Advanced SiLK Analysis
FloCon 2014

Geoffrey Sanders
Tim Shimeall
Modules

SiLK Prefix Maps

YAF Flow Table Timeouts

SiLK Flow Attributes

SiLK Application Labels
SiLK Prefix Maps
Learning Objectives

At the end of this module, analysts will have the knowledge and skills to perform the following tasks:

- Create prefix maps
- Display prefix maps
- Use prefix maps
Contents

Overview of prefix maps
Benefits of prefix maps
Prefix map modes
Common statements
Address mode statements
Protocol-Port mode statements
Creating and displaying prefix maps
Querying prefix maps
Using prefix maps
Overview of Prefix Maps

Commonly referred to as “pmap”

Map field values to text labels
  - IP addresses, ports, and protocols

Binary file created from text input statements
  - `rwpmapbuild(1)`

Flow record operations
  - Partition, sort, count, and display
Benefits of Prefix Maps

Partition flow records using text labels for:

- IP addresses
- Ports and protocols

Display text labels for flow record:

- IP addresses
- Ports and protocols

\[
\text{IP address pmap} + \text{rwfilter} = \text{Flow records that match IP address label}
\]

\[
\text{Protocol-port pmap} + \text{rwcutf} = \text{Flow record protocol-port labels}
\]
Prefix Map Modes

Prefix maps have two modes

Address mode

- Map IPv4/v6 address CIDR or range to text label

Protocol/port mode

- Maps protocol or protocol/port range to text label
Common Statements

Input file statements common to all prefix maps

map-name

- Creates a name for the data in a pmap file
- **map-name simple-string** format
- Simple-string is used to generate filtering switch names (rwfilter) or field names (rwcut, rwgroup, rwsort, rwstats, rwuniq)

label

- Associates a numeric identifier with a text label
- **label num label-text** format
- Must appear before the *default* or range definitions
Common Statements (cont’d)

default

• Provides a default label for ranges not explicitly defined
• \textit{default label-value} format
• UNKNOWN is assigned if a default statement does not appear

mode

• Specifies how to process the pmap file
• \textit{mode \{ ipv4 | ipv6 | proto-port \}} format
Address Mode Statements

Input file statements unique to address prefix maps

cidr-block

- Associate a label or label identifier with a CIDR block
- `cidr-block label-value` format

low-ip high-ip

- Associate a label or label identifier with an IP address range
- `low-ip high-ip label-value` format

low-int high-int

- Treat low-int/high-int as 32 bit values
- Convert values to IPv4 addresses
- Associate label with the converted IPv4 range
- `low-int high-int label-value` format
Protocol-Port Mode Statements

Input file statements unique to protocol-port prefix maps

proto/port

- Associate a label or label identifier with all protocols and port numbers between two inclusive values
- \textit{proto/port proto/port label-value} format

proto

- Associated a label or label identifier with all protocols between two values
- \textit{proto proto label-value} format
Example Protocols/Port Input File

default NONE
mode proto-port
map-name protocols
  1 1 icmp
  6 6 tcp
  17 17 udp
  6/0 6/1023 tcp/generic-reserved
  6/20 6/20 tcp/ftp-data
  6/21 6/21 tcp/ftp
  6/22 6/22 tcp/ssh
Example Address Input File

label 0 non-routable
label 1 internal
label 2 apple
label 3 NONE
default NONE
mode ipv4
map-name networks
0.0.0.0/8 non-routable
172.16.0.0/12 non-routable
192.168.1.0/24 internal
17.0.0.0/8 apple-inc
Creating and Displaying Prefix Maps

Pmaps are created by compiling a text file into a binary pmap using \textit{rwpmapbuild}

\begin{itemize}
  \item \texttt{rwpmapbuild --input-file=protos.pmap.txt --output-file=protos.pmap}
  \item \texttt{rwpmapbuild --input-file=protos.pmap.txt > protos.pmap}
\end{itemize}

Pmaps are displayed by printing their contents using \textit{rwpmapcat}

\begin{itemize}
  \item \texttt{rwpmapcat protos.pmap}
\end{itemize}

Country code prefix maps from GeoIP data are also supported

\begin{itemize}
  \item \texttt{gzip -d -c GeoIP.dat.gz | rwgeoip2ccmap --encoded-input > country_codes.pmap}
\end{itemize}
Querying Prefix Maps

Prefix map keys and values are queried using `rwpmaplookup`

To query the value of a protocol/port

- `echo "6/22" | rwpmaplookup --map-file=protos.pmap`

To query the value of an IP address

- `echo "17.0.0.1" | rwpmaplookup --map-file=networks.pmap`

To query the country code of an IP address

- `echo "2.22.230.1" | rwpmaplookup --country-code`
Using Prefix Maps

Prefix maps can be used with multiple SiLK tools

- rwfilter, rwcut, rwuniq, rwgroup, rwsort, rwstats, rwpmaplookup

**rwfilter**

- `rwfilter --pmap-file=[MAPNAME:]FILENAME [--pmap-src-MAPNAME=LABELS]`
- `rwfilter --pmap-name=protocols:protos.pmap --pmap-src-protocols=tcp/ssh --pass=stdout`

