
cisco
You make possible

Troubleshooting Firepower Threat Defense like a TAC Engineer

Kevin Klous, Technical Leader, CX
Justin Roberts, Technical Leader, CX
John Groetzinger, Technical Leader, CX
Foster Lipkey, Technical Leader, CX

TECSEC-3004

cisco Live!
Barcelona | January 27-31, 2020

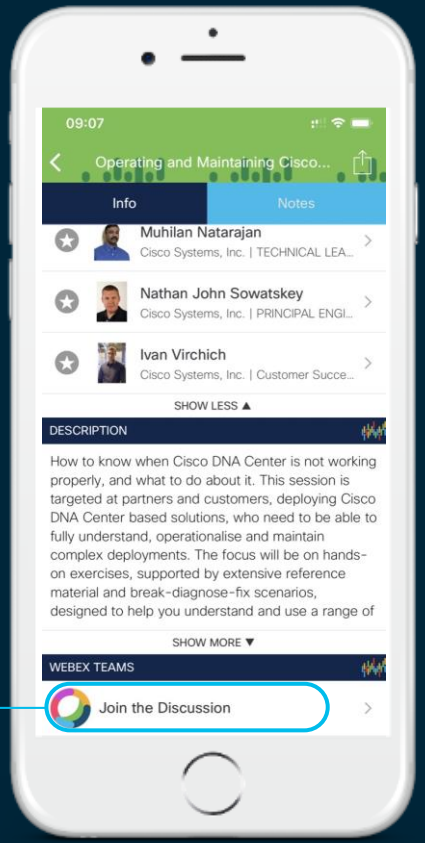
Cisco Webex Teams

Questions?

Use Cisco Webex Teams to chat with the speaker after the session

How

1. Find this session in the Cisco Events Mobile App
2. Click “Join the Discussion”
3. Install Webex Teams or go directly to the team space
4. Enter messages/questions in the team space



Why is FTD troubleshooting so important?

- ASA and Firepower technologies have merged into a unified solution: FTD
- FTD is more complex to troubleshoot; an understanding of both ASA and Firepower technologies is needed.
- Without expertise, there is more risk of network downtime or security breaches. Both are frustrating and impact the business.

Presentation Objectives and Outcomes

- To combat this, today we're going to arm you with knowledge, skills, and tools to more effectively troubleshoot and resolve incidents on the Cisco FTD platform
- We encourage you to think about past or potential future experiences where you can apply these skills

Goal: Fewer late night troubleshooting calls

Agenda

- Introduction
- Architecture Overview
- Path of the Packet
- Troubleshooting Tools
- Interactive Troubleshooting
- Q&A

Abstract

The Cisco Firepower Threat Defense (FTD) next-generation firewall (NGFW) solution combines battle-proven ASA firewall functionality with industry leading Firepower IDS/IPS, malware detection, and content filtering capabilities. Because of this substantial increase in security capacity, a familiarity with both Firepower and ASA technologies is important when troubleshooting the solution. In addition, a proper understanding of platform and datapath architecture is essential in order to properly isolate various components when troubleshooting connectivity issues through an FTD device.

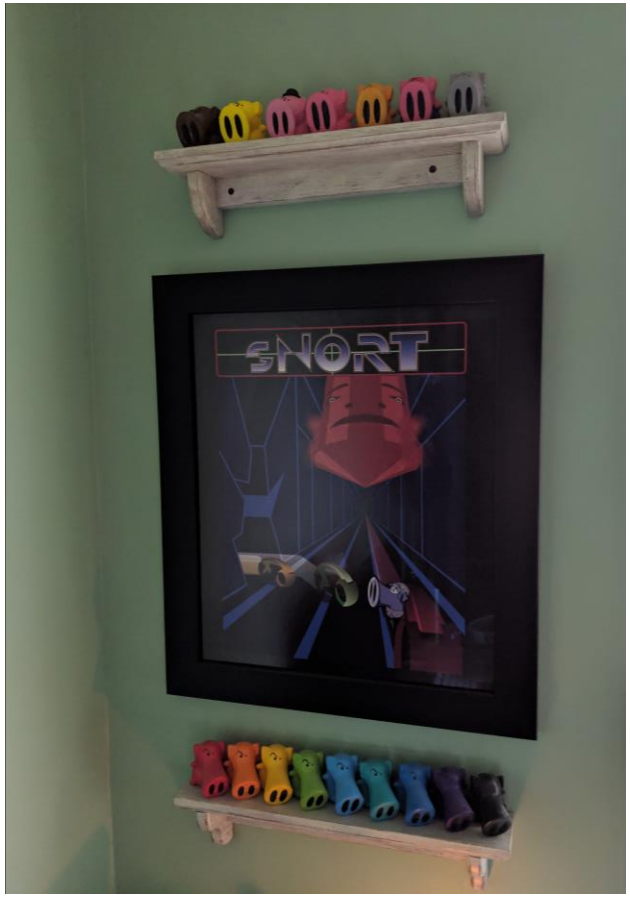
This session will leverage the knowledge of senior engineers from Cisco TAC with both ASA and Firepower backgrounds to instruct participants on how to more effectively troubleshoot the converged FTD platform. The session will primarily focus on FTD architecture, packet flow, and troubleshooting tools. It will also feature live and/or recorded demos along with real-world problem scenarios to help attendees see how they can apply these skills to everyday issues in the field.

The target audience is network security operators who have a working knowledge of ASA and/or Firepower technologies. For those who want to learn more about FTD and how to integrate Cisco NGFW in other Security products, consider attending TECSEC-2600 - Next Generation Firewall Platforms and Integrations.

Your Presenters

Justin Roberts

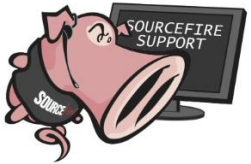
- Technical Leader CX Security
- 5 years in Cisco Firepower TAC
- Before Cisco, Solaris 10/11 Administrator
- Snorty collector
- Python enthusiast



Your Presenters

Foster Lipkey

- Firepower TAC TL
- Snort Expert
- Sourcefire Veteran
- Automation Enthusiast



Foster Lipkey

Your Presenters

John Groetzinger

- Technical Leader for Firepower TAC
- 7+ Years experience with Firepower and Snort
- Original Sourcefire employee
- Network security and Linux enthusiast

Your Presenters

Kevin Klous

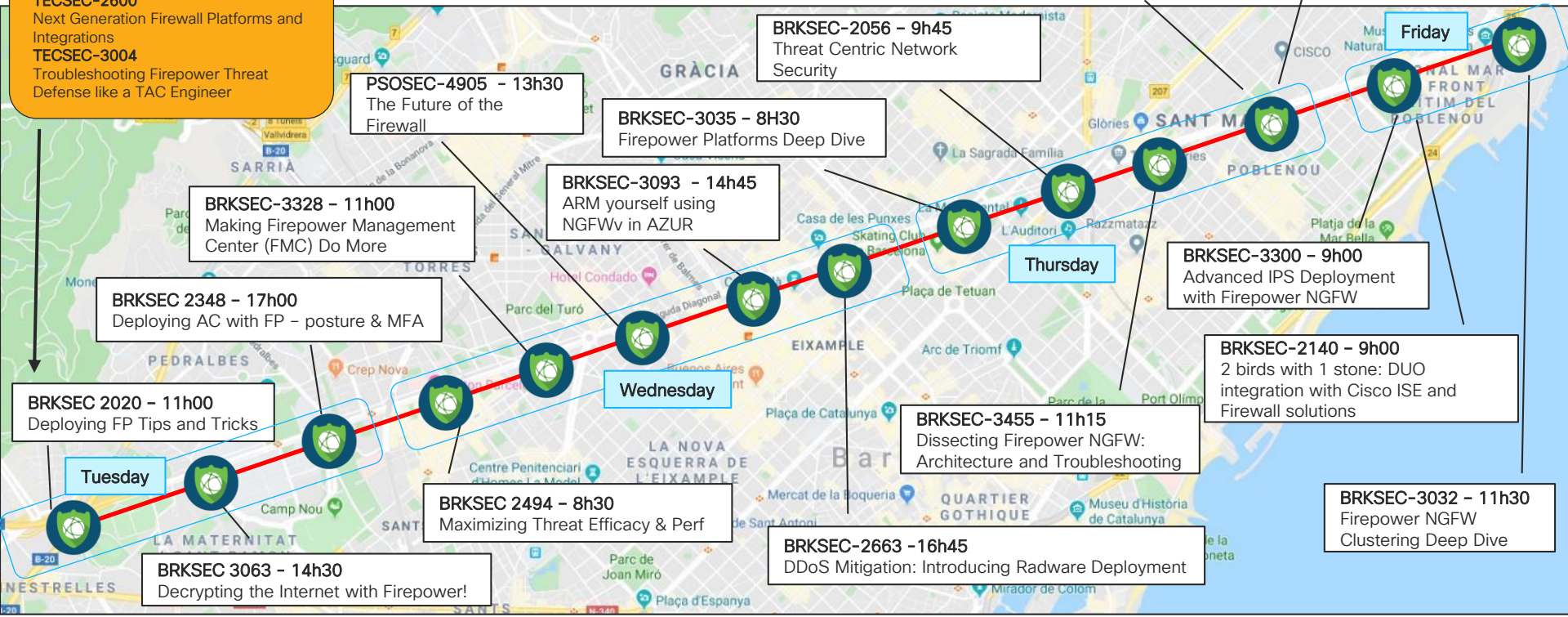
- Focused on Firewalls/NGFW in Cisco TAC since 2012
- Cisco Certified Internetwork Expert (Security – CCIE #43604)
- TAC Security Podcast host & panelist
- Pursuing M.S. in I.S. Engineering – Cybersecurity at JHU
- Serves as a Spanish translator for Guatemala missions

Introduction

Introduction – Presentation Focus Areas

- This is not an introductory session! General familiarity with either ASA or Firepower is assumed. If you do not have knowledge of the product you may want to consider attending TECSEC-2600 instead.
- Other Cisco Live presentations cover FTD features, design, deployment, and configuration. We are focused on product functionality and troubleshooting.
- Configuration and troubleshooting of the FXOS platform is out of scope although it will be referenced as needed.

Firepower Diagonal Learning Map

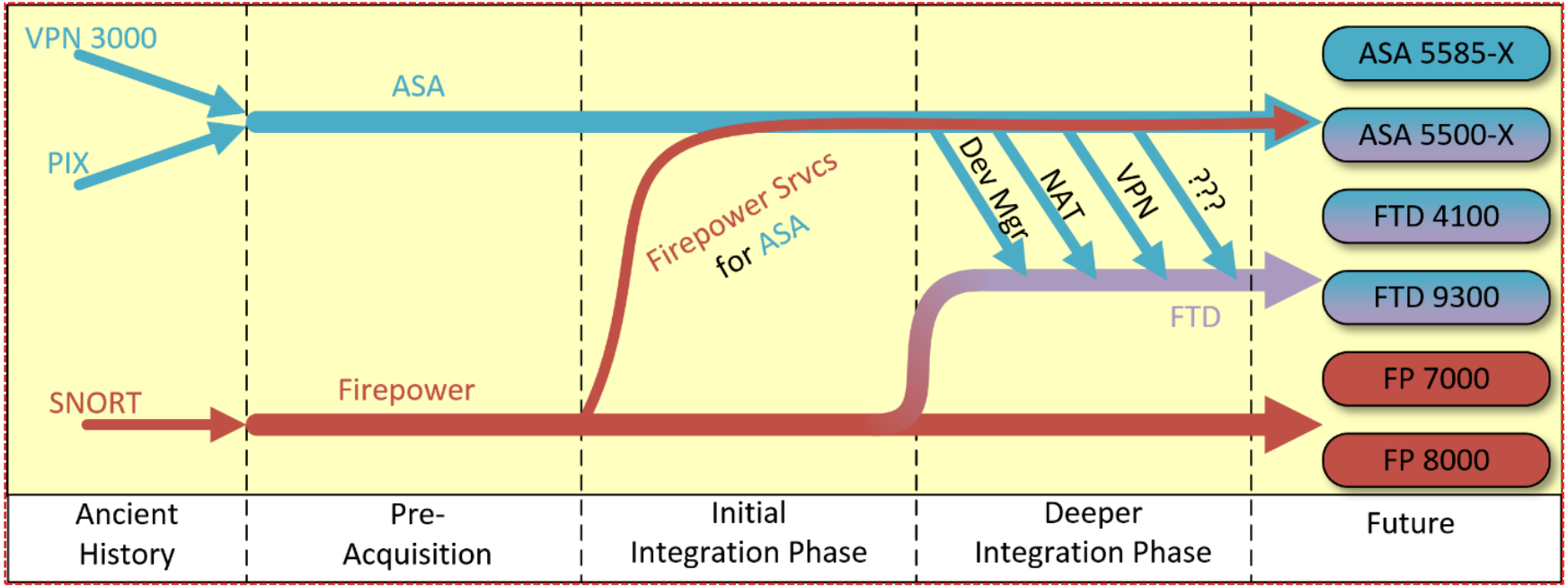


Introduction – Key Terminology

These terms are within the context of Firepower Threat Defense.

Term	Definition
Lina	Underlying ASA-derived process that is integrated into the FTD product
Snort	Components of the Firepower product integrated into FTD
FMC	Firepower Management Center – Off-box GUI used to manage FTD devices (Configuration, reporting, monitoring, etc.). Formerly the Firesight Management Center or Defense Center.
FDM	Firepower Device Manager – Web-based, on-box management option for low to mid-range platforms
FXOS	Firepower Extensible Operating System – System that manages the hardware platforms for Firepower 9300, 4100, and 2100 series products
FCM	Firepower Chassis Manager – On-box GUI used to manage FXOS platforms (Logical device configuration, interface assignments, monitoring, etc.)

NGFW evolution



Introduction – How did we get here?

2005

ASA 7.x introduced

2013

Cisco acquired Sourcefire on October 7, 2013

SOURCEfire®

Sourcefire is now part of Cisco.

2014

ASA w/ Firepower Services replaced ASA w/ CX

ASA 9.2(2)4+

Firepower 5.3.1+

2016

Firepower Threat Defense 6.0.1 introduced as integrated solution

Hardware Platforms:

ASA 5500 Series

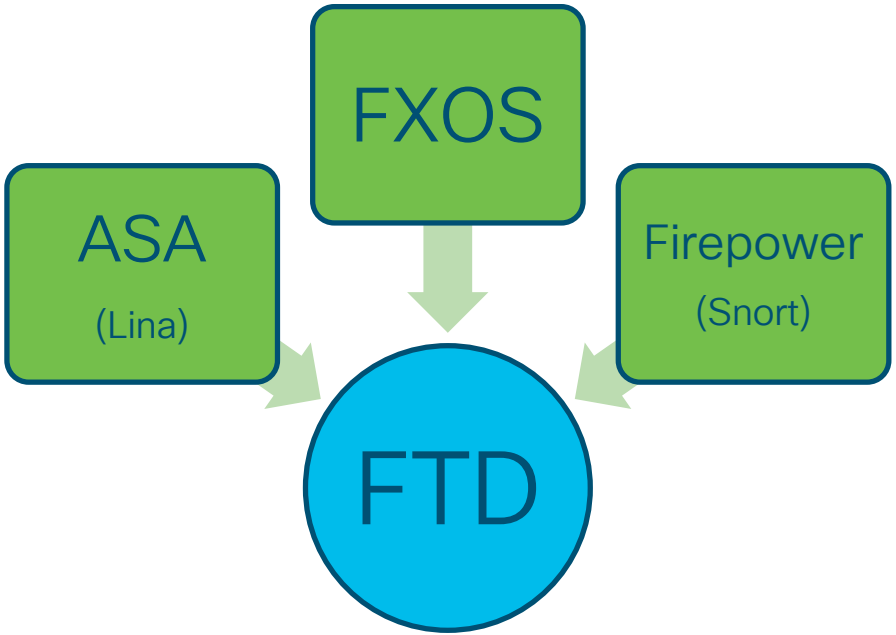
ASA 5500-X Series

ASA 5500-X, Firepower 2100, 4100, 9300 Series

Architecture Overview:
Software Functions

Introduction – What is Firepower Threat Defense?

