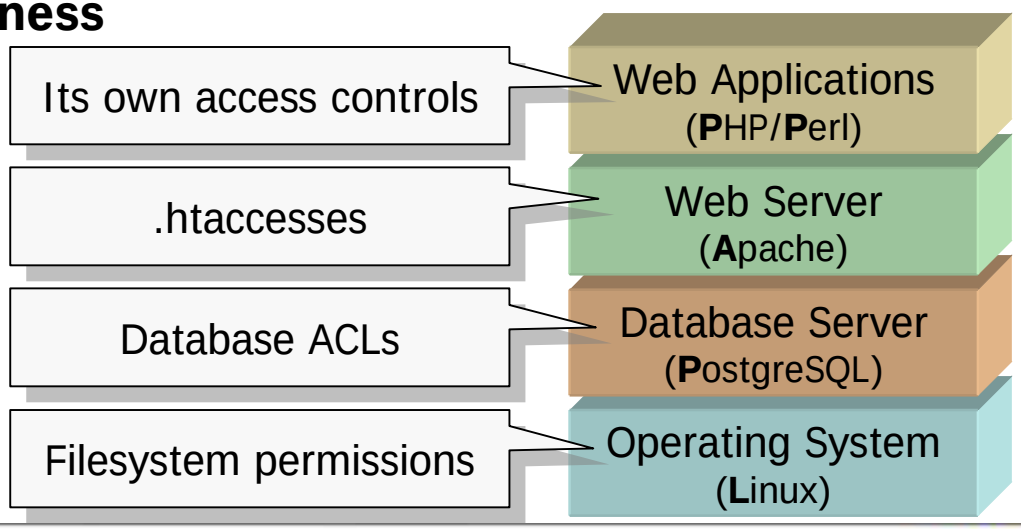
- Full OSS web application stack, also knows as LAMP
  - Linux, Apache, PostgreSQL and PHP/Perl

Security concerns

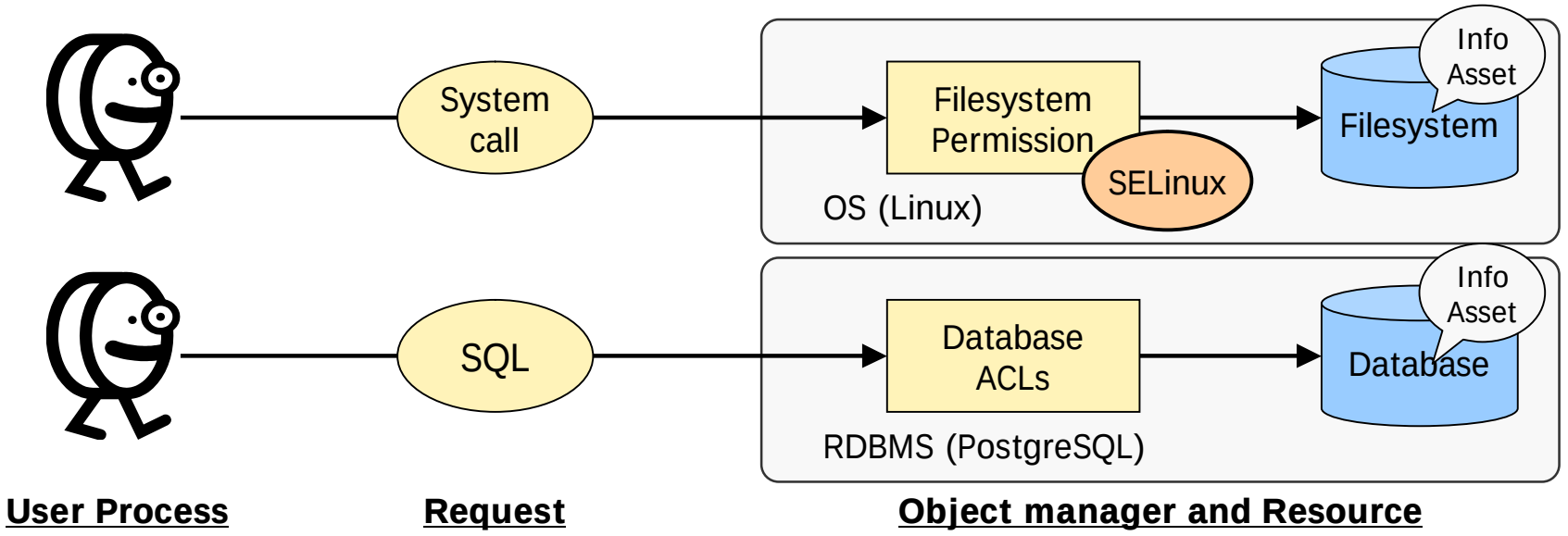
- Each layer has its own access controls
  - Lack of **consistency**
- No individual privileges for users via web-interfaces
  (Security depends on quality of web-applications in other word)
  - Lack of **comprehensiveness**

Keyword ... **Analogy**



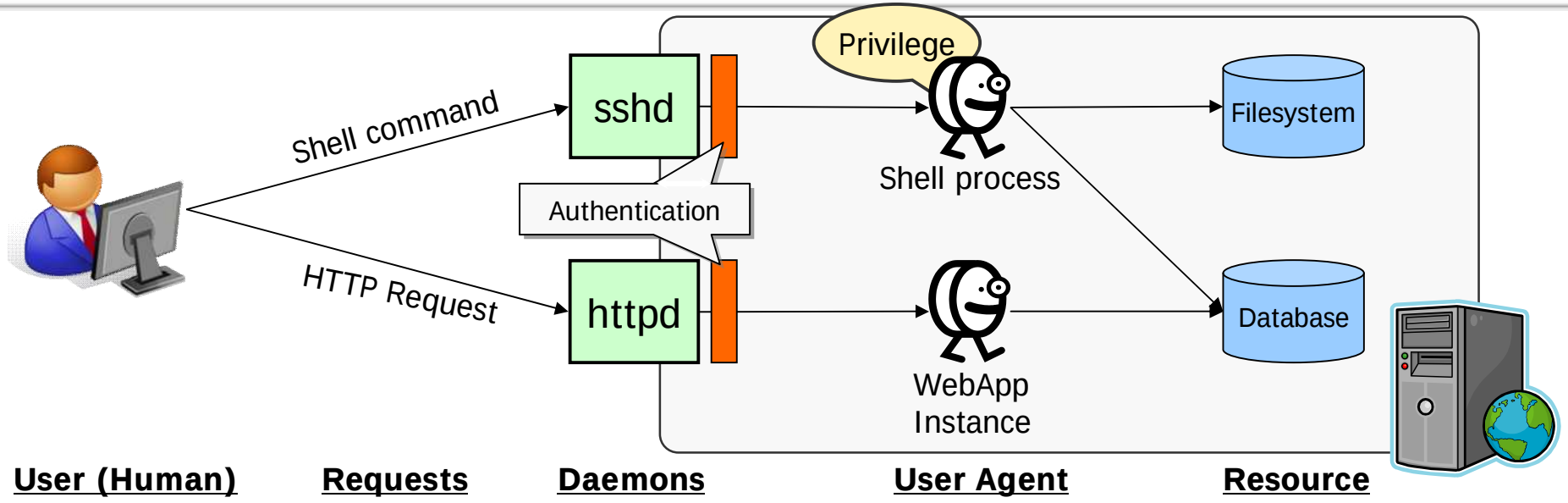


An analogy between OS and Database



- Same relationship in user processes and information assets
- Differences in the way to store and access
  - System call for Filesystem, SQL for Database
- Access control is to decide what are allowed or disallowed on relationships between certain users and resources.
  - No reason why we cannot apply a common security model. It guarantees consistency of access controls.

