PHAS: A Prefix Hijack Alert System

http://netsec.cs.colostate.edu/phas/

Dan Massey and Yan Chen
Colorado State University

Mohit Lad, Lixia Zhang
UCLA

Beichuan Zhang
University of Arizona

Outline

- Problem and Observations on Solution Space
- RouteViews Based PHAS Service
  - Overview of how it works and what it reports
  - How you can use PHAS now
- Customizing PHAS To Meet Site Requirements
  - How to incorporate local data and detection policies

BGP Origin Hijacking Problem

- BGP Prefix Origin Hijacking
  - Faulty/Malicious AS announces prefix it doesn't own
  - Some sites adopt path and route packets to wrong AS
  - Ex: AS 52 originates to path to 129.82.0.0/16
- If such a hijack does occur, then
  - Some routers select path originating from AS 52
  - Actual origin for 129.82.0.0/16 is AS 12145
  - The router is unlikely to know AS 52 is invalid
    - (and don't add that rule because 129.82/16 may change policy)
  - Legitimate AS 12145 unlikely to see the false path

Related Hijacking Problems

- SubAllocation Hijacking
  - More specific prefix announced by non-owner
  - Packets follow longer match to non-owner
  - Ex: hijack part of 129.82/16 by announcing 129.82.138/24
- Intermediate Path Hijacking (Harder)
  - Announce false links in the AS path to prefix
  - Packets follow AS path that differs from actual path
  - *Note prefix owner should know second to last AS in path*
- BGP routers may see these “bad” events occur, but
  - Can’t easily determine validity without input from owner
  - Owner unlikely to see the “bad” routes

Detecting Hijacks Requires

1. Ability to *see* the “bad” information

   BGP Data Collectors (RouteViews and RIPE)

2. Ability to *distinguish* between “good” and “bad” information

   Prefix owner knows legitimate origin, suballocations, and last hop.

3. Incentive to *fix* the problem if one is found

   Prefix owner is affected directly

PHAS connects data with prefix owners

RouteViews Based PHAS

- Step 1:

  Monitor RouteViews BGP Tables and Updates in (near) Real-Time

- Step 2:

  Keep Database of Origins Used to Reach Each Prefix

- Step 3:

  Report Any Change in Origins Used to Reach the Prefix

- Step 4:

  Owner Applies Local Filter Rules to Determine Significance

Similarly, PHAS tracks changes in SubAllocations and Last Hops (AS adjacent to origin AS)

PHAS Events: Single Peer View

- Monitor a Single Peer's Route To Every Prefix
  - Use initial RIB to determine origin AS for each reachable prefix
  - Monitor AS path in updates and track any change in origin.
  - Log an EVENT if peer changes origin used to reach prefix
- Ex: Monitor Peer 12.0.1.63's Route to 129.82/16
  - Initial route table reports AS path ends in AS 12145
  - Update reports change to new AS path ending AS 52
  - PHAS logs an origin change event (AS 12145 => AS 52)
- Provides Base PHAS Data, But Don't Report Events
  - Vast majority of updates do not change the origin AS
  - But remainder is still a very high volume of event changes.
    - Peer switches between origin AS for a multi-homed prefix
    - Peer loses and regains route to a prefix

Instant Sets: Multiple Peer View

- Instant Origin Set:
  combined set of origins derived from all peers
- Example Origin Set for Prefix 129.82.0.0/16
  - 12.0.1.63 reaches prefix via origin AS 12145
  - 206.186.255.223 reaches prefix via AS 52
  - 144.228.241.81 reaches prefix via AS 12145
  - Instant Origin Set = {12145, 52}
- Instant Set Changes Less Frequently
  - 144.228.241.81 changes to AS 52
  - Instant Origin Set Remains {12145, 52}
- But Instant Origin Set Still Too Dynamic For Reporting