**rwcut**

- `rwcut --pmap-file=[MAPNAME:]FILENAME --fields=fields`
Additional References

Analyst’s Handbook: Using SiLK for Network Traffic Analysis

Manual pages
  • pmapfilter(3), rwcut(1), rwfilter(1), rwgroup(1), rwpmapbuild(1),
    rwpmapcat(1), rwpmaplookup(1), rwsort(1), rwgeoinp2ccmap(1)
Summary

Benefits of prefix maps
Creating prefix maps
Displaying prefix maps
Using prefix maps
Hands-On

Apply the knowledge from this module with use cases in the *SiLK Prefix Maps Workbook*
CERT

YAF Flow Table Timeouts
Learning Objectives

At the end of this module, analysts will have the knowledge and skills to perform the following tasks:

Identify flow table timeouts

Identify how flow table timeouts effect network flow records
Contents

Overview of flow table timeouts
Benefits of flow table timeouts
Idle timeout
Active timeout
Overview of Flow Table Timeouts

Network flow records begin in the YAF sensor ‘flow table’

The flow sensor monitors packets for a five-tuple session and updates the respective flow table entry

Flow table timeouts determine when five-tuple sessions should be finalized (flushed) from the table and written to a flow record

YAF implements two types of flow table timeouts

- Idle
- Active
Benefits of Flow Table Timeouts

Timeouts help to record flow records in the repository
  • Persistent sessions are finalized periodically

Timeouts help the sensor with stateless protocols
  • UDP, ICMP, others
  • Finalize flow records when packets on the wire stop

Timeouts indicate to an analyst packet activity between source and destination IP addresses
  • Packets occurred until a defined period
Idle Timeout

Flow sensor monitors five-tuple session for ‘packet inactivity’

If packets are not seen for a specified time period, the flow record is flushed from the table

- Default time period is 300 seconds (5 minutes)

Idle timeout is configurable (yaf --idle-timeout)
Active Timeout

Flow sensor monitors five-tuple sessions for ‘continued packet activity’

All flow records meeting a specified time period are flushed from the table

- Default time period is 1800 seconds (30 minutes)
- Adds timeout attribute to flow record

New record is created with same five-tuple for new period

Active timeout is configurable (yaf --active-timeout)
Additional References

Analyst’s Handbook: Using SiLK for Network Traffic Analysis

Manual pages
  • yaf(1), rwfilter(1), rwgroup(1), rwmatch(1)
Summary

Benefits of flow table timeouts

Identify flow table timeouts

Identify how flow table timeouts effect network flow records
Hands-On

Apply the knowledge from this module with use cases in the YAF Flow Table Timeouts Workbook
SiLK Flow Attributes
Learning Objectives

At the end of this module, analysts will have the knowledge and skills to perform the following tasks:

- Identify flow record attributes
- Mask flow record attributes
- Partition flow records using attributes
- Display flow record attributes
Contents

Overview of flow attributes
Benefits of flow attributes
Flow attribute values
Unique flow attributes
Flow attribute masks
Partitioning flow records using flow attributes
Displaying flow record attributes
Overview of Flow Attributes

Attributes are a field in SiLK flow records

Describe characteristics of

- Flow record generation
- Packets that comprise a flow record

Attributes are set by the flow sensor - Yet Another Flowmeter (YAF)

Provides analysts with an understanding of what occurred between source and destination IP addresses
Benefits of Flow Attributes

Flow attributes provide analysts insight into sessions between two IP addresses

- Persistent (long-running) sessions
- Sessions with all packets the same size
- Sessions with packets that follow a TCP FIN (excluding ACK packets)
Flow Attribute Field Values

There are five flow attribute field values

- **Null/empty**
  - There are no attributes for the flow record
- **‘S’**
  - All packets for the flow record were the same size
- **‘T’**
  - The flow record reached an initial active timeout
- **‘C’**
  - The flow record was a continuation of an initial active timeout
- **‘F’**
  - Additional packets were seen following a packet with a FIN flag (excluding ACK packets)
  - TCP flows only
Unique Flow Attributes

Persistent (long-lived) sessions have unique combinations of flow attributes

First flow record will have the ‘T’ attribute

Second through next-to-last flow records will have combined ‘TC’ attributes

Last flow record will have the ‘C’ attribute

Five-Tuple Session (4000 seconds)

1800 sec (T)  3600 sec (TC)  4000 sec (C)
Flow Attribute Masks

Attribute ‘masks’ are used to partition flow records based on attribute values

Similar to SiLK TCP flag masks

Masks are defined in an ‘ATTRIBUTES_LIST’

An ATTRIBUTES_LIST is a comma separated list of one or more HIGHATTRIBUTES/MASKATTRIBUTES pairs

```
T/T,TC/TC,C/C,S/S,F/F
```

![Diagram of Flow Attribute Masks]