An analogy between ssh and web



- User is a human; an user agent performs instead of himself.
- The user agent must have correct privilege set of the human.
  - Authentication can identify the human and assign privileges.
  - Httpd launches web-apps without individual privileges.
  - OS/DB cannot distinguish who is behind on the user agent.
- Need to assign privileges of the human user on the web-apps.

What can we find out from the analogies?

SE-PostgreSQL

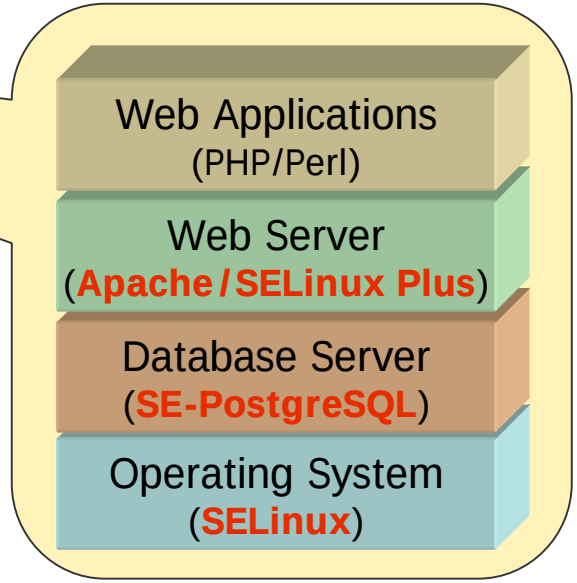
- Advanced access controls for SQL queries based on SELinux
- Consistency in access controls

Apache/SELinux Plus

- Advanced privilege mechanism for web applications based on SELinux
- Comprehensiveness in web-application security

LAPP/SELinux

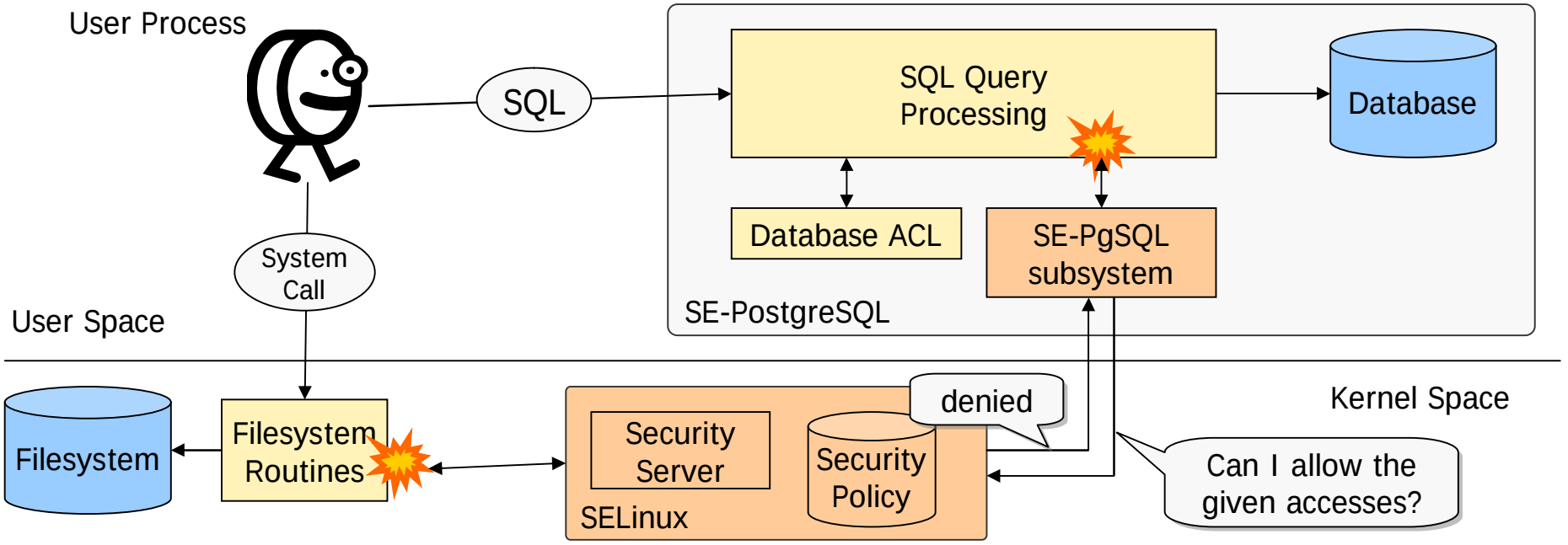
- Utilization of SELinux at the LAPP stack
  - SELinux + SE-PostgreSQL + Apache/SELinux Plus



Empowered by Innovation NEC

1. Background
2. SE-PostgreSQL
3. Apache/SELinux Plus
4. LAPP/SELinux

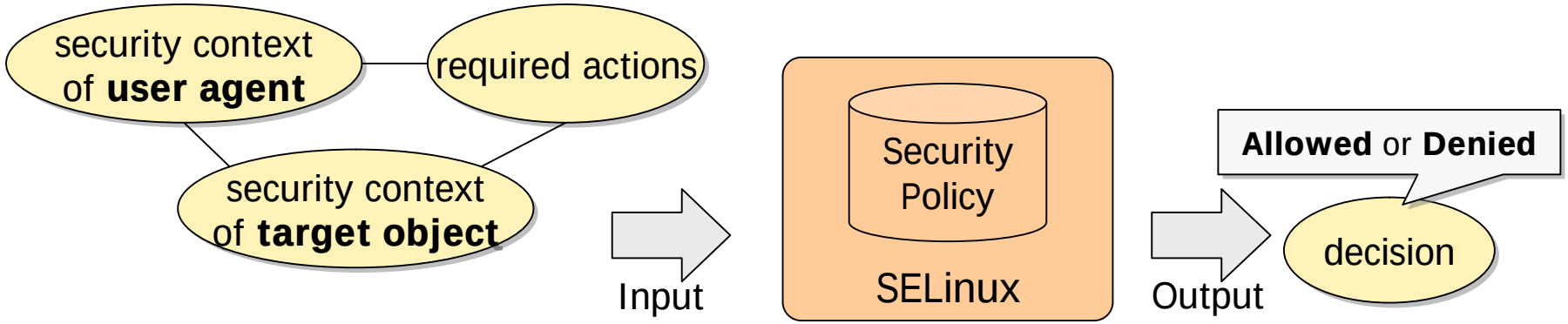
Architecture of SE-PostgreSQL



- SELinux hooks system-call processing
- SE-PostgreSQL also hooks SQL Query processing
- SELinux makes its decision based on its security policy
- SE-PostgreSQL controls execution of the SQL query according to the SELinux's access control decision.

➡ It means the security policy controls both of accesses on OS and DB.

Decision-making in SELinux



SELinux looks like a function

- SELinux returns a binary state for the given arguments.
- Kernel internally gives them to SELinux and follows its decision.
- Userspace application also can utilize this mechanism, as long as it can provide the pair of security contexts.

Security context

- A SELinux specific identifier of processes and any other objects
- The kernel manages security context of the kernel objects.
- Applications must manage security context of the userspace objects.