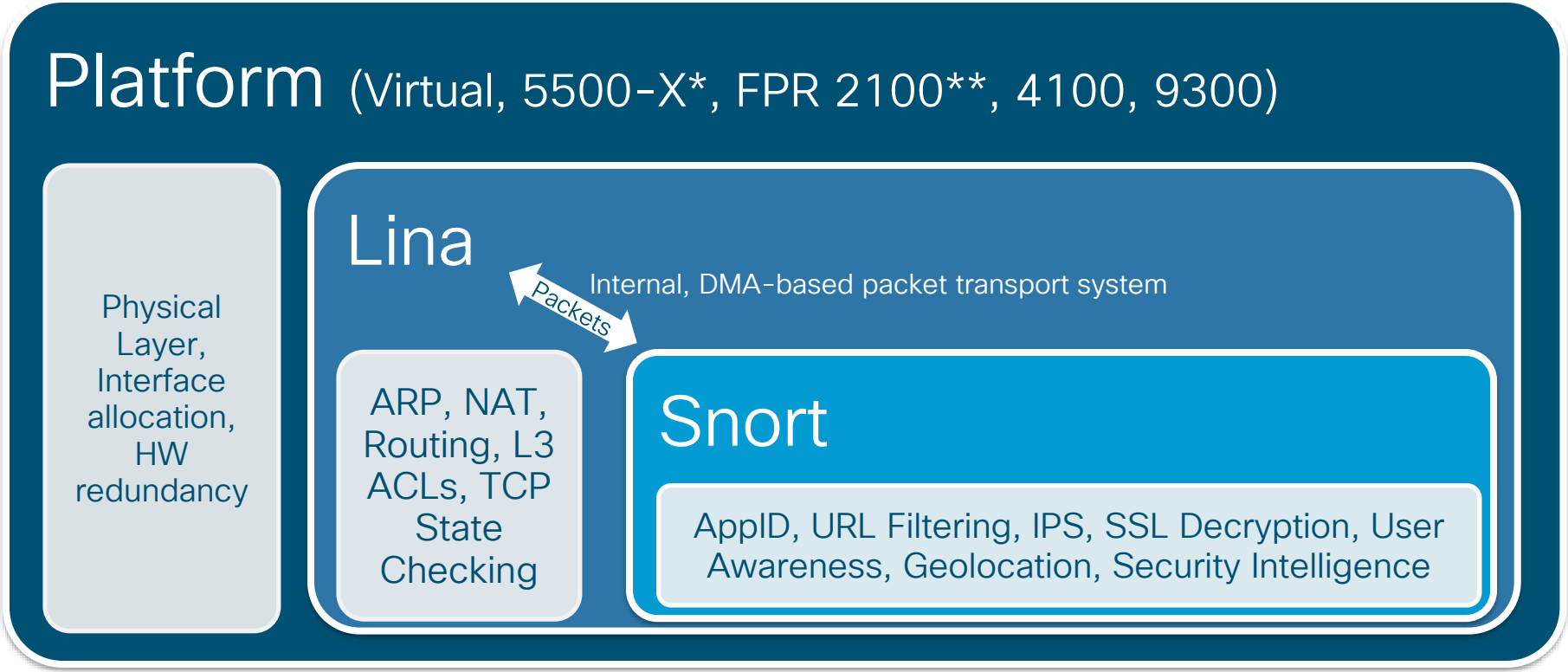
- ASA and Firepower functionality wrapped into a single, unified image
- All processes run within single operating system
- Latest hardware platforms introduce Firepower Extensible Operating System (FXOS) as platform layer beneath the FTD application



Functional Overview – A Layered Approach

OSI Layer	Component	Examples
L1 - Physical	FXOS, 5500-X, Virtual platforms	Interface allocation, L1 configuration
L2 - Data Link	Lina (FXOS handles LACP on Firepower platforms - 2100, 4100, 9300)	Interface MAC Addressing, ARP
L3 - Network	Lina	IP Address assignment, Routing, NAT
L4 - Transport	Lina	TCP State checking, L4 ACLs
L5-7 - Session, Presentation, and Application Layers	Snort (Lina L7 inspection via MPF)	AppID, URL Filtering, IPS, SSL Decryption, User Awareness

Firepower Threat Defense – Functional Diagram

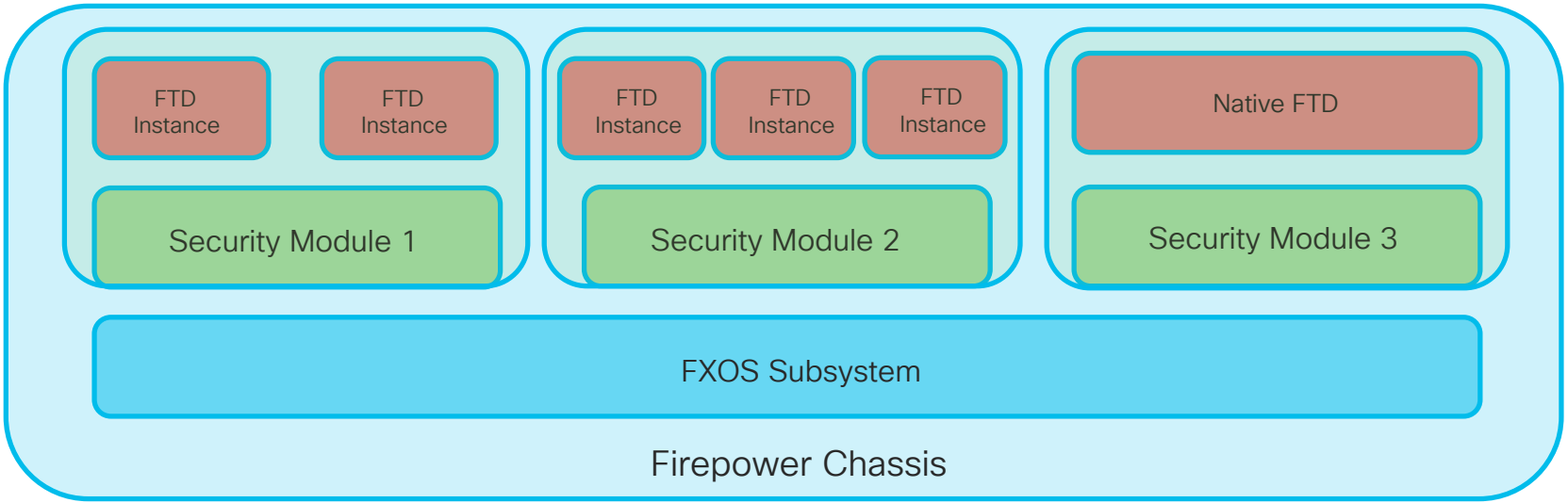


Multi-Instance FTD on FXOS Platforms

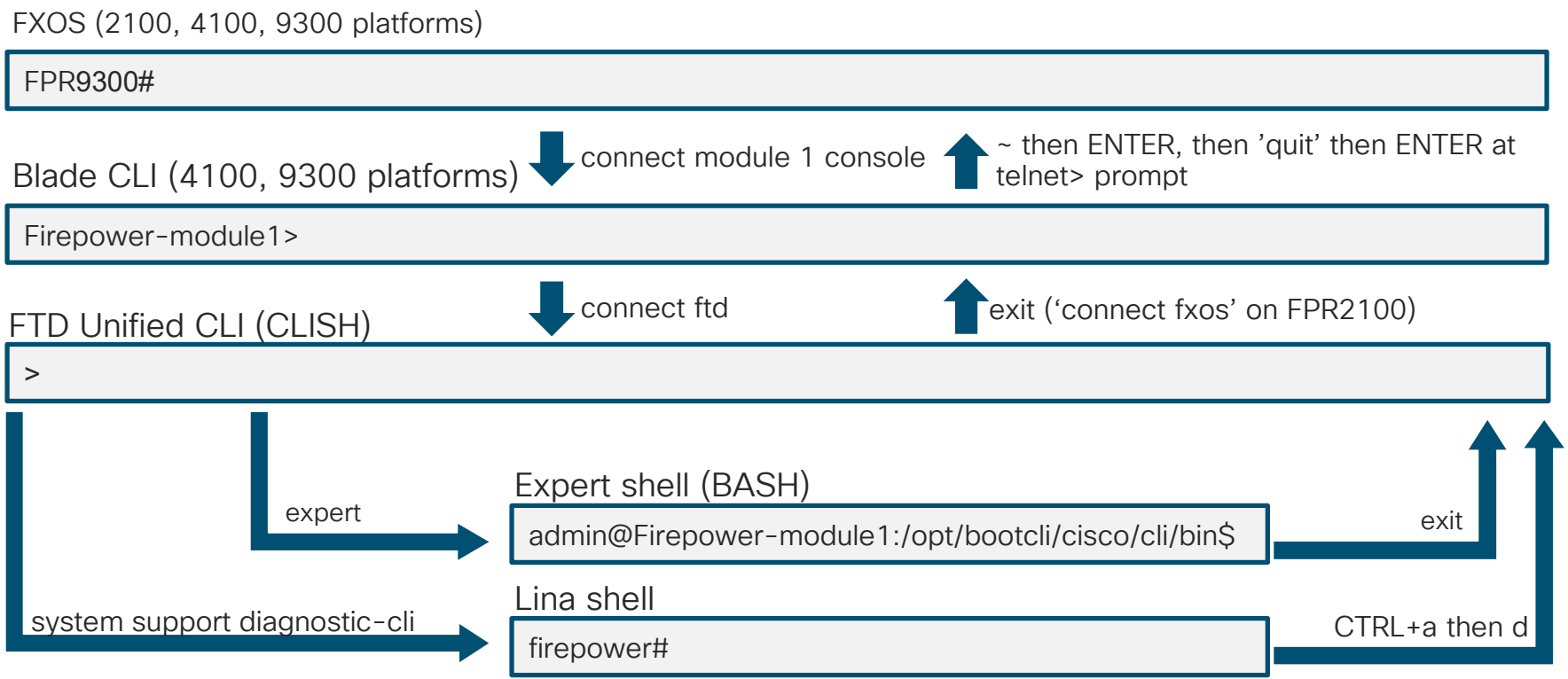
- MI feature was released in FTD 6.3 (December 2018)
- Similar to ASA multi-context feature but implementation is different:
  - Docker container instances instead of a single, partitioned application provides better tenant separation
  - Enables reboot/upgrade of individual instances without affecting other instances (FTD version of instances can be different)
  - Improved hardware resource separation since each instance has its own dedicated CPU cores, disk space, and memory
  - Instances sizes can be changed according to throughput/resource requirements

Multi-Instance Architecture

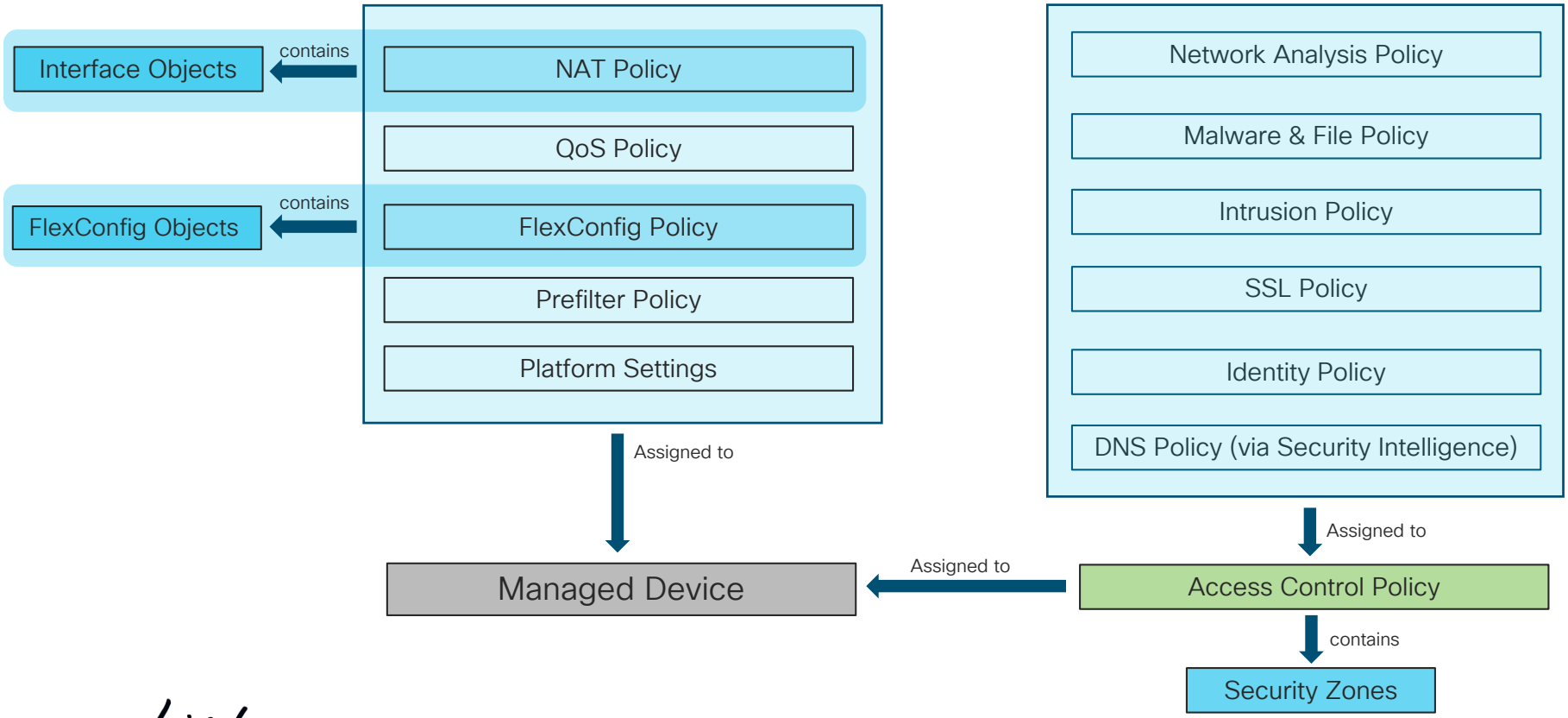
Firepower 9300 Example:



FTD – Navigating between the CLIs

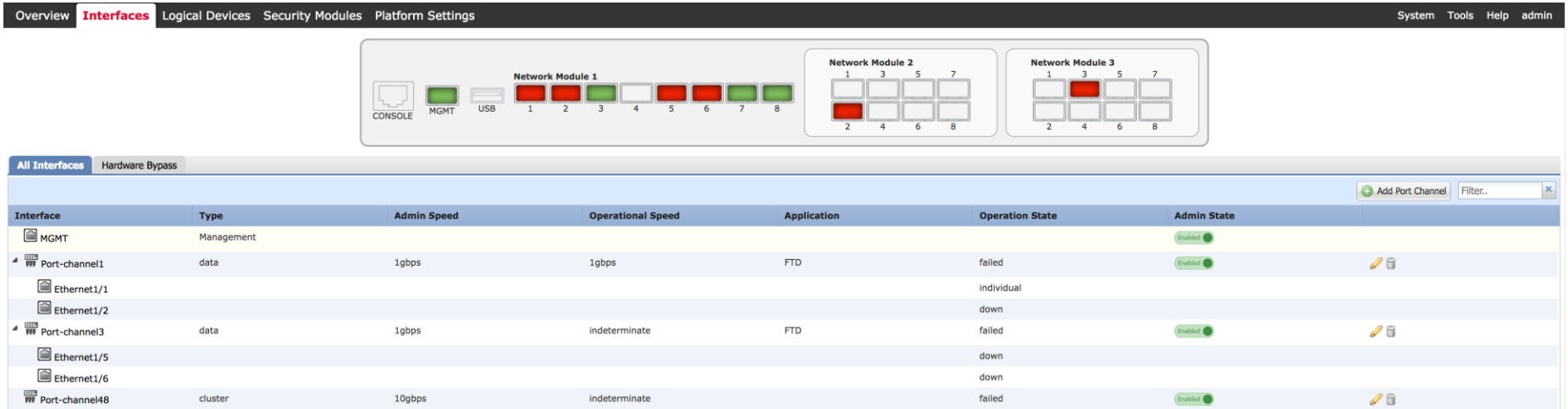


FMC – Object Relationship Diagram



Functional Overview – Physical Layer (L1)