**MASK**

**ATTRIBUTES_LIST**

```
T/T,TC/TC,C/C,S/S,F/F
```
Example Attribute Masks

Identify flow records with active timeout attributes
  - T/T

Identify flow records where all packets are of equal size
  - S/S

Identify flow records with second to next-to-last active timeouts
  - TC/TC

Identify flow records with a final active timeout
  - C/C

Identify flow records without attributes
  - /SCTF
Partition Flow Records Using Attributes

Flow records are partitioned using *rwfilter* with the `--attributes` option

- `rwfilter --attributes=ATTRIBUTES_LIST`

Example usage

- Partition outweb TCP flow records on 2012/01/01 with initial active timeouts and second through next-to-last active timeouts
- `rwfilter --start-date=2012/01/01 --proto=6 --type=outweb --attributes=T/T,TC/TC`
Sort and Display Flow Record Attributes

Flow records can be sorted and displayed using the ‘attributes’ field of rwuniq, rwsort, rwcut, rwstats, and other rw* tools

- `rwuniq --fields=attributes`
- `rwsort --fields=attr`

Example usage

- Display unique sip, dip, and attribute field bins
  - `rwuniq --fields=sip,dip,attributes`
Additional References

Analyst’s Handbook: Using SiLK for Network Traffic Analysis

Manual pages
  • rwstats(1), rwcut(1), rwfilter(1), rwsort(1), rwgroup(1), rwuniq(1)
Summary

Benefits of flow attributes
Identify flow record attributes
Mask flow record attributes
Partition flow records using attributes
Display flow record attributes
Hands-On

Apply the knowledge from this module with use cases in the 
SiLK Flow Attributes Workbook
SiLK Application Labels
Learning Objectives

At the end of this module, analysts will have the knowledge and skills to perform the following tasks:

Identify application labels

Use application labels

Use the app-mismatch plugin
Overview of Application Labels

Application labels are a numeric field in SiLK flow records

Also referred to as “applabels”

Flow sensors examine packet payloads and tag the flow with an application number

Applabels are set by the flow sensor - Yet Another Flowmeter (YAF)

Provides analysts with an understanding of the application traversing a port

Presently considered ‘experimental’ and may not be 100% accurate
Benefits of Application Labels

Application labels provide analysts insight into an application that traverses a port

- Applications that match the IANA assigned port (match)
- Applications that do not match the assigned port (mismatch)

**Match**

- Client → Port 80: Application 80
- Server

**Mismatch**

- Client ← Port 80: Application 22
- Server
Application Field Values

There are a fixed number of total possible field value

- 0 – 65535 (inclusive)

Common values

- 0 (Undetermined)
- 80 (HTTP)
- 22 (SSH)
- 53 (DNS)
- 194 (IRC)
- 443 (SSL/TLS)
- 65534 (Poison Ivy)
- Many others

Complete list of default values

- http://tools.netsa.cert.org/yaf/applabel.html
Partition Flow Records Using Applabels

Flow records can be partitioned using application labels

- `rwfilter --application=INTEGER_LIST`

Example

- Partition outweb TCP client flow records on 2012/01/01 with SSH and TLS application numbers
- `rwfilter --start-date=2012/01/01 --type=outweb --proto=6 --flags-initial=S/SURFPACE --application=22,443`
Sort and Display Flow Record Applabels

Flow records can be sorted and displayed using the 'applications' field of rwuniq, rwsort, rwcut, rwstats, and other rw* tools

- `rwuniq --fields=application`
- `rwsort --fields=app`

Example usage

- Sort using destination IP, destination port, and application bins
  - `rwsort --fields=dip,dport,app`
SiLK provides the application mismatch plugin

- Used with `rwfilter`
- Identifies flows where the ‘application’ field does not match the source or destination port value
- ‘FAILS’ flow records where the application field value = 0
- ‘FAILS’ flow records that are not TCP or UDP
- `rwfilter --plugin=app-mismatch.so`

Example usage

- Identify out type TCP flow records on 2012/01/01 that do not match destination port 80
- `rwfilter --plugin=app-mismatch.so --start-date=2012/01/01 --proto=6 --type=out --dport=80 --pass=stdout`
Additional References

YAF Application Labeling
  • http://tools.netsa.cert.org/yaf/applabel.html

Manual pages
  • rwstats(1), rwcut(1), rwfilter(1), rwsort(1), rwgroup(1), rwuniq(1), applabel(1)

NetSA Tooltip: Identifying Tunnels Using Application Labels
  • https://tools.netsa.cert.org/confluence/display/tt/Identifying+Tunnels+Using+Application+Labels
Summary

Benefits of application labels
Application label values
Partition flow records using application labels
Displaying application labels
Application Mismatch Plugin
Hands-On

Apply the knowledge from this module with use cases in the SiLK Application Labels Workbook
Contact Information

netsa-contact@cert.org
Geoffrey Sanders – gtsanders@cert.org
Tim Shimeall — tjs@cert.org
Software Engineering Institute
Carnegie Mellon University
Arlington, VA / Pittsburgh, PA