"security_context" system column

security context of the regular relation

```
postgres=# SELECT security_context, * FROM drink;
            security_context              | id | name  | price
------------------------------------------+----+-------+-------
 system_u:object_r:sepgsql_table_t:s0     |  3 | juice |   130
 system_u:object_r:sepgsql_table_t:s0     |  4 | cofee |   180
 system_u:object_r:sepgsql_table_t:s0:c0  |  5 | beer  |   240
 system_u:object_r:sepgsql_table_t:s0:c0  |  6 | sake  |   320
 system_u:object_r:sepgsql_table_t:s0:c1  |  7 | wine  |   380
 system_u:object_r:sepgsql_table_t:s0:c1  |  8 | tea   |   140
(6 rows)
```

System catalog

security context of the system relation

```
postgres=# SELECT security_context, attname, attnum FROM pg_attribute
        WHERE attrelid = 'drink'::regclass AND attnum > 0;
            security_context              | attname | attnum
------------------------------------------+---------+--------
 system_u:object_r:sepgsql_table_t:s0     | id      |      1
 system_u:object_r:sepgsql_table_t:s0     | name    |      2
 system_u:object_r:sepgsql_ro_table_t:s0  | price   |      3
(3 rows)
```

Privileges of the client

- SE-PostgreSQL applies the security context of peer process.
  - It does NOT depend on database authentication.
  - SELinux provides an API to obtain the security context of peer process.
    - See the **`getpeercon(3)`**
- Labeled IPsec
  - It enables to deliver the security context of remote processes
  - An enhancement of IPsec, available at kernel-2.6.18 or later

```
[ymj@saba ~]$ id -Z
uid=1002(ymj) gid=100(users) groups=100(users) ¥
        context=staff_u:staff_r:staff_t:s0-s0:c0.c15

[ymj@saba ~]$ psql -q postgres -U dbguest
postgres=> SELECT sepgsql_getcon(), current_user;
            sepgsql_getcon             | current_user
---------------------------------------+--------------
 staff_u:staff_r:staff_t:s0-s0:c0.c15 | dbguest
(1 row)
```

Usage of SE-PostgreSQL (1/2)

Row level access control

```
postgres=# SELECT security_context, * from drink;
                  security_context                   | id |  name   | price
-----------------------------------------------------+----+---------+-------
 system_u:object_r:sepgsql_ro_table_t:Unclassified   |  1 | water   |   100
 system_u:object_r:sepgsql_ro_table_t:Unclassified   |  2 | coke    |   120
 system_u:object_r:sepgsql_table_t:Unclassified      |  3 | juice   |   130
 system_u:object_r:sepgsql_table_t:Unclassified      |  4 | coffee  |   180
 system_u:object_r:sepgsql_table_t:Classified        |  5 | beer    |   240
 system_u:object_r:sepgsql_table_t:Classified        |  6 | sake    |   320
 staff_u:object_r:sepgsql_table_t:Unclassified       |  7 | soda    |   150
```

- when SELECT?
  - The Classified tuples are invisible for Unclassified clients.
- when UPDATE/DELETE?
  - It also prevents to update Read-Only (sepgsql_ro_table_t) tuples.
  - But, Classified client can update Read-Writable and Classified tuples.
- when INSERT a tuple?
  - A default security context is assigned on the new tuple.

Usage of SE-PostgreSQL (2/2)

Table/Column level access control

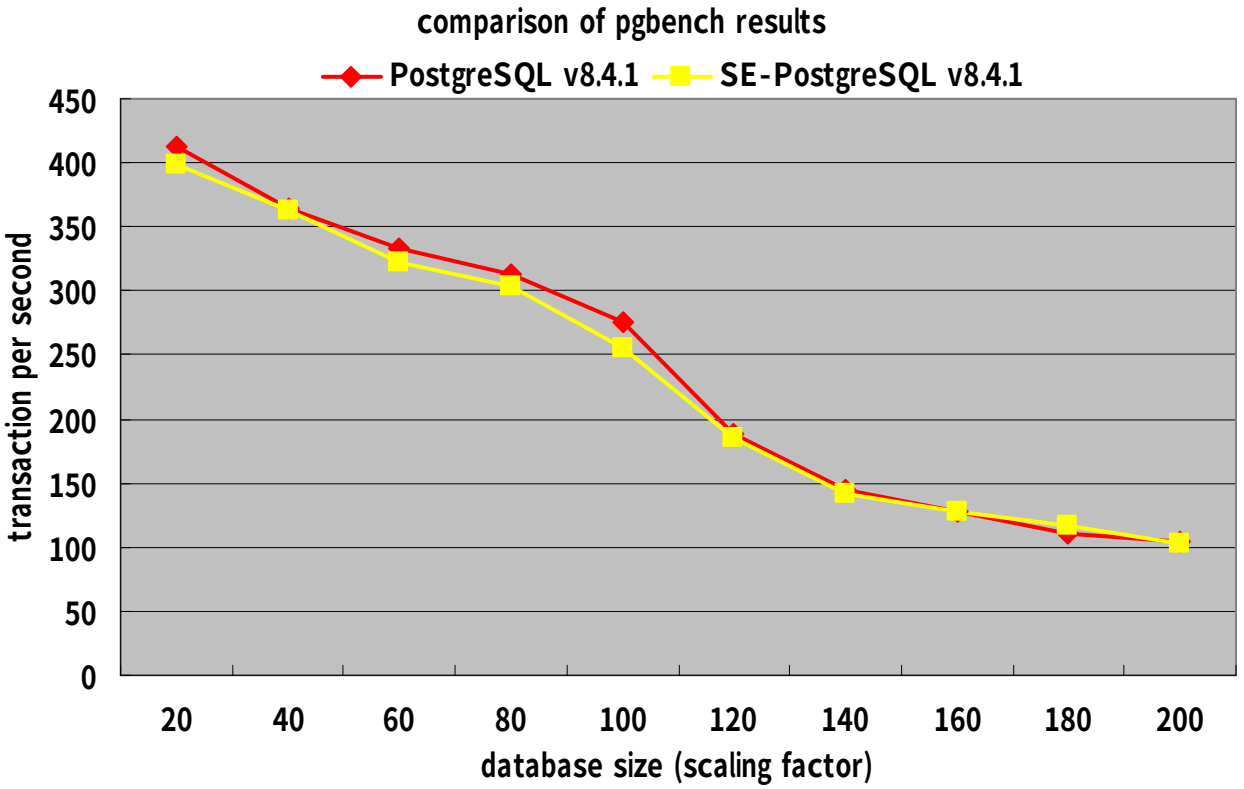
```
postgres=# CREATE TABLE customer (
      cid       integer primary key,
      cname     varchar(32),
      ccredit   varchar(32)
          SECURITY_CONTEXT = 'system_u:object_r:sepgsql_secret_table_t:s0'
      );
CREATE TABLE
```

It allows to assign an individual security context on a certain table/column.

```
postgres=> SELECT * FROM customer;
LOG:  SELinux: denied { select } ¥
    scontext=staff_u:staff_r:staff_t:Unclassified ¥
    tcontext=system_u:object_r:sepgsql_secret_table_t:Unclassified ¥
    tclass=db_column name=customer.ccredit
ERROR:  SELinux: security policy violation
postgres=> SELECT cid, cname FROM customer;
 cid | cname
-----+-------
  10 | jack
  13 | adam
  14 | liza
(3 rows)
```