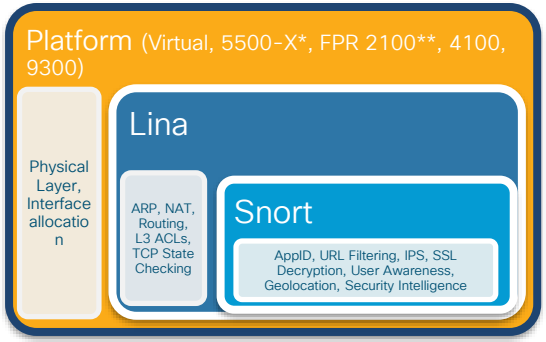
On FXOS platforms, interface allocation is handled via Firepower Chassis Manager (FCM) or the FXOS CLI. FCM example:



Functional Overview – Physical Layer (L1)

Performing interface allocation in FXOS CLI:

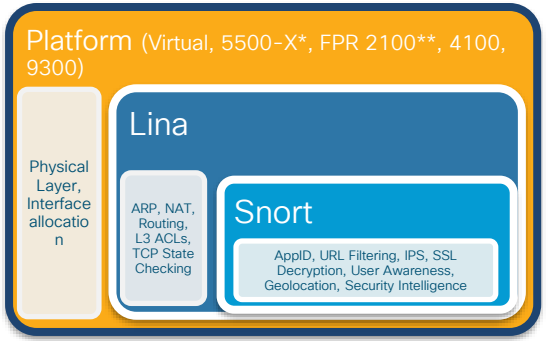
```
FPR9300-A# scope ssa
FPR9300-A /ssa # show logical-device

Logical Device:
    Name    Description Slot ID     Mode        Operational State
Template Name
    ---------- -----------      ----------   ----------    ---------
---------------        -------------
     FTD                          1           Standalone   Ok
ftd
FPR9300-A /ssa #
FPR9300-A /ssa # scope logical-device FTD
FPR9300-A /ssa/logical-device # show configuration
 enter logical-device FTD ftd 1 standalone
     enter external-port-link Ethernet13_ftd Ethernet1/3 ftd
         set decorator ""
         set description ""
        set port-name Ethernet1/3
     exit
```



Functional Overview – Physical Layer (L1)

Viewing interface statistics in FXOS CLI:

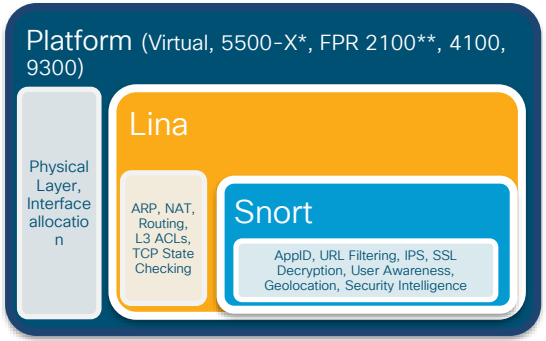
```
FPR9300-A# scope eth-uplink
FPR9300-A /eth-uplink # scope fabric a
FPR9300-A /eth-uplink/fabric # show interface detail

Interface:
    Port Name: Ethernet1/3
...
FPR9300-A /eth-uplink/fabric # scope interface 1 3
FPR9300-A /eth-uplink/fabric/interface # show stats
...
Ether Rx Stats:
    Time Collected: 2017-04-17T23:45:33.906
    Monitored Object: sys/switch-A/slot-1/switch-ether/port-3/rx-
stats
    Suspect: No
    Total Packets (packets): 8968254
    Total Bytes (bytes): 1798297716
    Unicast Packets (packets): 1098012
    Multicast Packets (packets): 2480578
    Broadcast Packets (packets): 5389664
```



Functional Overview – Data/Network Layer (L2/3)

You can see L2 and L3-related interface information in the Unified CLI:

```
> show interface Ethernet1/3
Interface Ethernet1/3 "diagnostic", is up, line protocol is up
  Hardware is EtherSVI, BW 1000 Mbps, DLY 1000 usec
     MAC address b0aa.772f.849c, MTU 1500
     IP address 10.10.1.1, subnet mask 255.255.255.0
  Traffic Statistics for "diagnostic":
     4380985 packets input, 201525318 bytes
     0 packets output, 0 bytes
     162 packets dropped
      1 minute input rate 9 pkts/sec,  437 bytes/sec
      1 minute output rate 0 pkts/sec,  0 bytes/sec
      1 minute drop rate, 0 pkts/sec
      5 minute input rate 9 pkts/sec,  446 bytes/sec
      5 minute output rate 0 pkts/sec,  0 bytes/sec
      5 minute drop rate, 0 pkts/sec
     Management-only interface. Blocked 0 through-the-device packets
```

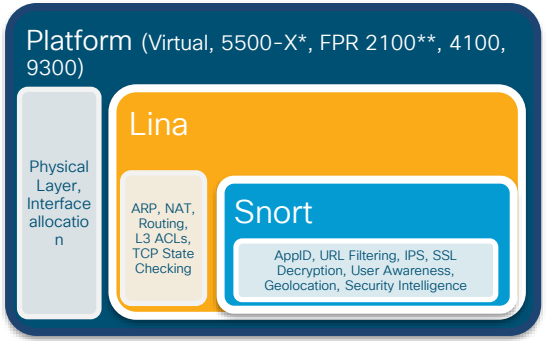


*Note that the above interface is a management-only interface

Functional Overview – Network Layer (L3)

You can also view NAT configuration and active routes in the Unified CLI:

```
> show running-config nat
nat (inside,outside) source dynamic INSIDE_NETS interface
!
object network SRV-10.10.1.100-REAL
 nat (inside,outside) static SRV-10.10.1.100-GLOBAL
!
> show route
...
S*        0.0.0.0 0.0.0.0 [1/0] via 172.18.249.1, outside
C         169.254.1.0 255.255.255.252 is directly connected, nlp_int_tap
L         169.254.1.1 255.255.255.255 is directly connected, nlp_int_tap

>
```



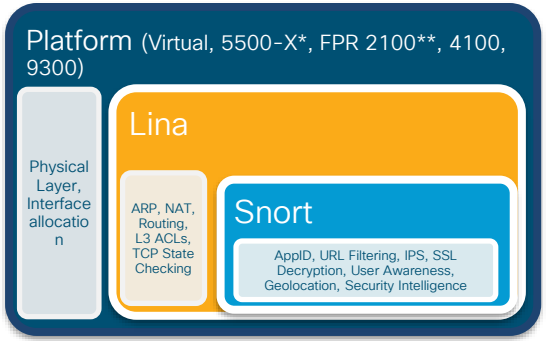
TAC Tip

All legacy ASA show and debug commands are still available in FTD via the ‘system support diagnostic-cli command

Functional Overview – Network/Transport (L4)

TCP state and L3/L4 ACL checking are performed by the Lina process

```
> show conn protocol tcp
165 in use, 54084 most used

TCP outside  10.106.45.60:443 inside38  14.38.104.110:56946, idle 0:00:18…
TCP outside  108.171.133.146:8080 inside38  14.38.104.1:25148, idle 0:00:03…
TCP outside  108.171.133.146:8080 inside38  14.38.104.1:13080, idle 0:00:21…
>
```

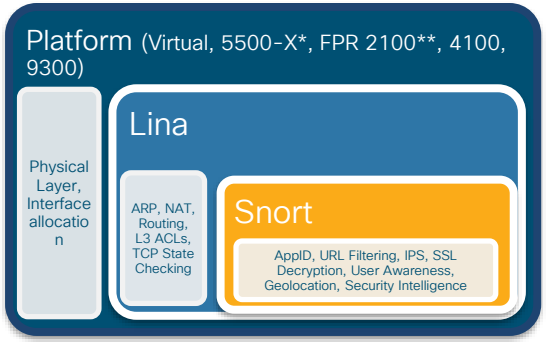



```
> show running-config access-list
access-list CSM_FW_ACL_ remark rule-id 268445405: PREFILTER POLICY: Default Prefilter Policy_1
access-list CSM_FW_ACL_ remark rule-id 268444672: ACCESS POLICY: FTD-ACPolicy-201703230950 - Default/1
access-list CSM_FW_ACL_ remark rule-id 268444672: L7 RULE: from_outside_#1
access-list CSM_FW_ACL_ advanced permit udp ifc outside 10.2.2.0 255.255.255.0 host 10.1.1.100 eq syslog
rule-id 268444672
…
```

Functional Overview – Upper Layers (5-7)

Snort-handled functions that occur at upper OSI layers:

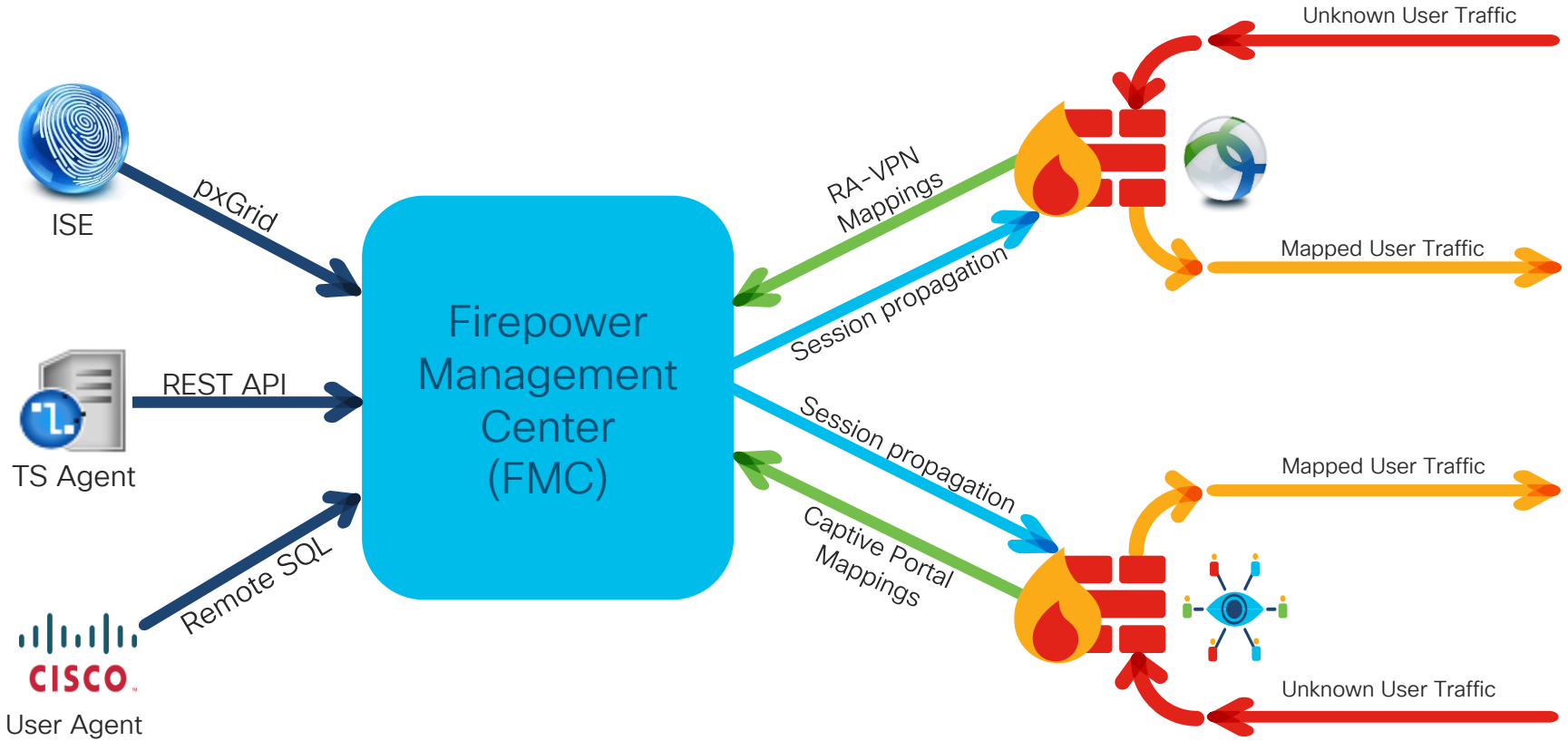
- Intrusion Prevention System (IPS)
- App Detection and OpenAppID
- URL Filtering
- SSL/TLS Decryption
- User Identity Awareness
- File and malware inspection



User Identity Overview

- Allows for auditing of user activity
- Allows for User and Group based access control
- Types of authentication
  - Active:
    - Captive Portal
    - Remote Access VPN (RA-VPN)
  - Passive:
    - Cisco Firepower User Agent (CFUA)
    - Terminal Services Agent (TSAgent)
    - Identity Services Engine (ISE / ISE-PIC)

Identity Architecture



Architecture Overview: CPU and Memory Allocation

FTD CPU and Memory Allocation

- CPU and memory are allocated to Lina and Snort via the use of Linux cgroups
- This resource pool (cgroup) separation limits scope of problem impact
- Troubleshooting approach depends on where issue resides

Example of Lina and Snort CPU allocations on a Firepower 9300

```
firepower# show kernel cgroup-controller cpuset | begin lina
group "restricted/lina"  ← Lina
  cpuset.cpus: 1-8,18-25,37-44,54-61
  cpuset.mems: 0-1
  tasks:
     12507  12794  12803  12804
     12805  15917  15918  15943

group "restricted/qemu" ← Snort
  cpuset.cpus: 9-17,26-35,45-53,62-71
  cpuset.mems: 0-1
```

Lina Memory – Overview

- Lina memory is broken into two categories: Shared memory and DMA memory

```
firepower# show memory
Free memory:          250170904 bytes (47%)
Used memory:          286700008 bytes (53%)
-------------      ------------------
Total memory:         536870912 bytes (100%)
```

- If available memory trends down over time, call Cisco TAC

```
%ASA-3-211001: Memory allocation Error
```

  - Use CISCO-ENHANCED-MEMPOOL-MIB.my for accurate SNMP counters
  - Free memory may not recover immediately after conn spike due to caching

Lina Memory Blocks (Direct Memory Access)

- DMA memory involves fixed-size blocks allocated at startup
- Used for packet processing, VPN, etc.

```
firepower# show blocks
  SIZE     MAX     LOW     CNT
     0     400     397     400
     4     100      99      99
    80     403     379     401
   256    1200    1190    1195
  1550    6511     803     903
  2048    1200    1197    1200
  2560     264     264     264
  4096     100     100     100
  8192     100     100     100
  9344    2000    2000    2000
 16384     102     102     102
 65536      16      16      16
firepower#
```

Current number of free blocks available

1550, 2048, and 9344 byte blocks are used for processing Ethernet frames

When DMA memory for a specific block size runs low, the following syslog will be generated for the specific block size:

```
%ASA-3-321007: System is low on free memory blocks of size 1550 (10 CNT out of 7196 MAX)
```

Lina CPU Utilization by Processes

- **show processes cpu-usage** command displays the amount of CPU used on a per-process basis for the last 5 sec, 1 min, and 5 min

Heavy CPU load from SNMP traps.

```
> show process cpu-usage sorted non-zero
PC         Thread       5Sec     1Min     5Min    Process
0x08dc4f6c   0xc81abd38    14.4%     8.2%     8.0%   SNMP Notify Thread
0x081daca1   0xc81bcf70     1.3%     1.1%     1.0%   Dispatch Unit
0x08e7b225   0xc81a28f0     1.2%     0.1%     0.0%   ssh
0x08ebd76c   0xc81b5db0     0.6%     0.3%     0.3%   Logger
0x087b4c65   0xc81aaaf0     0.1%     0.1%     0.1%   MFIB
0x086a677e   0xc81ab928     0.1%     0.1%     0.1%   ARP Thread
```