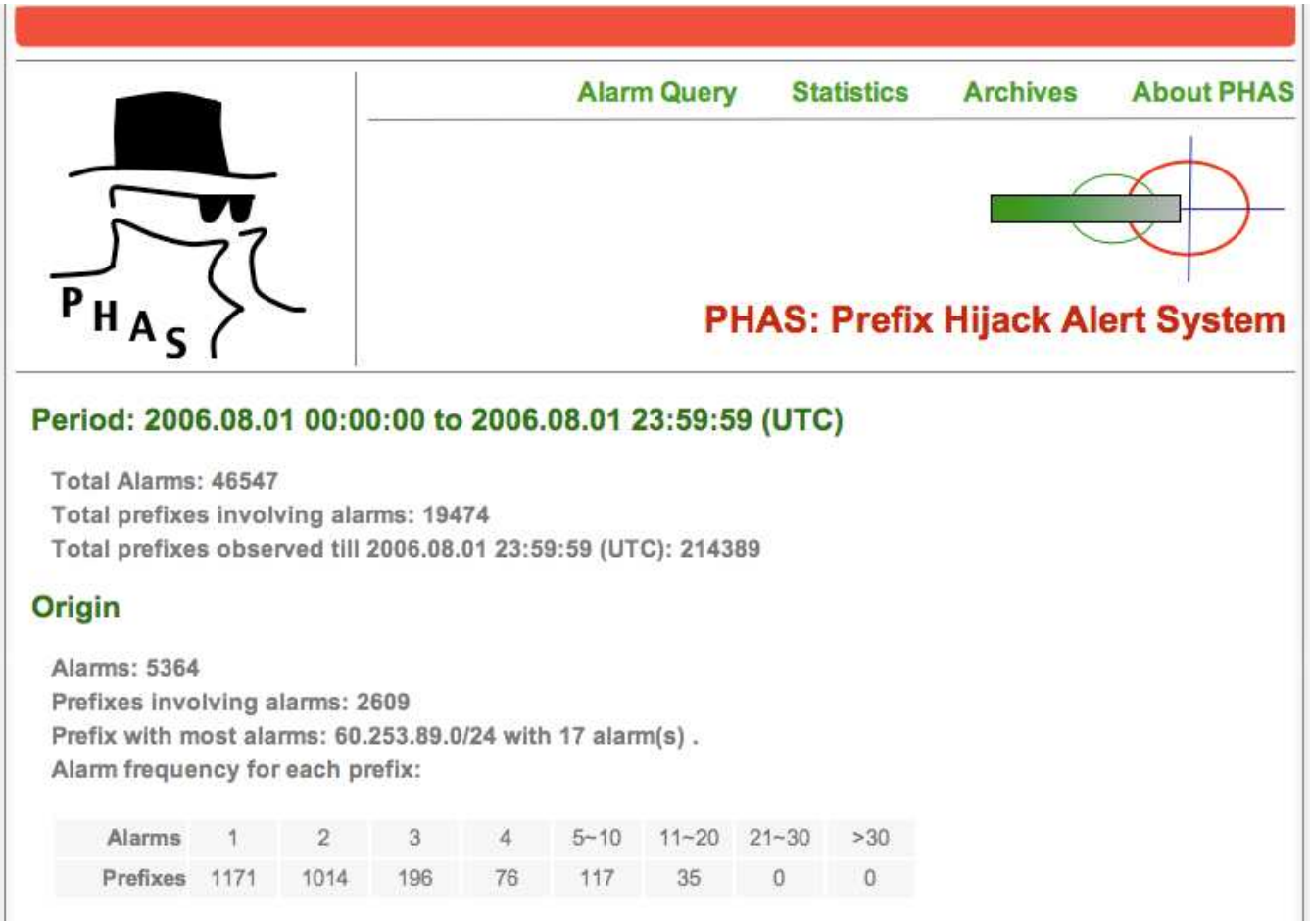
PHAS Notifications

- Instant Origin Set May Still Change Dramatically
  - Most prefixes see no changes in instant origin set
  - Some prefixes see thousands changes per day
  - Origin oscillation results in origin sets of:

    {12145}, {12145,52}, {52}, {12145, 52}, {12145}, ...
- Solution: Apply Basic Dampening To Set
  - Always immediately report any new origin AS (may be hijack)
  - Increase prefix penalty for each set change
  - Based on penalty, delay ***removing*** an origin from the set
  - Dampening removes oscillation and set changes become:

    {12145}, {12145, 52} and remains stable

Resulting PHAS System



Web: http://netsec.cs.colostate.edu/phas/

PHAS-RouteViews Services

- ***Using RouteViews Data to Track Your Prefix***
  - Origins used to reach your prefix and any origin changes
  - Suballocations below your prefix and any changes
  - Last Hop used to reach your prefix and any changes
- PHAS Query Reports Changes in Last 24 Hours
  - *Use Query Link to check on your prefix now*
- PHAS Email Sends Changes in Near Real-Time
  - *Use Subscribe Link to request email notifications*
- PHAS Archive Provides Longer Term Data
  - Useful for pulling more detailed data if an event occurs

Customizing PHAS Notifications

- PHAS Delivers Text Data in a Simple Format:

```
SEQUENCE_NUMBER: 1160417987
TYPE: origin
BGP-UPDATE-TIME: 1160396231
PHAS-DETECT-TIME: 1160414387
PHAS-NOTIFY-TIME: 1160417987
PREFIX: 60.253.29.0/24
SET: 30533
GAINED:
LOST: 33697
```

- Readable By People, ***But Intended for Scripts***

  Script receives notifications and applies local policies

Sample PHAS Notification Filters

- Fixed Set Filtering
  - Configure filter with list of valid origins
  - Filter discards any change within the valid origin set
  - Effective if origin set is well known and relatively static
  - Note this is similar to RIPE MyASN functionality
- Policy Database Filtering
  - Configure filter with policy database (pick your favorite)
  - Filter discards any change within registered origin data
  - Effective if origin set is not directly known, but some other database is trusted
- Planned Support For Common Filters Such as Above
  - Relatively simple to build your own custom filters at any time

More Aggressive Customization

- PHAS Designed Around Three Components:
  PHAS_INPUT, PHAS_TRACKER, PHAS_NOTIFY
- Primary Component is PHAS_TRACKER
  - Expects to receive MRT format messages via TCP
  - Calculates events and instant sets
  - Applies dampening rules based on configuration settings
  - Writes update, instant set, and notification logs
  - Sends notification messages via TCP
- Helper components provide input and process notification
  - You select the input data
  - You determine what to do with the notify messages

Customizing PHAS Input

- PHAS Works With Your Data Source
  - Write (or request PHAS team) build PHAS_INPUT
  - PHAS_INPUT reads your data, places data in MRT format, and sends via TCP to PHAS_TRACKER
- PHAS Input Example
  - PHAS_INPUT_RV obtains data from RouteViews and sends MRT format data to PHAS_TRACKER
  - Building PHAS_INPUT_RIPE
  - Working an ISP to build PHAS_INPUT_ISP that uses their private monitoring system

Customizing PHAS Notifications

- PHAS Provides Your Notification Format
  - Write (or request PHAS team) build PHAS_NOTIFY
  - PHAS_NOTIFY accepts notifications from PHAS_TRACKER via TCP and takes the desired actions
- PHAS Notification Example
  - PHAS_NOTIFY_EMAIL accepts notifications from PHAS_TRACKER, compares notifications against an email list and generates email messages for the interested users
  - Working an ISP to build PHAS_NOTIFY_ISP that applies local rules and forwards notification into ISPs private ops system

PHAS Web Current Status

- Use PHAS Website to Query Your Prefix
  - Website reports last 24 hours of notifications
  - Query link was added to main page in July
- Register An Email Address To Receive Notifications
  - First try a query to see what notifications you might get
  - If you want this data, subscribe your email address
  - Email subscribe link added in past few weeks
  - No known issues in early tests…

PHAS Work in Progress

- Developing and Releasing Email Notification Filters
  - Fixed origin, suballocation, and last lop data
  - Compare PHAS notifications to known databases
- Better Management for Large-Scale Users
  - Code currently working on 190K prefixes
  - Interface works well for sites with small number of prefixes
  - Interface not optimized for user with hundreds of prefixes
- Release PHAS_TRACKER Code
  - Release notification filters to link in policy databases
  - Hardening PHAS_TRACKER for open source public release
  - Move PHAS from research labs to RouteViews
- Seeking feedback on current system and future features

http://netsec.cs.colostate.edu/phas/