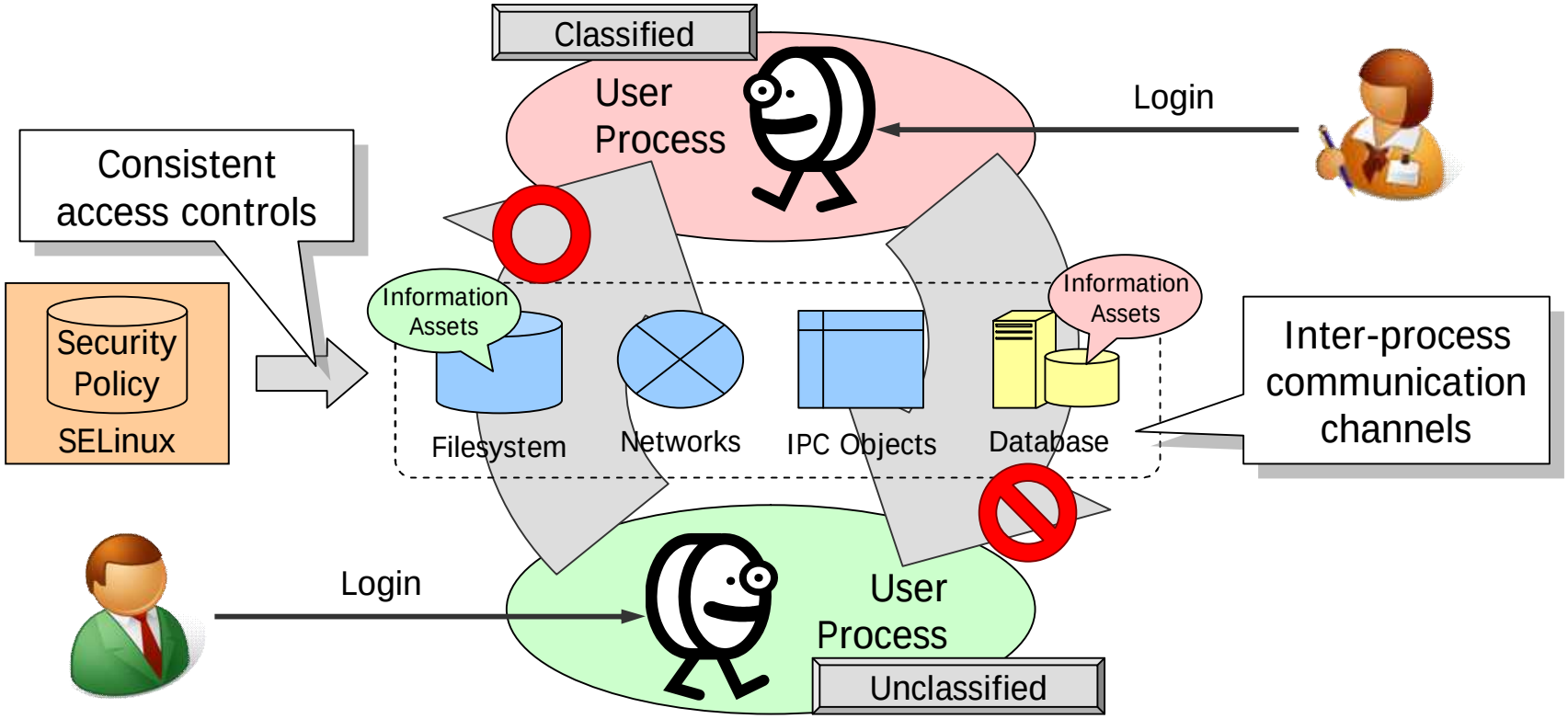
SE-PostgreSQL prevent unprivileged client to access to the column labeled as "Secret".

Performance - SE-PostgreSQL



- 2~4% of trade-off in performance
  - userspace AVC minimizes the number of kernel invocations
- Environments
  - CPU Xeon (2.33GHz) Dual, Mem: 2GB (shared_buffer=512m)
  - measured by **`pgbench -c 2 -t 200000`**

System image: system-wide consistency in access control

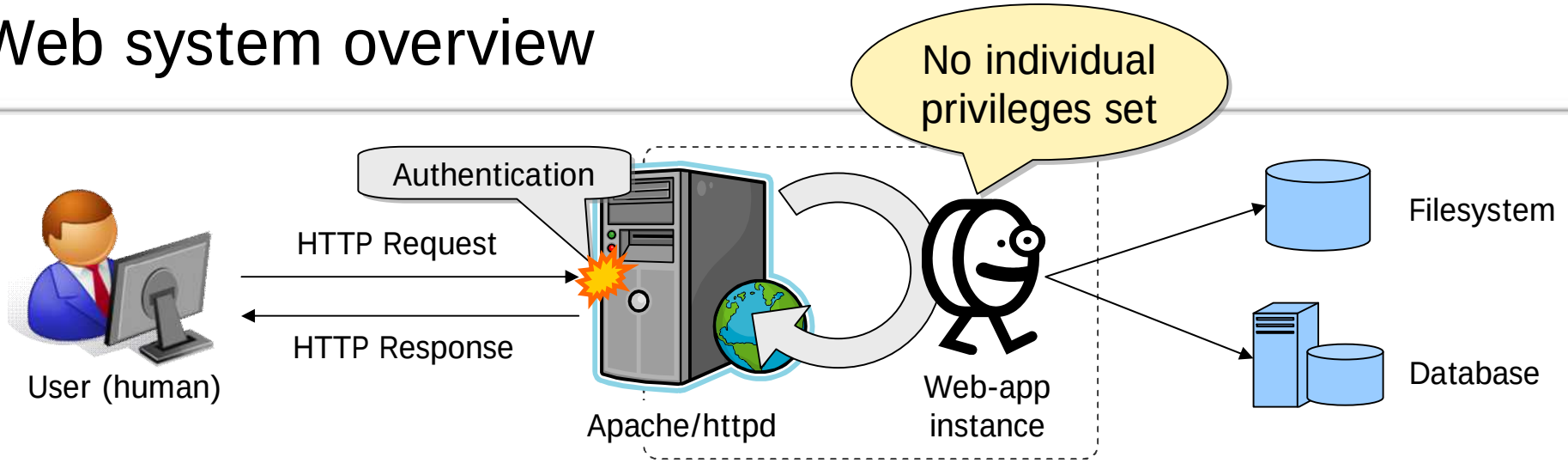


- SELinux controls **ANY** inter-processes communication channels.
  - No read-up, No write-down
- SE-PostgreSQL allows to deploy RDBMS in this scheme.
  - No differences in FS and DB from the viewpoint of access control

Empowered by Innovation NEC

1. Background
2. SE-PostgreSQL
3. Apache/SELinux Plus
4. LAPP/SELinux

Web system overview



Steps to handle user's request

1. User sends HTTP request.
2. Apache/httpd may (not) apply HTTP authentication.
3. It launches a web-app instance which performs as an user agent.
   - But its privilege set is identical to the web-server process.
4. Apache/httpd replies HTTP response.