If you have high CPU utilization for a generic process such as DATAPATH, contact the TAC as there are more granular CPU profiling tools available for deeper investigation

Snort, Lina, and the Firepower ecosystem

- Many processes run on Linux to support event collection and other management, including:

Process	Primary Purpose
Lina	ASA-like functions: L4 ACLs, ALG, Routing, Failover, Clustering, etc
Snort	Inspects traffic and writes events to unified log files
SFDataCorrelator	Read unified logs written by snort, and send events to FMC
sftunnel	Manage an encrypted connection back to the FMC over TCP/8305
ids_event_alerter	Sends syslogs and SNMP traps from sensor for intrusion events

- Process status can be verified with: **`> pmtool status`**
- Standard Linux troubleshooting tools, such as "**`top,`**" can be used to verify CPU and memory

Expert Mode – CPU Utilization by Processes

Open "**top**" program from BASH (Sorting by CPU is the default)

```
> expert
admin@firepower:~$ top

Cpu(s): 15.3%us,  5.8%sy,  0.0%ni, 78.4%id,  0.0%wa,  0.0%hi,  0.5%si,  0.0%st
Mem:  12321960k total,  5605756k used,  6716204k free,   148992k buffers
Swap:  3998716k total,      780k used,  3997936k free,  1222064k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
12221 root       0 -20 1896m 299m  75m S  100  2.5   2733:37 lina
22420 root      20   0  618m 8048 2980 S   42  0.1   1539:57 sftunnel
14777 root      20   0 2185m  60m  12m S    0  0.5   8:11.23 SFDataCorrelator
25979 root      20   0 1893m 347m  12m S    0  2.9   2:15.42 snort
```

Processes sorted by CPU

- Lina handles its own resources. Disregard high CPU and memory readings for Lina in "top"
- Occasional high CPU for Snort is determined by current flow

Expert Mode – Memory Utilization by Processes

```
> expert
admin@firepower:~$ top
```

1. Open “top” program
2. Type “shift + f” to choose sorting field
3. Type “n” to select resident memory

```
Current Sort Field:  N  for window 1:Def
Select sort field via field letter
  k: %CPU        = CPU usage
  l: TIME        = CPU Time
  m: TIME+       = CPU Time, hundredths
* N: %MEM        = Memory usage (RES)
  o: VIRT        = Virtual Image (kb)
```

Processes sorted by resident memory

```
Tasks: 465 total,   1 running, 464 s                 opped,   0 zombie
Cpu(s): 41.6%us,  0.3%sy,  0.0%ni, 58.1%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:  132166192k total, 43796884k used, 88636864k free,      252k buffers
Swap:  7810780k total,        0k used,  7810780k free,  1732192k cached
  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
12506 root       0 -20 26.1g 1.1g 643m S 1993  0.8  97328:59 lina
11949 root       1 -19 7813m 671m  37m S    2  0.5   6:15.66 snort
12902 root      20   0 4129m  68m  16m S    2  0.1  41:54.55 SFDataCorrelator
```

Expert Mode – Memory Management Example

- Snort is the primary memory consumer, and will use more memory over time
- Low system memory is not necessarily a sign of a problem

Round numbers used to simplify example

```
"System" cgroup
Limit: 5 GB

Memory        Process
1 GB      lina
1 GB      SFDataCorrelator
1 GB      Database
1 GB      DiskManager
1 GB      ids_event_alerter
```

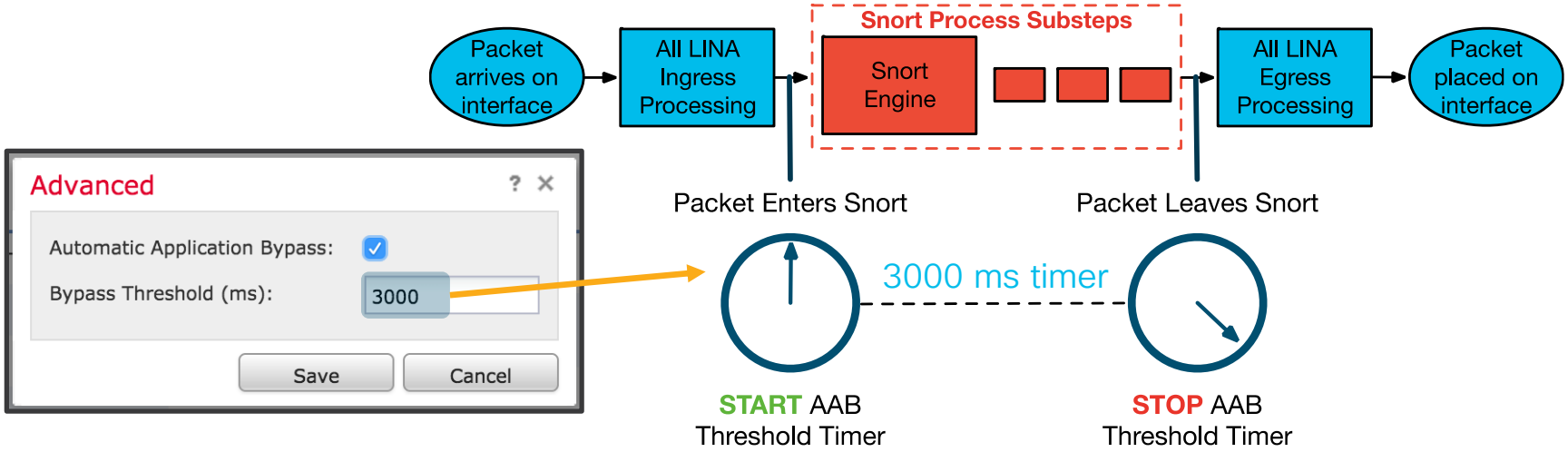
```
"Detection" cgroup
Limit: 10 GB

Memory    Process
2 GB snort
2 GB snort
2 GB snort
2 GB snort
2 GB snort
```

Errors in `/var/log/messages`

```
kernel: SFDataCorrelator invoked oom-killer: gfp_mask=0xd0, order=0, oom_adj=0
kernel: Task in /System killed as a result of limit of /System
```

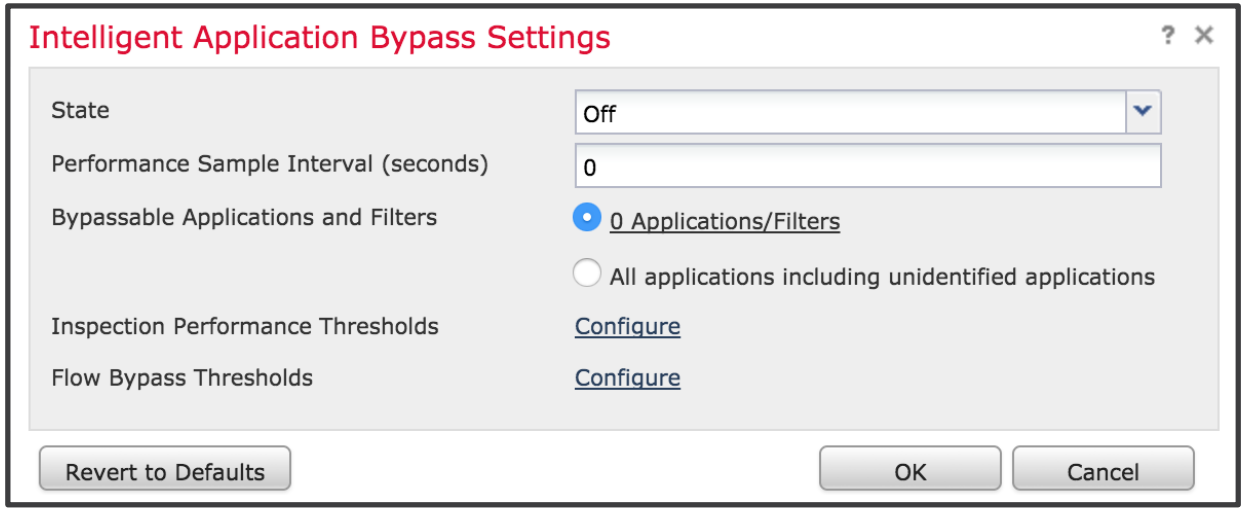
- Snort is protected from low-memory issues caused by processes in other cgroups

Snort – Automatic Application Bypass



- AAB is a per packet timer for snort
- A snort instance is killed if a packet fails to egress before the threshold
- A snort core file is collected for root cause analysis
- The process manager will respawn snort
- Do not go below 3000 milliseconds threshold unless recommended by TAC

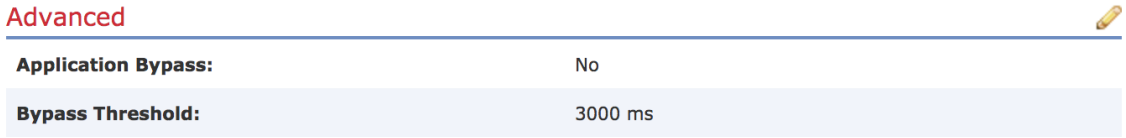
Snort – Intelligent Application Bypass



- IAB is a performance optimization tool for elephant flows
- Invoked in a simple 2-step process:
  1. Does snort exceed the ”Inspection Performance Thresholds” (high CPU, % dropped traffic, etc)?
  2. If yes, then dynamically Trust flows which match “Flow Thresholds” (bytes/sec, packets/flow, etc).
- Configured under Access Control Policy > Advanced tab

Snort - AAB vs IAB

- Automatic Application Bypass (AAB) mitigates dataplane impact of:
  - Out of memory situations impacting snort
  - Snort brownout / deadlock scenarios
  - Misconfiguration or over-subscription of IPS*
  - Always enable AAB at the default 3000 ms threshold - NOT enabled by default!



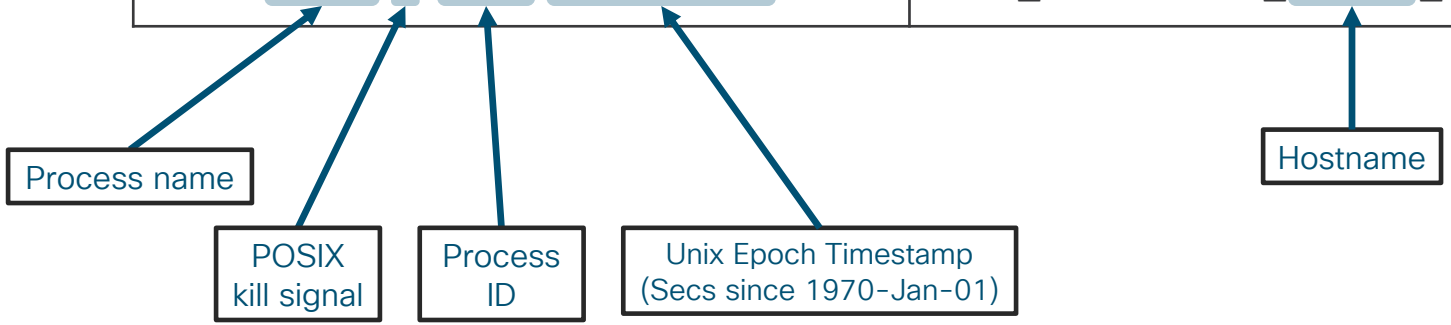
* AAB is occasionally invoked when too many snort rules are enabled, or too much traffic is inspected

- Intelligent Application Bypass addresses large flow performance, whereas AAB is a stability feature
- Enable IAB on a case-by-case basis where prefilter and Trust rules do not fit requirements

Expert Mode – Core Files

- If a process on Linux exits unexpectedly, a core file may be written to the file system

FTD on FP2100, FP4100, FP9300	FTD on ASA and Virtual Platforms (VMware, KVM, AWS, Azure)
• Cores written to `/opt/cisco/csp/cores/` • Core automatically compressed and moved to **`/ngfw/var/data/cores/`**	• Cores written uncompressed to **`/ngfw/var/common/`**
`core.snort.6.5373.1496879772.gz`	`core_1496879772_sensor_snort_6.5373`



Expert Mode – Disk Management

- The DiskManager process manages collections of files called “silos”
- If space is low, DiskManager will prune each silo based on a preconfigured threshold

```
> show disk-manager
Silo                                Used          Minimum      Maximum
Temporary Files                     0 KB          584.291 MB   2.282 GB
Backups                             0 KB          4.565 GB     11.412 GB
Updates                             0 KB          6.847 GB     17.118 GB
Archives & Cores & File Logs        0 KB          4.565 GB     22.824 GB
RNA Events                          0 KB          4.565 GB     18.259 GB
File Capture                        0 KB          11.412 GB    22.824 GB
Connection Events                   0 KB          413.320 MB   826.642 MB
IPS Events                          0 KB          13.694 GB    34.236 GB
[lines_removed]
```

Expert Mode – Disk Management

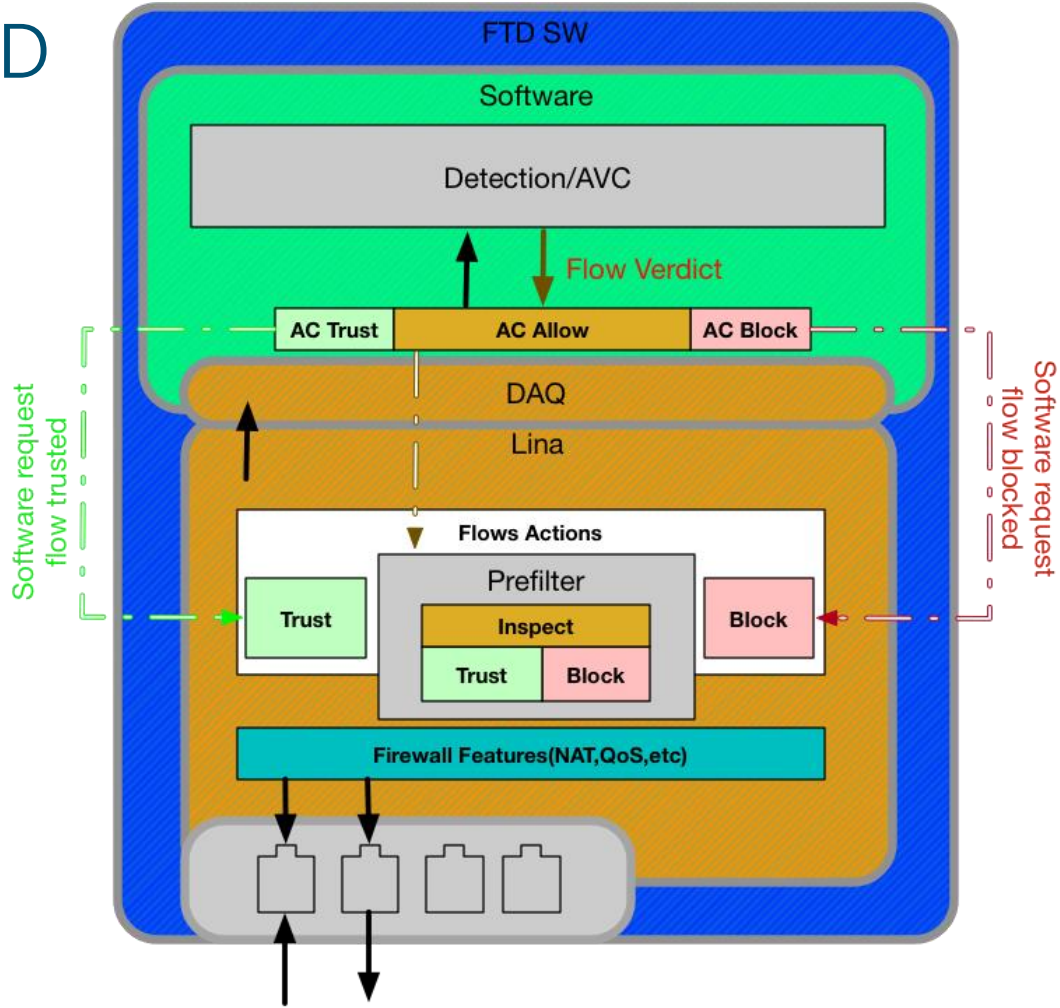
- The Lina file system is accessible from expert mode via /mnt/disk0

```
# Create a capture from the unified CLI
> capture CAPTURE match ip any host 8.8.8.8

# Enter the diagnostic (lina-only) CLI
> system support diagnostic-cli
firepower# copy /pcap capture:CAPTURE disk0:CAPTURE.pcap

# Enter expert mode and browse to /mnt/disk0
> expert
admin@FPR4100:/mnt/disk0 $ ls
CAPTURE.pcap
[lines_removed]
```

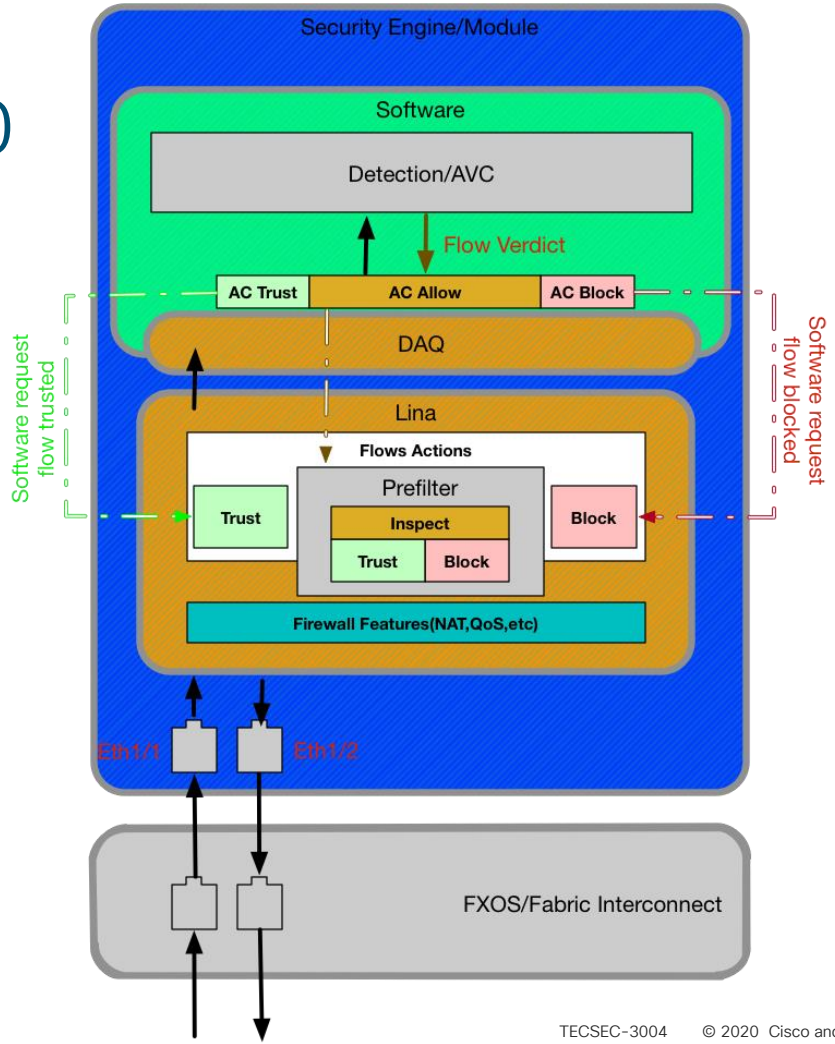
The Path of the Packet (Platform Architecture)

Virtual FTD

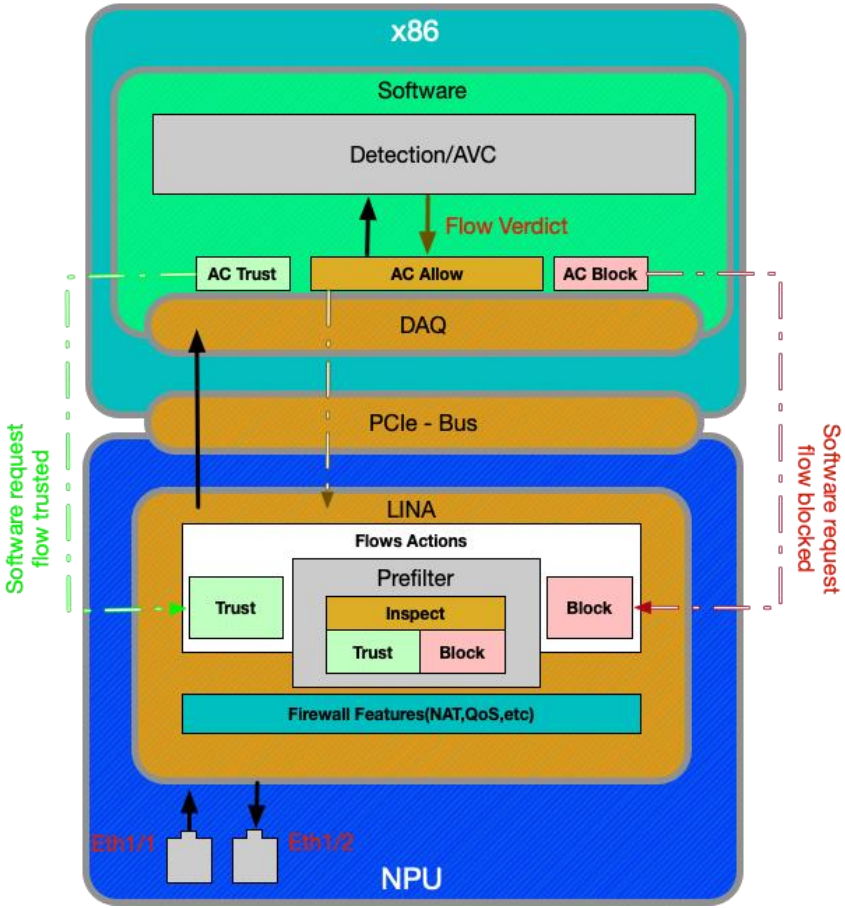
VMWare
AWS
Azure
KVM
Hyper-V



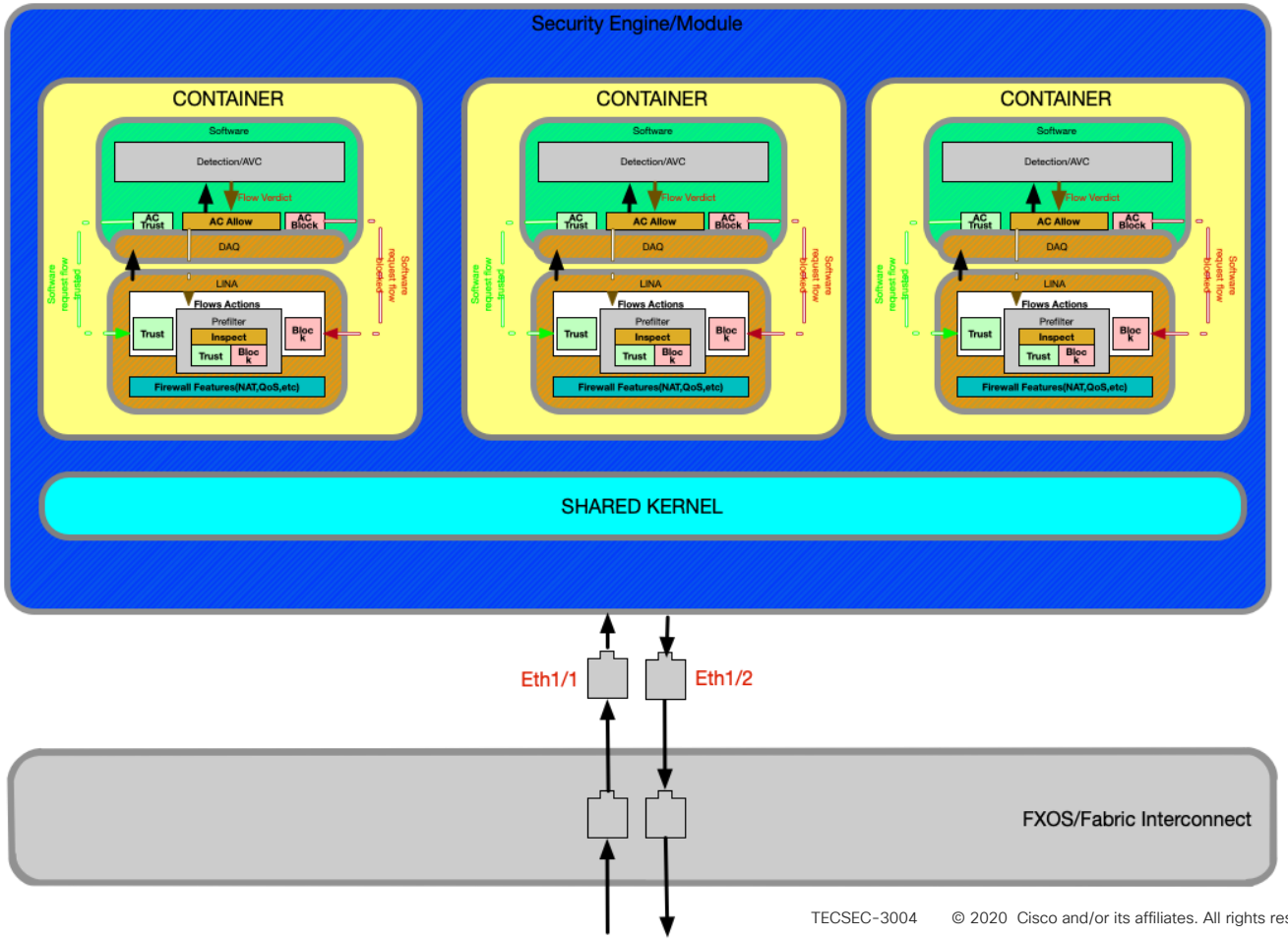
Firepower 4100 & 9300



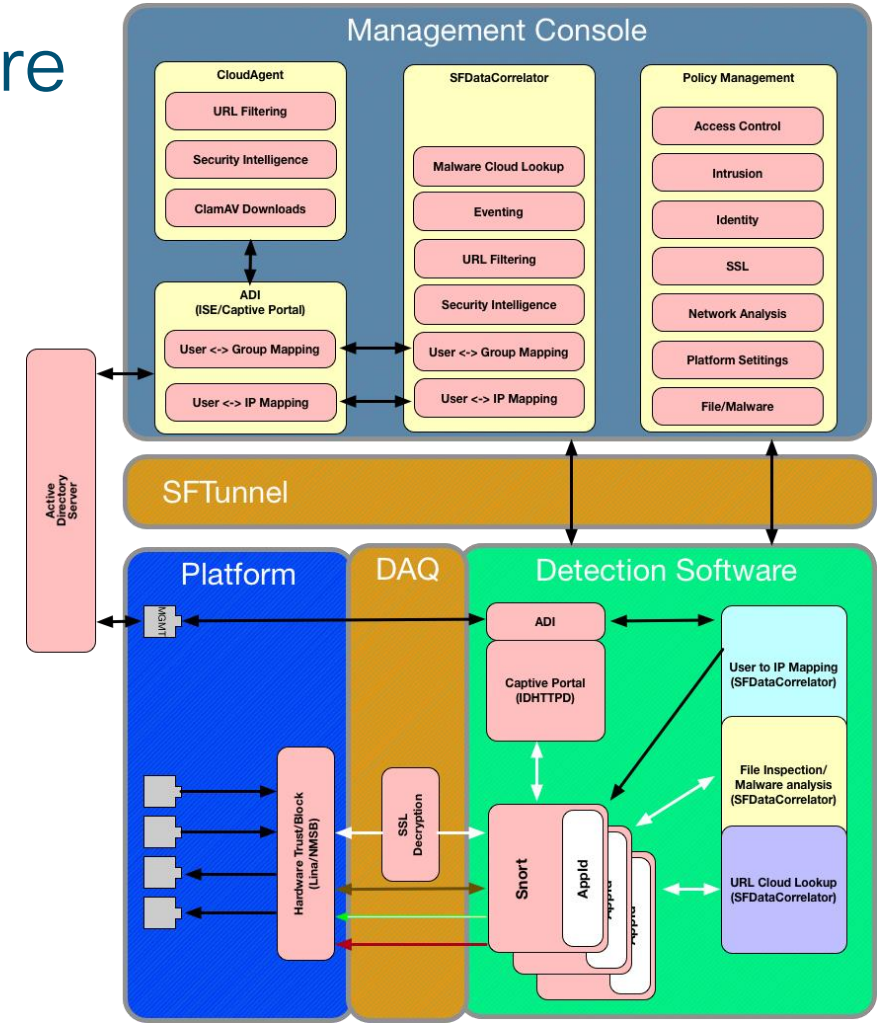
Firepower 2100



Multi-Instance architecture overview(9300/4100)



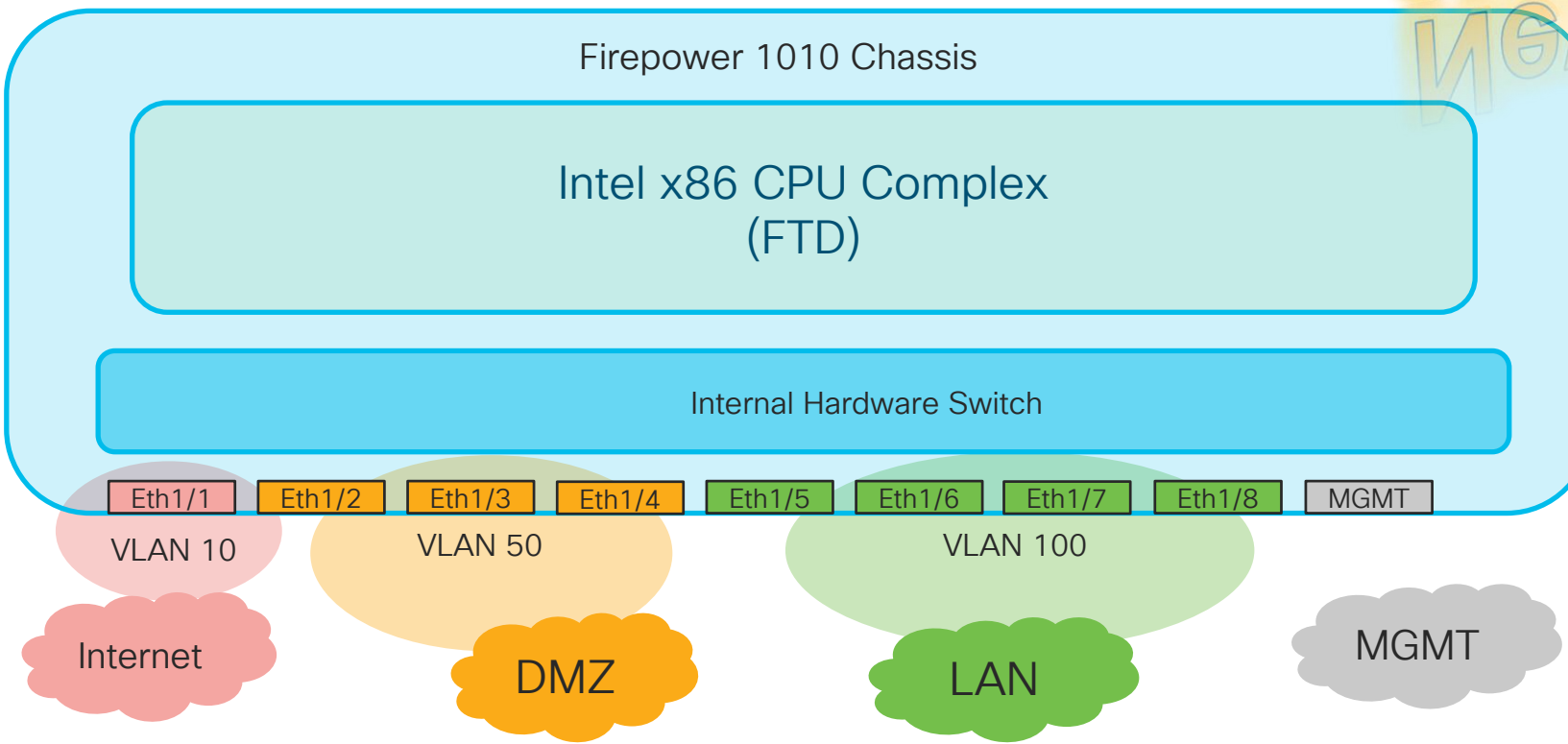
Virtual/Software diagram



Firepower 1010 – L2 Switching Overview

New!