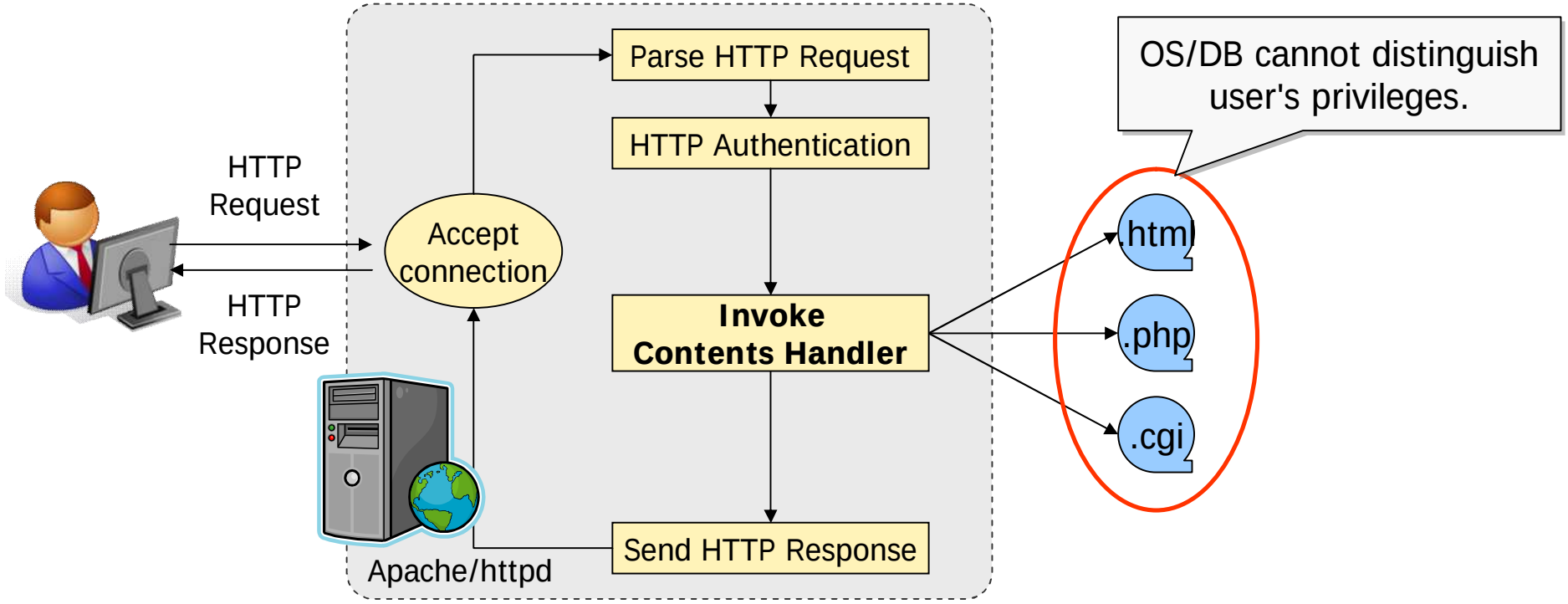
Our headache and prescription

- OS/DB cannot apply valid access controls on user agents.
- Need to assign correct privileges prior to launch web-apps.
- **Apache/SELinux Plus** module does it.

Apache/SELinux Plus (1/2)

Apache without SELinux support

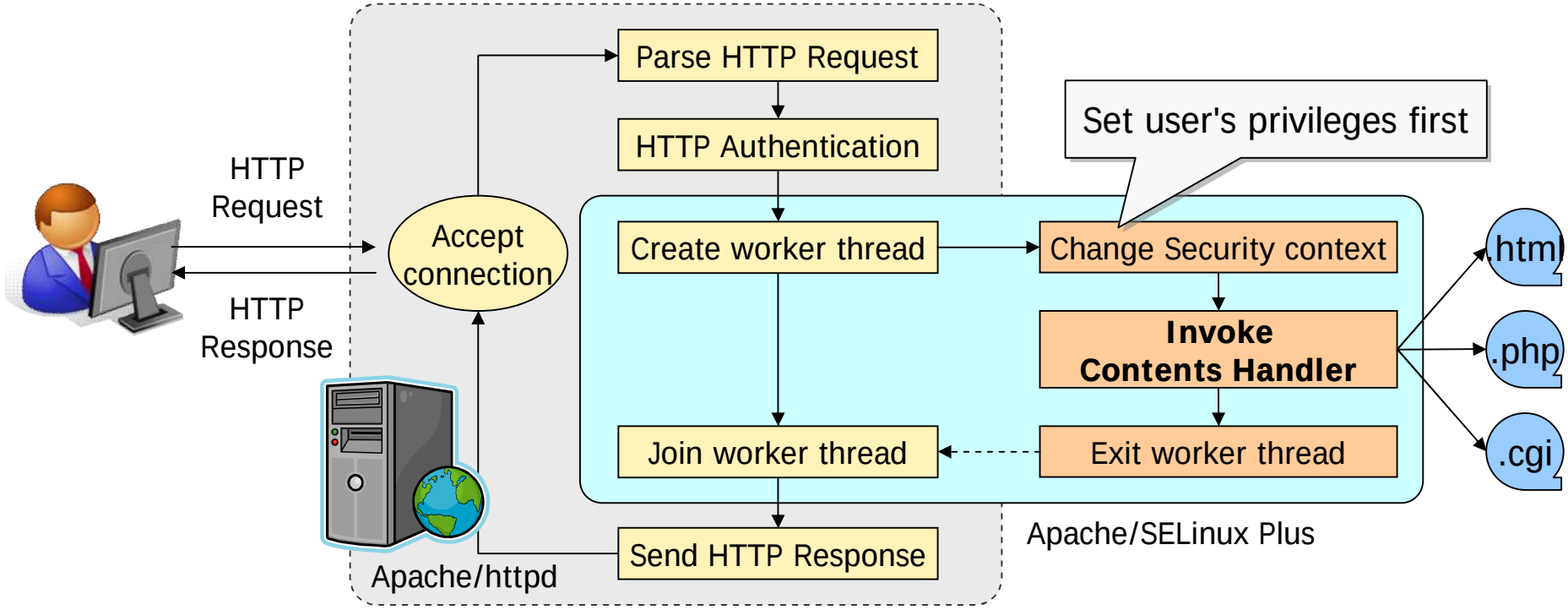
1. HTTP request from users
2. HTTP authentication may be applied
3. Required contents handler is invoked with server process's privileges
   - It works off the burden of access controls to web applications.



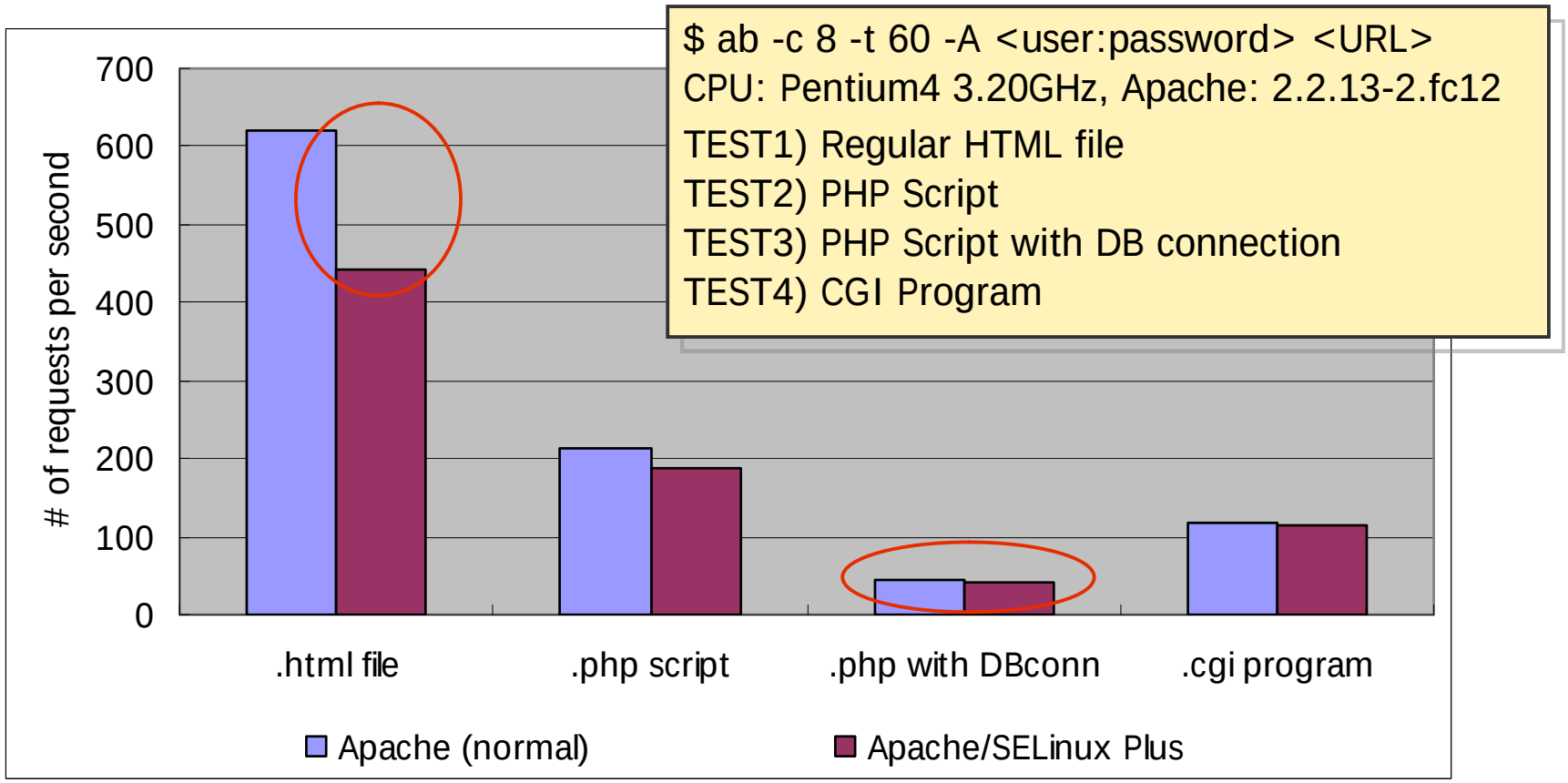
Apache/SELinux Plus (2/2)