- New in 6.5 – Eliminates the need for an external switch in SOHO environments



The Path of the Packet (Software / Logical Flow)

Understanding Packet Flow

Effective troubleshooting requires an understanding of the packet path in network

1. Attempt to isolate the problem down to a single device
2. Perform a systematic walk of the packet through device to identify problem

For problems relating to FTD, <u>always</u>

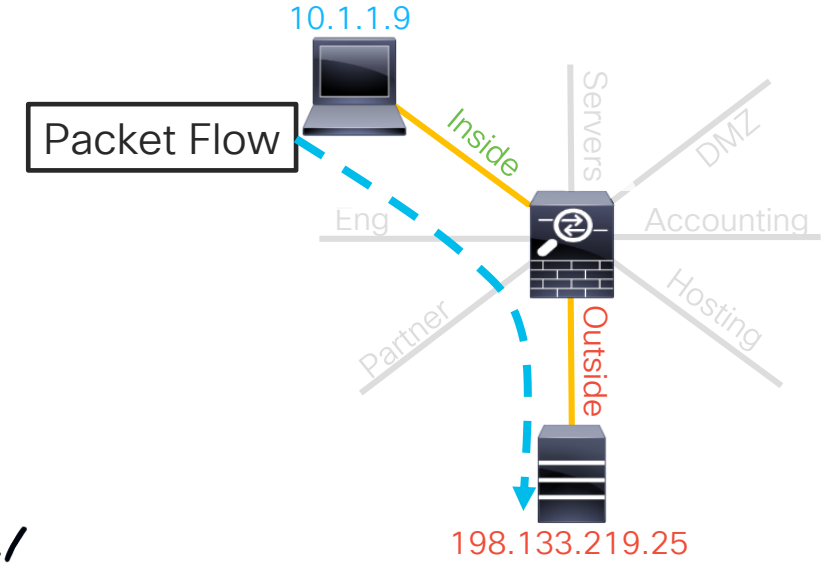
- Determine the flow: Protocol, Source IP, Destination IP, Source Port, Destination Port
- Determine the logical (named) interfaces through which the flow passes

```
TCP outside 172.16.164.216:5620 inside 192.168.1.150:50141, idle 0:00:00, bytes 0, flags saA
```

All firewall connectivity issues can be simplified to two interfaces (ingress and egress) and the policies tied to both

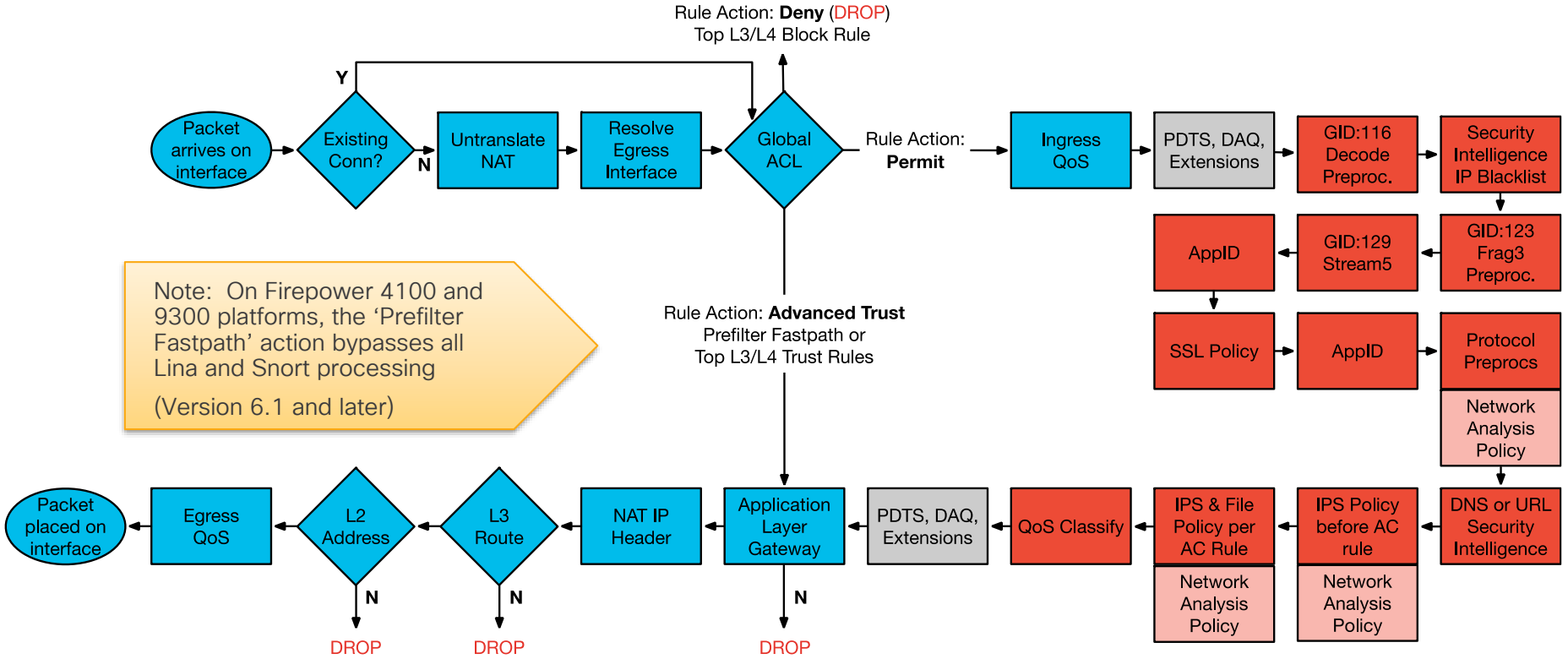
Example Flow

- TCP Flow
  - Source IP : 10.1.1.9 Source Port : 11030
  - Destination IP : 198.133.219.25 Destination Port : 80
- Interfaces
  - Source: Inside Destination: Outside



With the Flow defined, examination of configuration issues boils down to just the two Interfaces: Inside and Outside

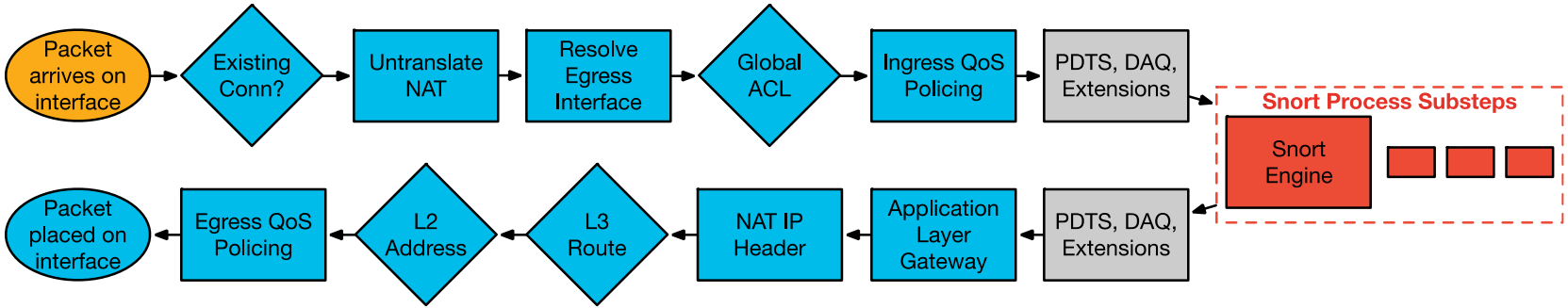
Reference Slide: Routed FTD Path of Packet



LINA ASA Engine = BLUE

Snort Engine = RED

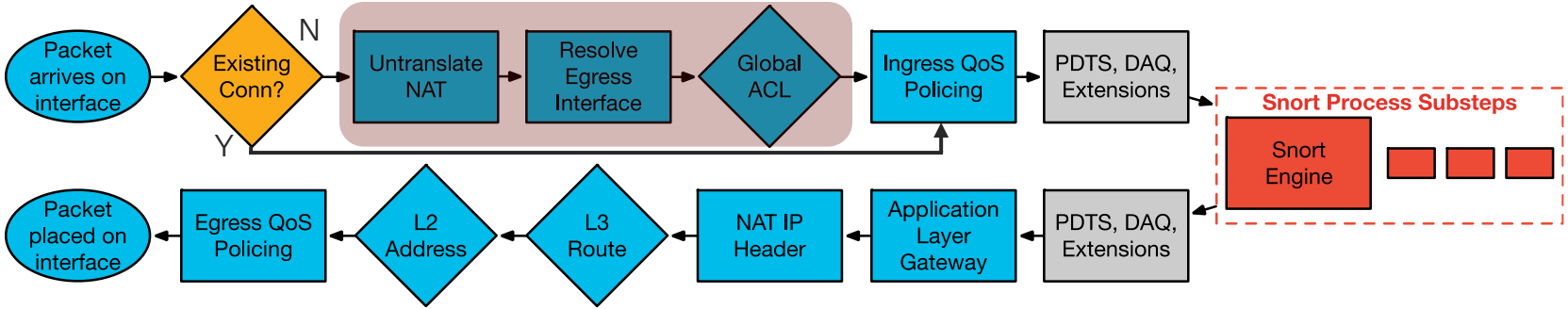
Packet Processing: Ingress interface



- Packet arrives on ingress interface
- Input counters incremented by NIC and periodically retrieved by CPU
- Software input queue (RX ring) is an indicator of packet load
- Overrun counter indicates packet drops (usually packet bursts)

```
> show interface outside
Interface GigabitEthernet0/3 "outside", is up, line protocol is up
  Hardware is i82546GB rev03, BW 1000 Mbps, DLY 10 usec
[…]
        IP address 148.167.254.24, subnet mask 255.255.255.128
        54365986 packets input, 19026041545 bytes, 0 no buffer
        Received 158602 broadcasts, 0 runts, 0 giants
        0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
[…]
        input queue (blocks free curr/low): hardware (255/230)
        output queue (blocks free curr/low): hardware (254/65)
```

Packet Processing: Locate Connection



- Check for existing connection in conn table

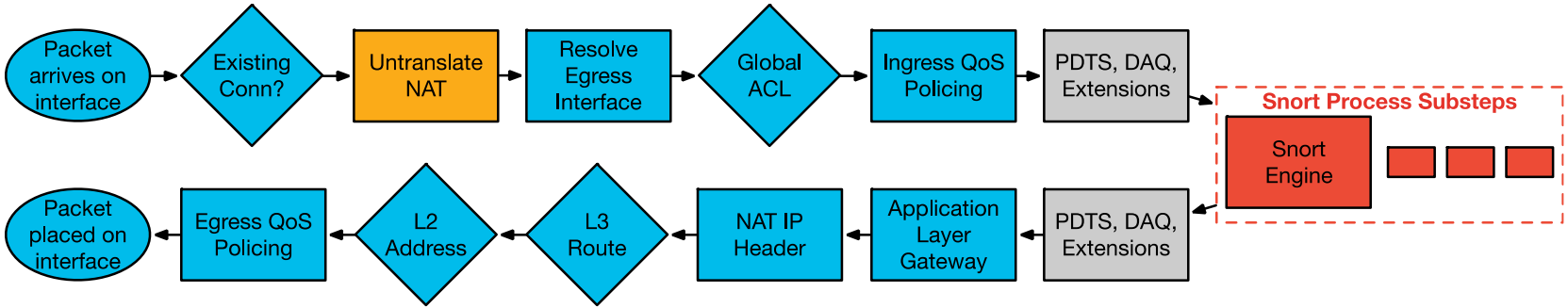
```
> show conn
TCP out 198.133.219.25:80 in 10.1.1.9:11030 idle 0:00:04 Bytes 1293 flags UIO
```

- If no existing connection
  - TCP SYN or UDP packet, pass to ACL and other policy checks in Session Manager
  - TCP non-SYN packet, drop and log

```
ASA-6-106015: Deny TCP (no connection) from 10.1.1.9/11031 to 198.133.219.25/80 flags PSH ACK  on
interface inside
```

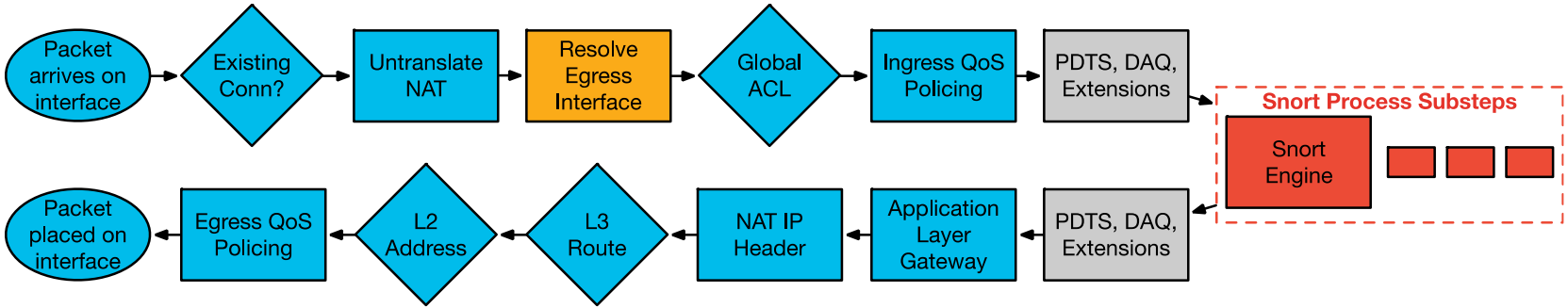
If connection entry exists, bypass ACL check and process in Lina fastpath

Packet Processing: NAT Un-Translate

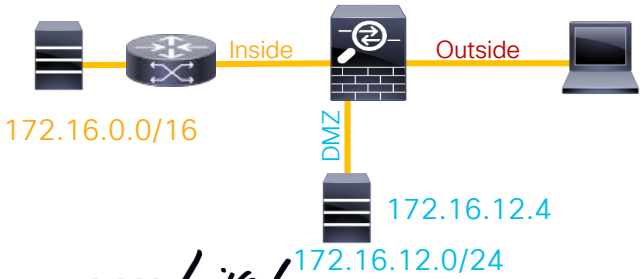


- Incoming packet is checked against NAT rules
- Packet is un-translated first, before ACL check
- NAT rules that translate the destination of the packet can override the routing table to determine egress interface (NAT divert)
  - Could also override policy-based routing (PBR)

Packet Processing: Egress Interface



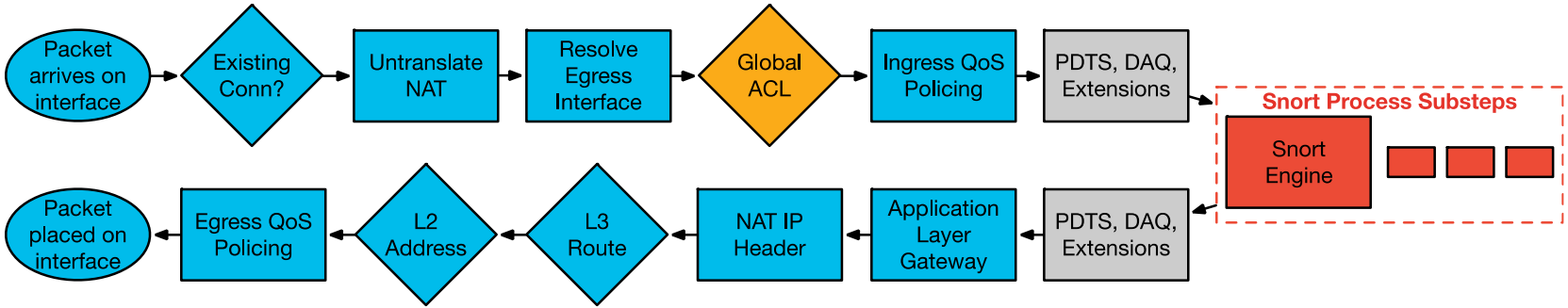
- Egress interface is determined first by translation rules or existing conn entry
- If NAT does not divert to the egress interface, the global routing table is consulted to determine egress interface