Apache/SELinux Plus

1. HTTP request from users
2. HTTP authentication may be applied
3. Creation of one-time worker thread
4. The worker assigns user's privileges on itself, then invokes the handler
   - Web-apps can perform with the least privilege set

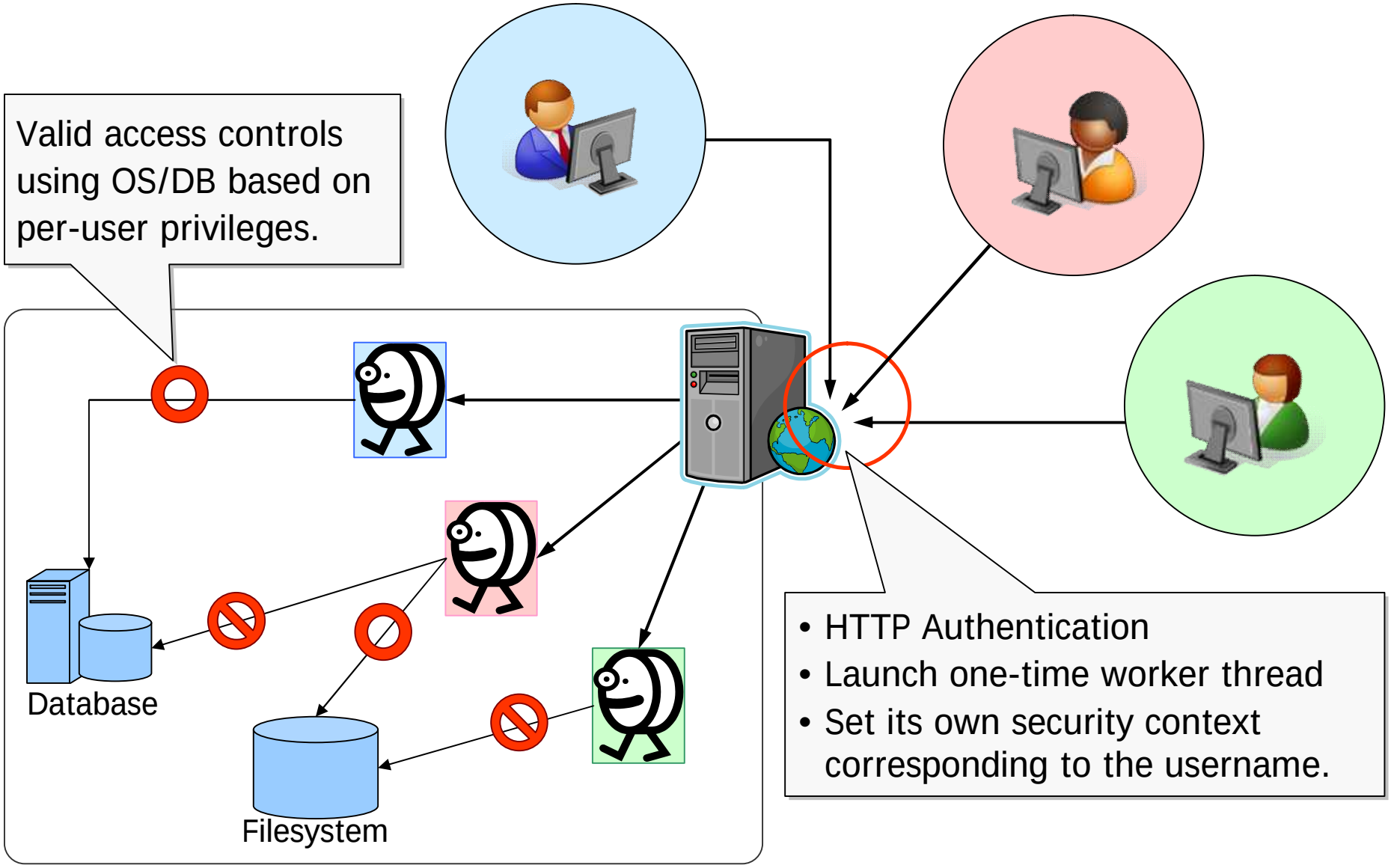


Performance - Apache/SELinux Plus



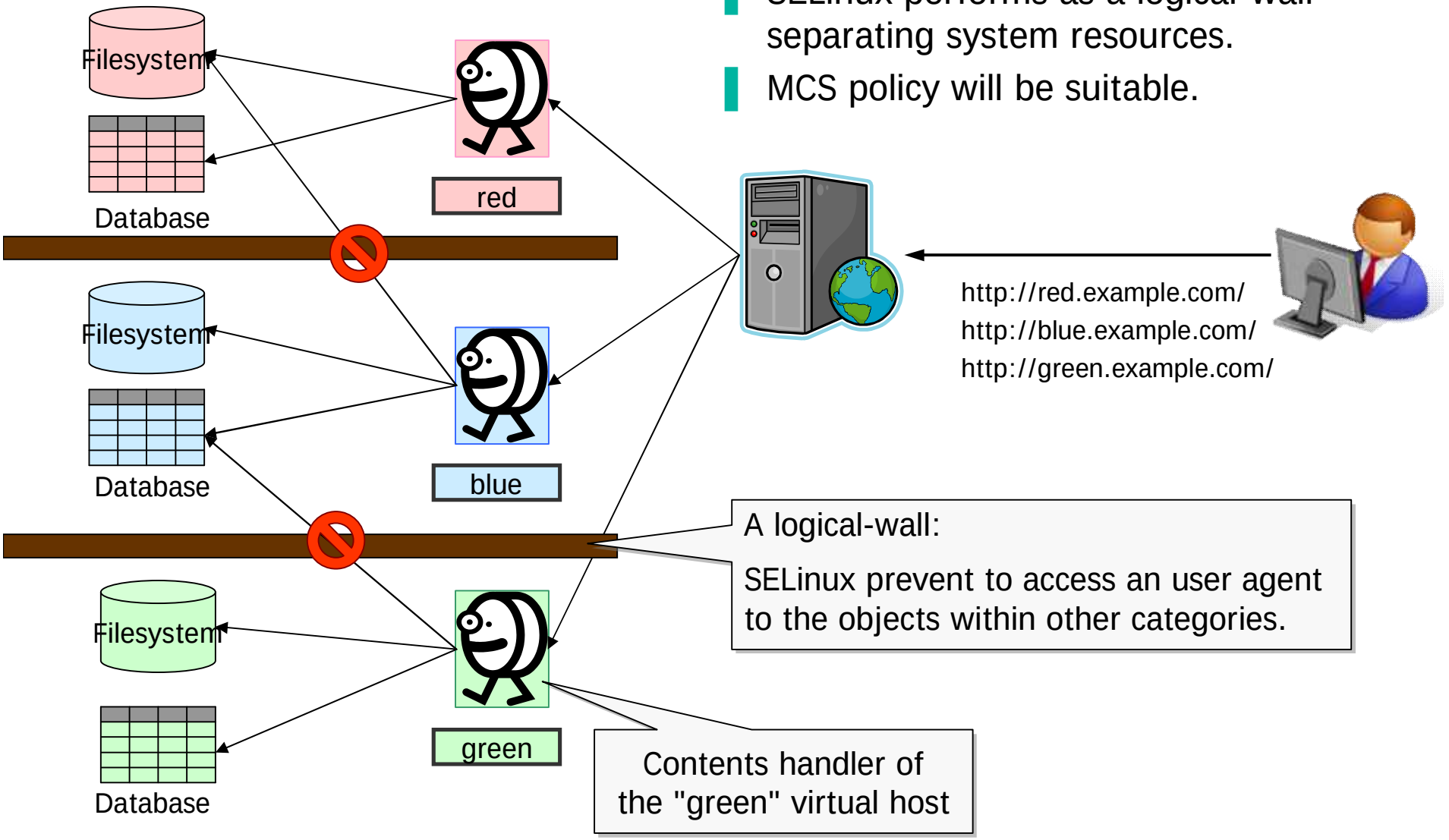
- The cost to assign privileges is relatively large in lightweight request.
- Less differences in our major target (Web+DB applications)
  - Database-queries need higher cost than creation of worker threads.

System image (1/2) : Per web-user privileges



System image (2/2) : Per virtual host separation

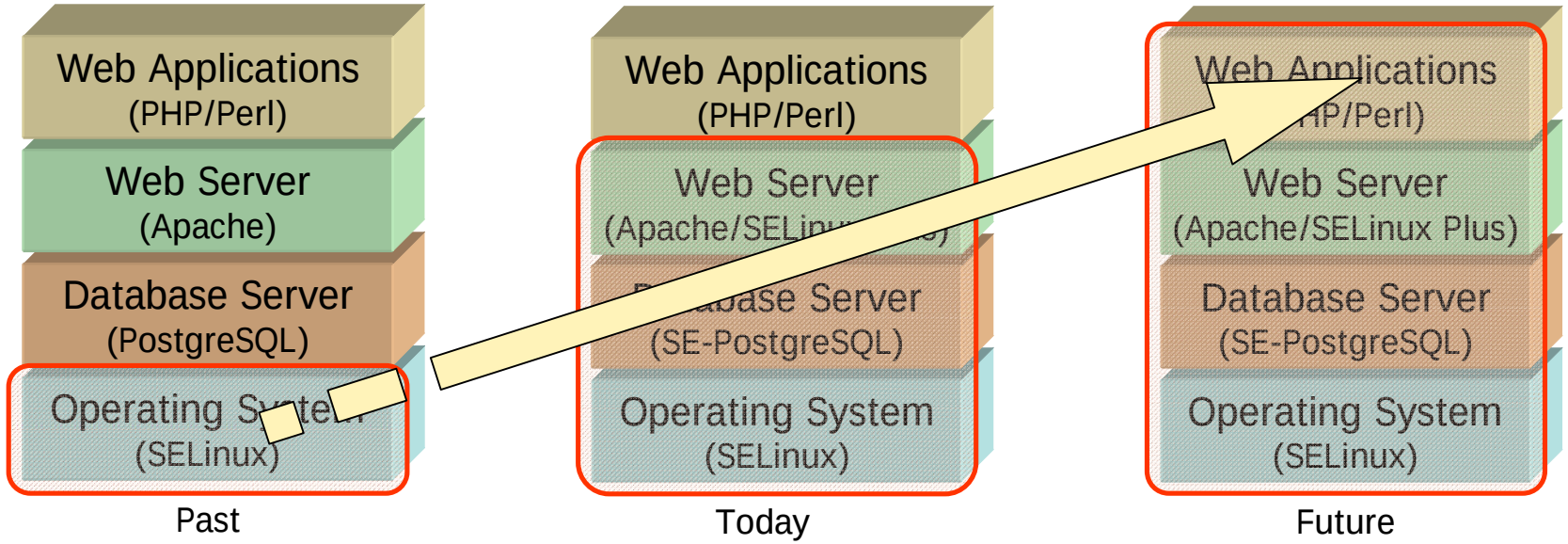
- SELinux performs as a logical-wall separating system resources.
- MCS policy will be suitable.



Empowered by Innovation NEC

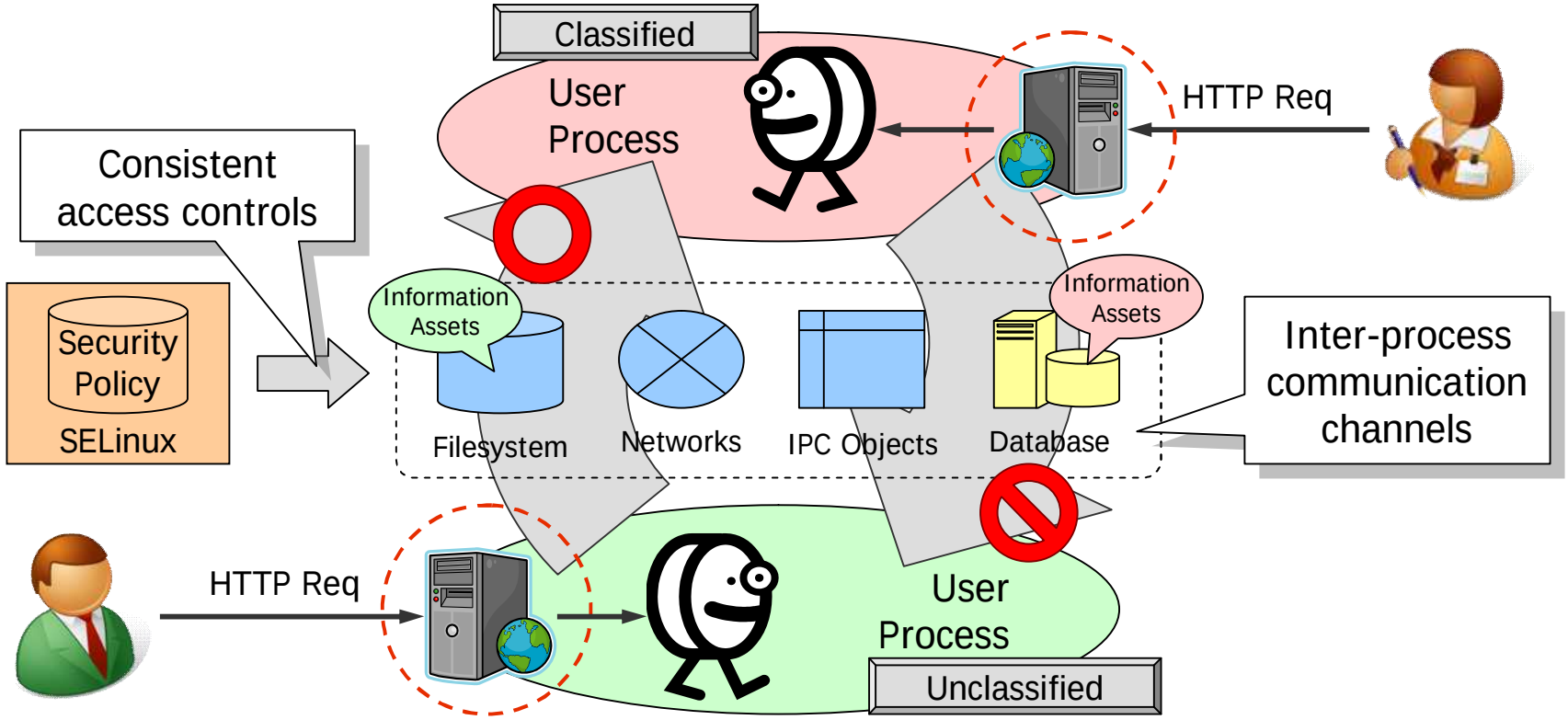
1. Background
2. SE-PostgreSQL
3. Apache/SELinux Plus
4. LAPP/SELinux