Packets received on **outside** and destined to 192.168.12.4 get routed to 172.16.12.4 on **inside** based on NAT configuration.

```
nat (inside,outside) source static 172.16.0.0-net 192.168.0.0-net
nat (dmz,outside) source static 172.16.12.0-net 192.168.12.0-net
```

Packet Processing: Global ACL Check

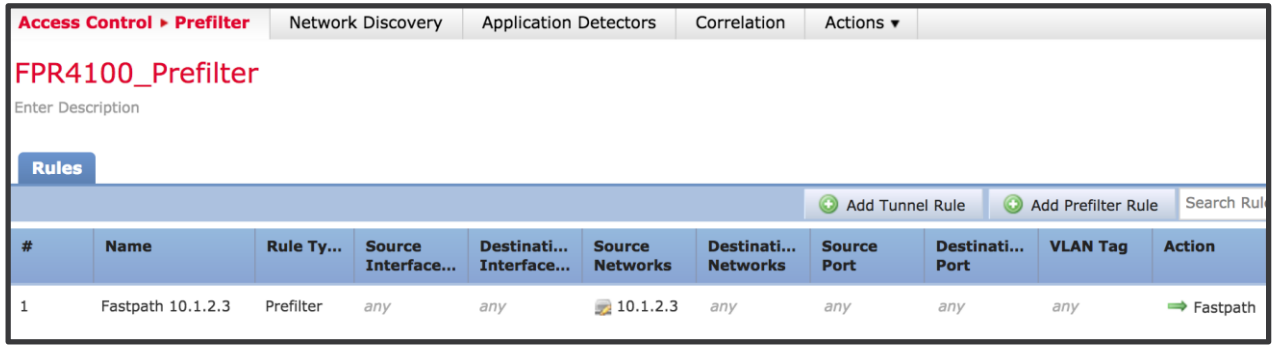


- First packet in flow is processed through ACL checks
- ACLs are first configured match
- First packet in flow matches ACE, incrementing hit count by one

```
> show access-list
…
CSM_FW_ACL_ line 5 advanced permit tcp any any rule-id 9998 (hitcnt=5) 0x52c7a066

> show running-config access-group
access-group CSM_FW_ACL_ global
```

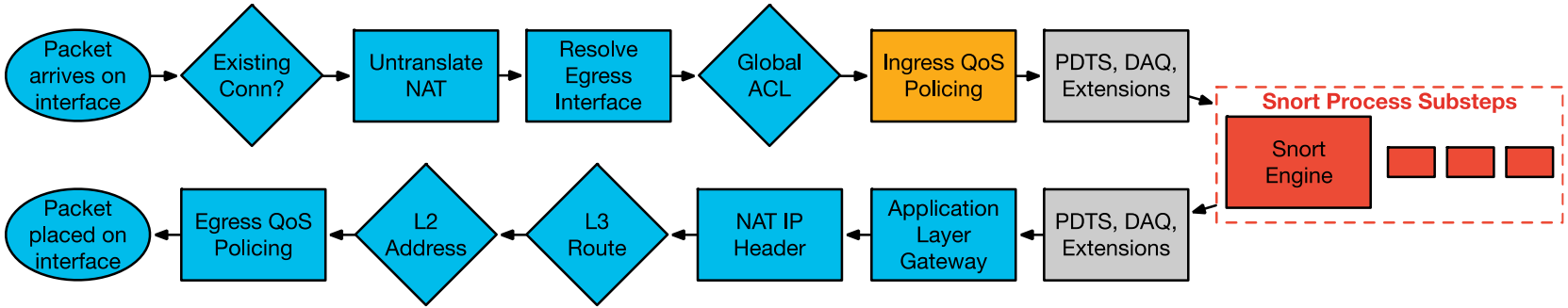
Packet Processing: Global ACL Check

- All L4 access control entries are in one global ACL
- Prefilter Fastpath rules skip snort and show up as "**Advanced Trust**" in Lina global ACL




```
> show running-config access-group
access-group CSM_FW_ACL_ global

> show access-list
[lines_removed]
access-list CSM_FW_ACL_ line 1 remark rule-id 268435484: PREFILTER POLICY: FPR4100_Prefilter
access-list CSM_FW_ACL_ line 2 remark rule-id 268435484: RULE: Fastpath 10.1.2.3
access-list CSM_FW_ACL_ line 3 advanced trust ip host 10.1.2.3 any rule-id 268435484 event-
log flow-end (hitcnt=0) 0x98824a05
```

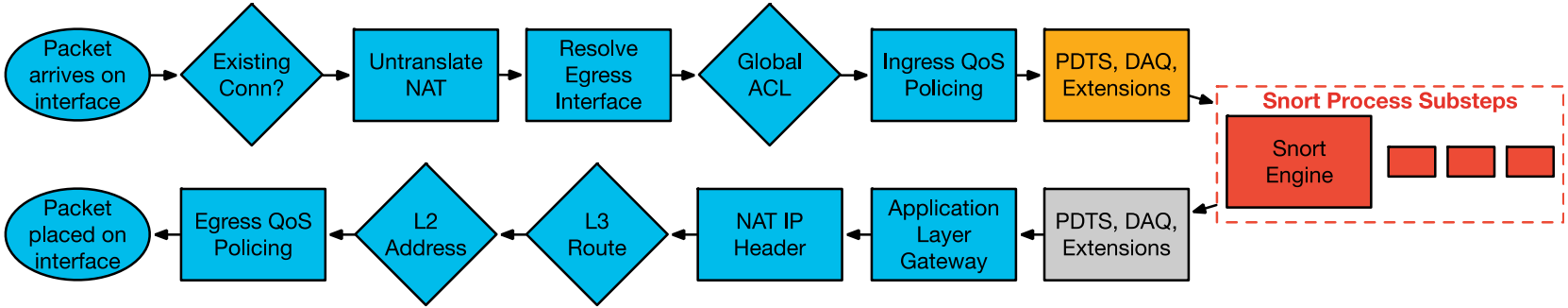
Packet Processing: Ingress QoS Policing



- QoS policing is enforced within the Lina process

```
> system support diagnostic-cli

firepower# show service-policy interface inside
Interface inside:
  Service-policy: policy_map_inside
    Flow-rule QoS id: 268435467
      Input police Interface inside:
        cir 1000000 bps, bc 31250 bytes
```

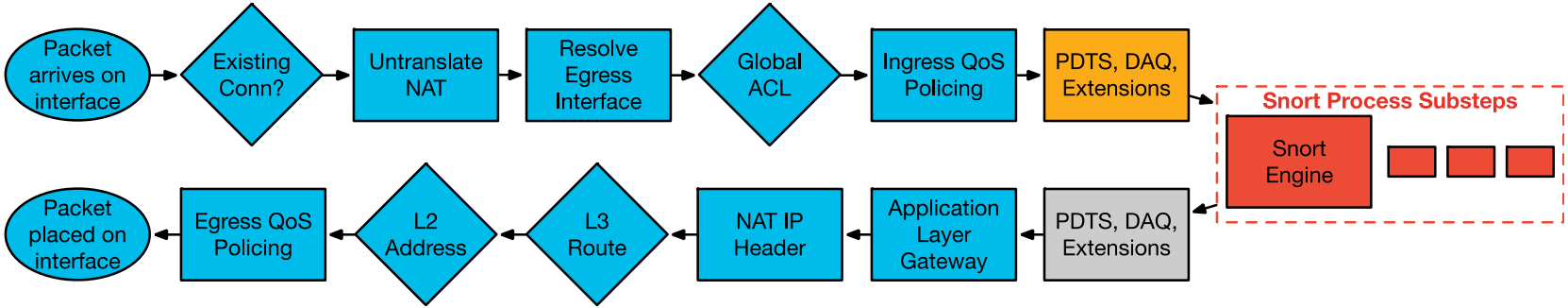
Packet Processing: Packet Data Transport System



The Packet Data Transport System sends packets to Snort after initial Lina inspections

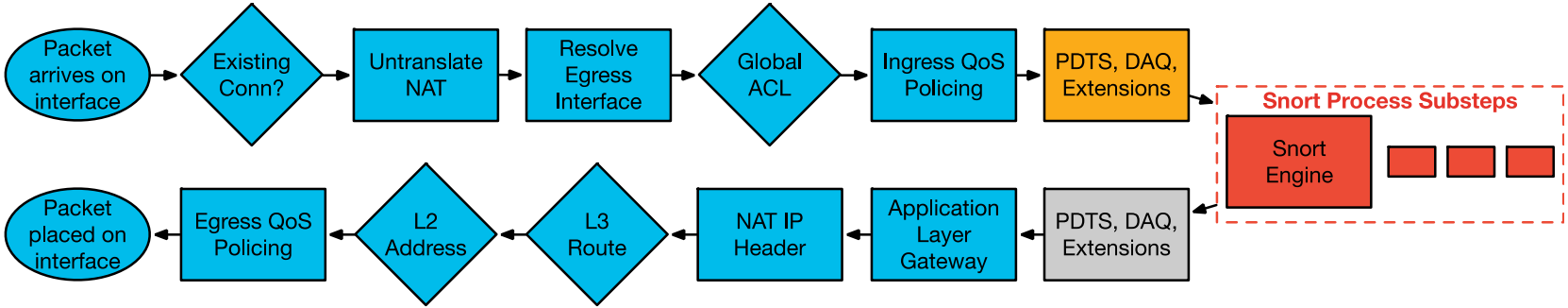
`show asp inspect-dp snort`	Displays conns and packets sent to each snort instance and process ID, as well as snort status
`show asp inspect-dp snort counters summary`	Display frames, bytes, and conns for snort instances
`show asp inspect-dp snort queues`	Display rx and tx queue utilization for snort instances
`clear asp inspect-dp snort`	Clear all of the above PDTS counters
`show asp inspect-dp snort queue-exhaustion`	Display automatic capture of PDTS ring when snort is unable to service queue

Packet Processing: Data Acquisition Library



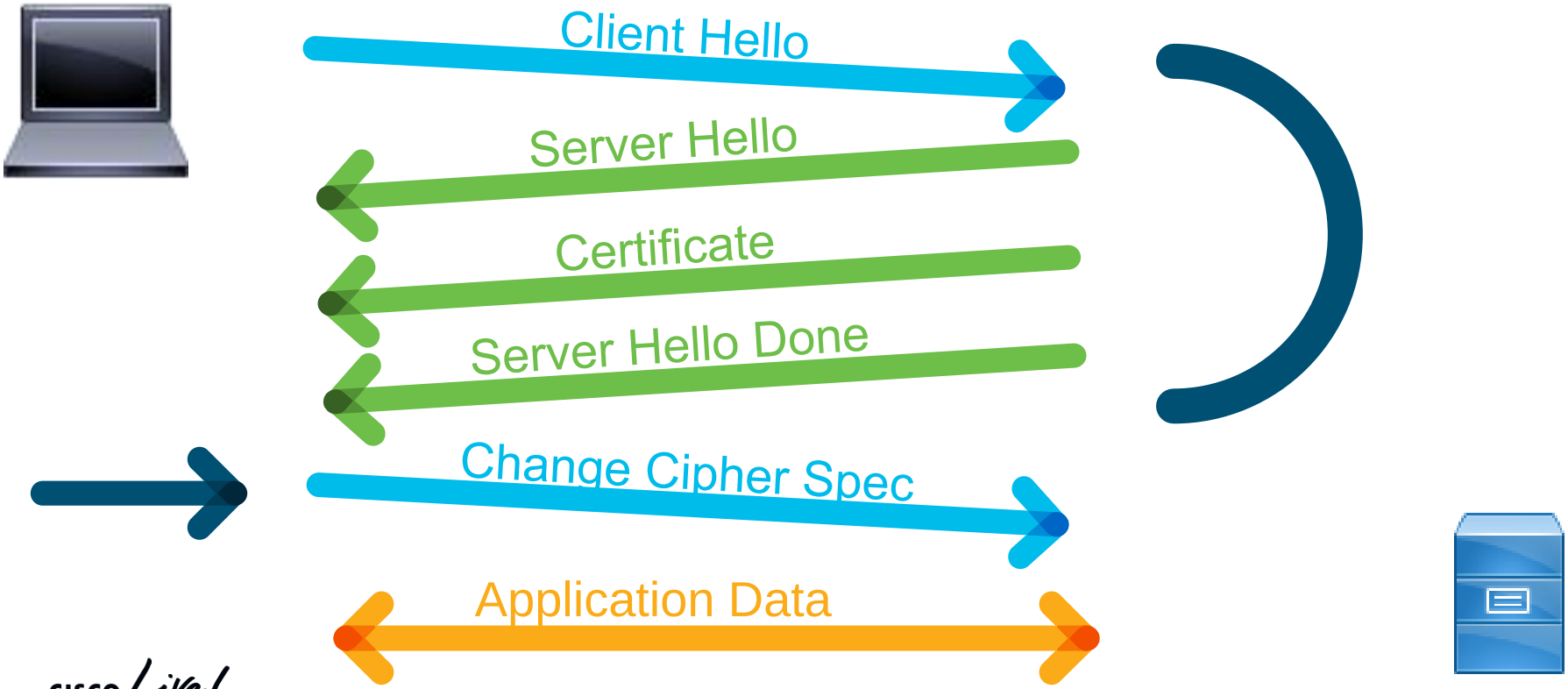
- The Data Acquisition Library (DAQ) enables snort to run on different hardware and software platforms
- Platform-specific changes are made in the DAQ
- DAQ extensions facilitate TLS decryption and a TCP proxy
- Decrypted flows are sent to snort for inspection
- Packets should not be dropped by the DAQ

Packet Processing: SSL Decryption

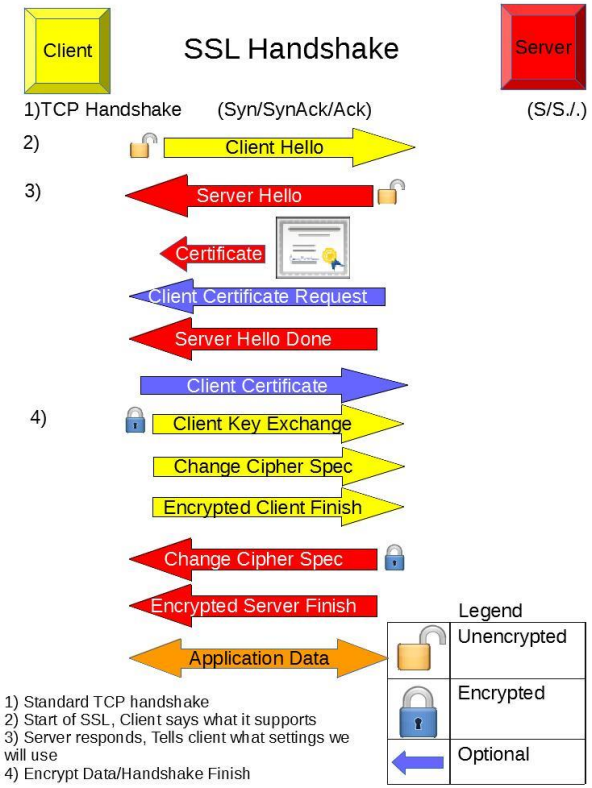


- SSL Decryption touches Lina, DAQ, and Snort
- Lina and DAQ:
  - Proxy TCP sessions
  - Track keys/sessions
  - Decrypt (software) / send to crypto chip to decrypt
- Snort:
  - Enforces policies
  - Makes decisions on whether to decrypt flow or not