SELinux has expanded its coverage



- Prehistory, we have no MAC security.
- God said "let there be SELinux".
  - SELinux applies MAC policy on the operating system.
- Today, it expanded its sphere of life.
  - SE-PostgreSQL, Apache/SELinux Plus, XACE/SELinux, sVirt, ...
- Future, it will cover whole of the web application stack.

Conceptual diagram of LAPP/SELinux

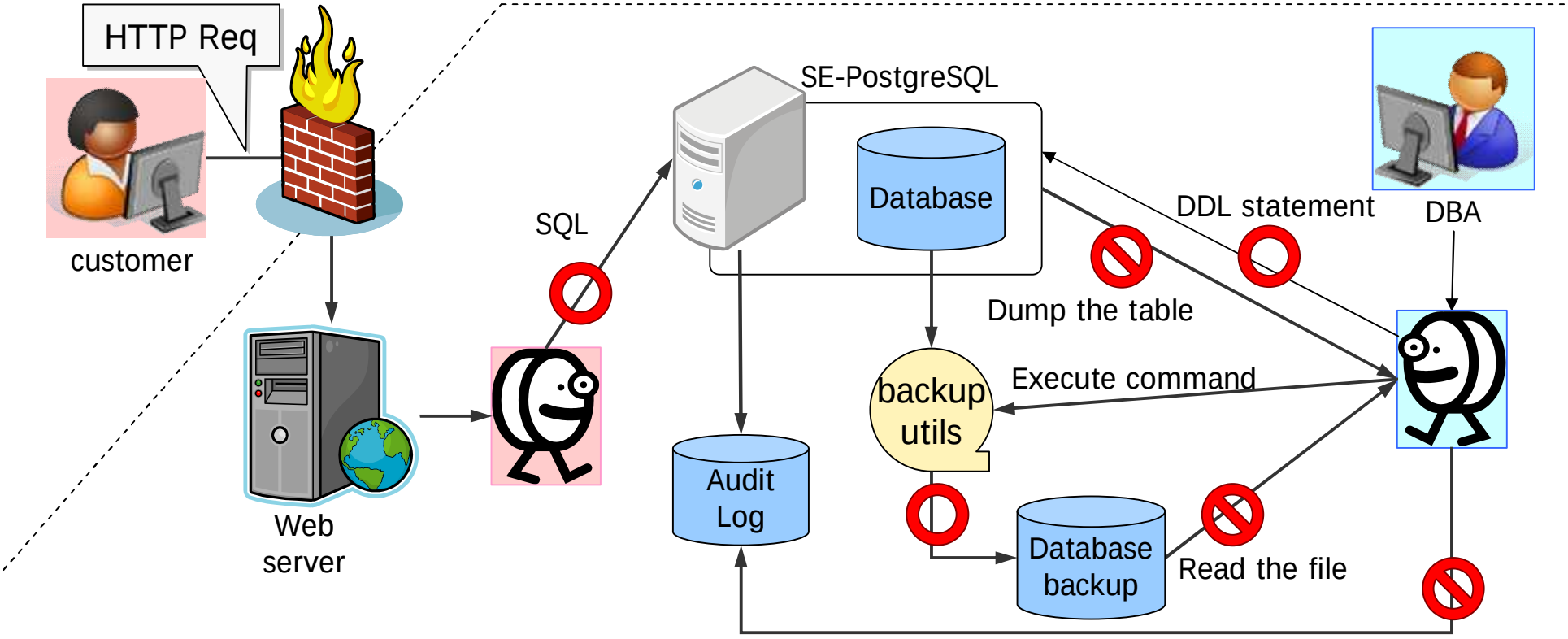


- SE-PostgreSQL provides system-wide consistency in access controls.
- Apache/SELinux plus enables web-apps to perform correct privileges.
- The LAPP/SELinux enables us to set up web systems with **consistent** and **comprehensive** access controls.

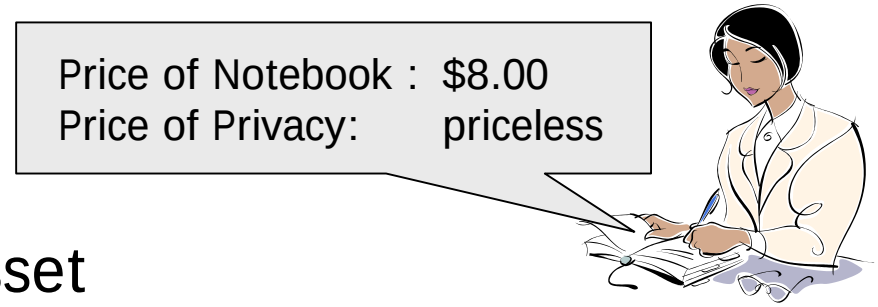
LAPP/SELinux applicability: threats by malicious insider

Database Administrator (DBA)

- In normally, he is allowed anything on databases.
- Need to prevent accesses unnecessary information, including backups.
- Consistent access controls using SELinux, prevent information leaks.



Our principle



Worth of information asset

- It depends on the contents, not the way to store them
- Need to apply consistent access control on the same relationship

Purpose of access controls

- It decides what are allowed and disallowed on the relationship of a certain human-user and information asset
- Again, it is a relationship between a human and information

Principle in LAPP/SELinux

- Common security identifier
- Common access control decision
- Utilization of the platform features in maximum

Project status & history

Status

- Now, kernel supports all the needed features of LAPP/SELinux
- Now, Fedora includes sepostgresql and mod_selinux package
- SE-PostgreSQL is now discussed in the pgsql community

History

- '06/09 launched to develop SE-PostgreSQL
- '07/03 SELinux Symposium & Developer Summit 2007 (Baltimare, USA)
- '07/08 Fedora merged SE-PostgreSQL package (F8 or later)
- '07/11 IPA gave an award due to the development of SE-PostgreSQL
- '08/03 The PostgreSQL conference 2008 (Ottawa, CA)
- '08/05 SE-PostgreSQL was proposed to pgsql-8.4.x development.
- '08/12 Bounds domain feature got merged (2.6.28 or later)
- '09/04 Fedora merged Apache/SELinux Plus (F11 or later)
- '09/07 SE-PostgreSQL was proposed to pgsql-8.5.x development.
- '09/10 Japan Linux Symposium 2009

Any Questions?

Empowered by Innovation NEC

Thank you!