Abbreviated SSL handshake



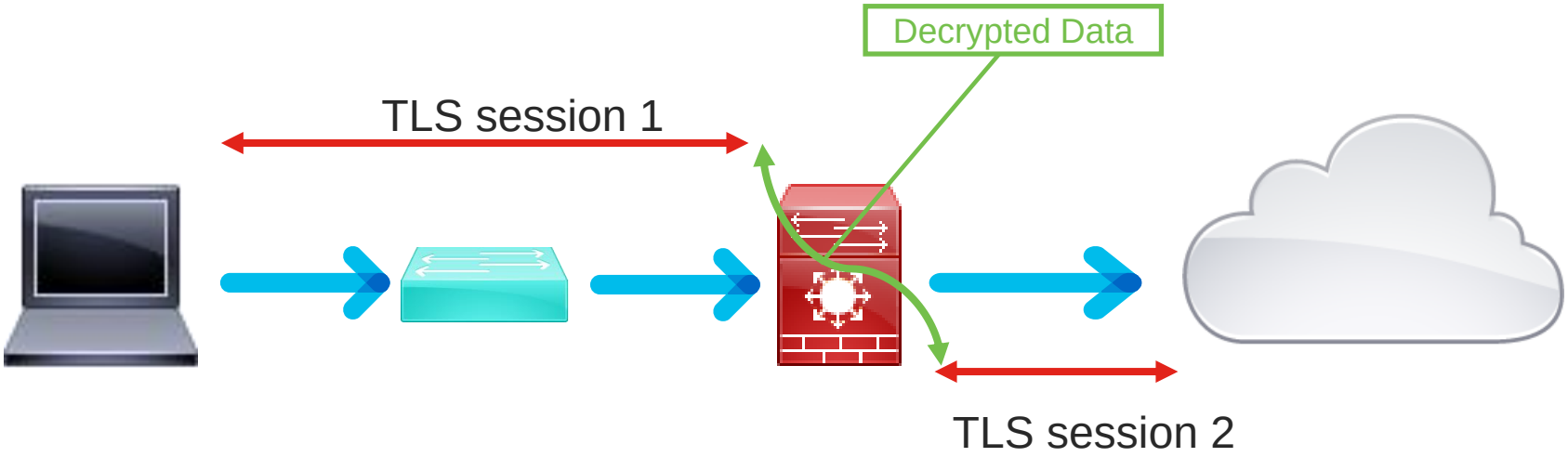
Standard SSL handshake for HTTPS



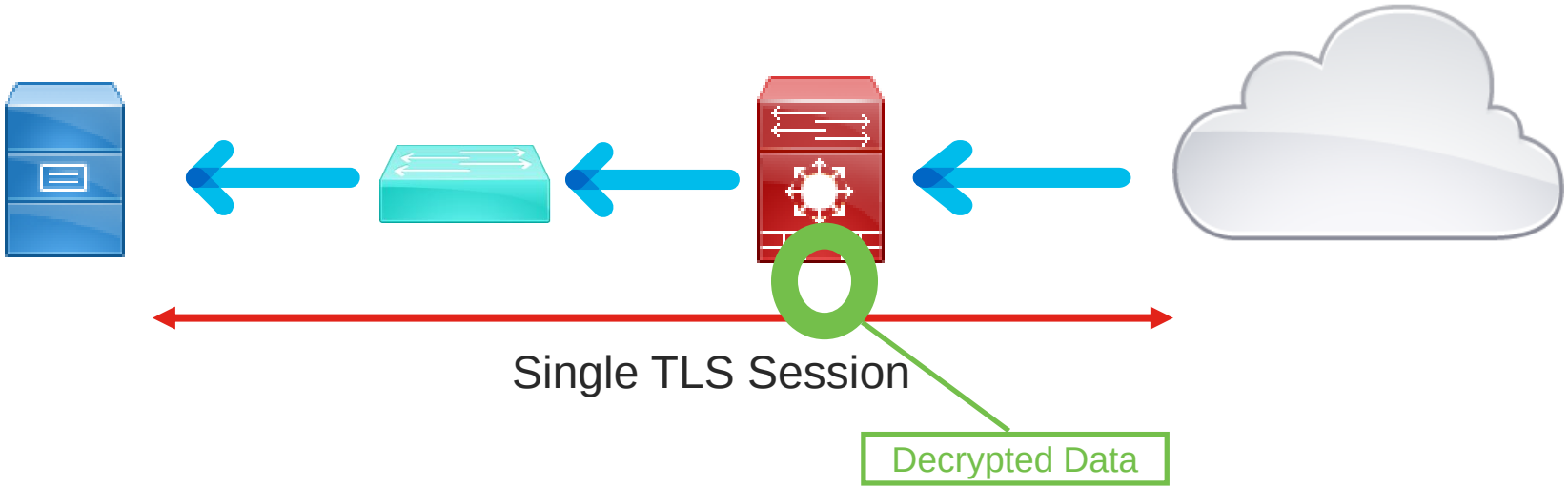
Excellent Reference

https://tls.ulfheim.net/

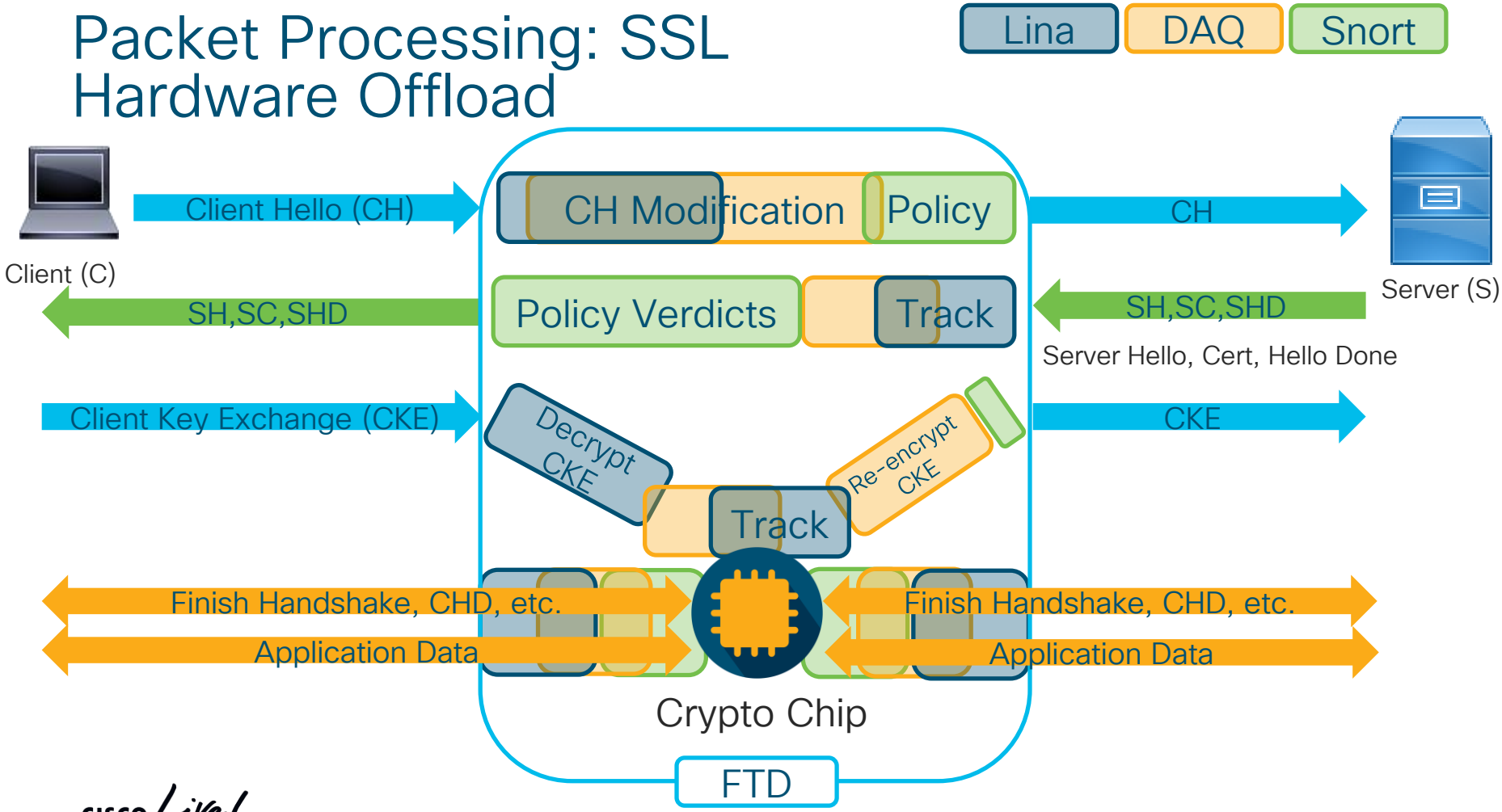
Typical deployment: Decrypt Resign



Typical deployment: Decrypt Known-key

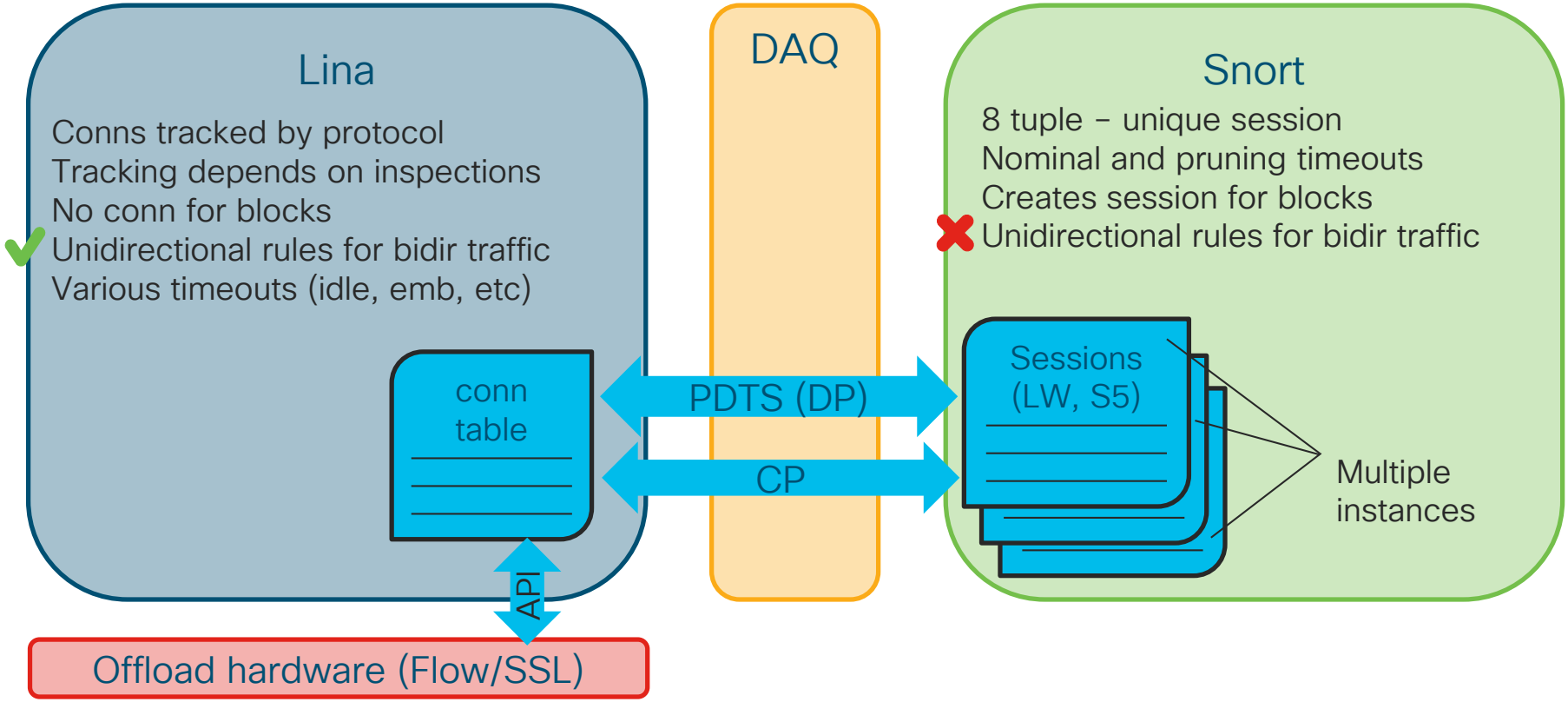


Packet Processing: SSL Hardware Offload

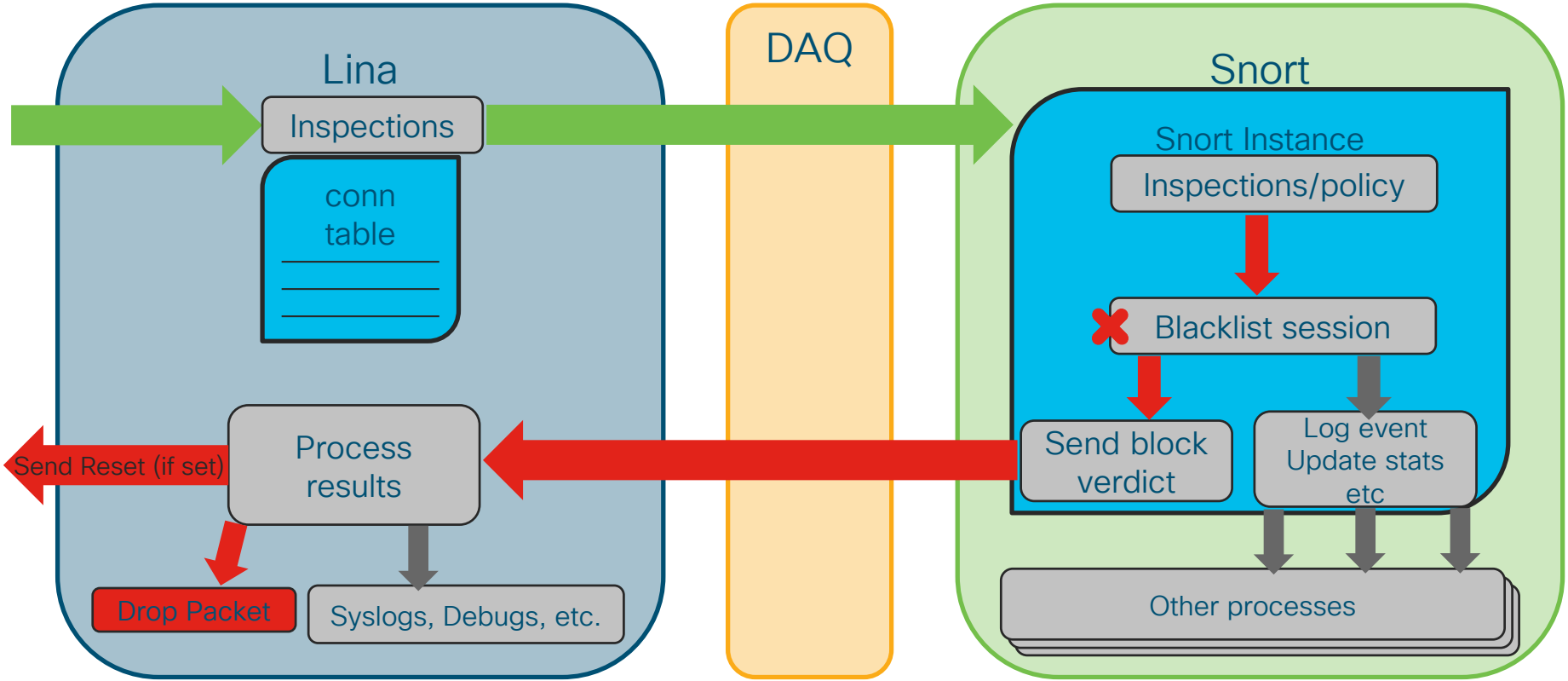


Snort and Lina Interactions

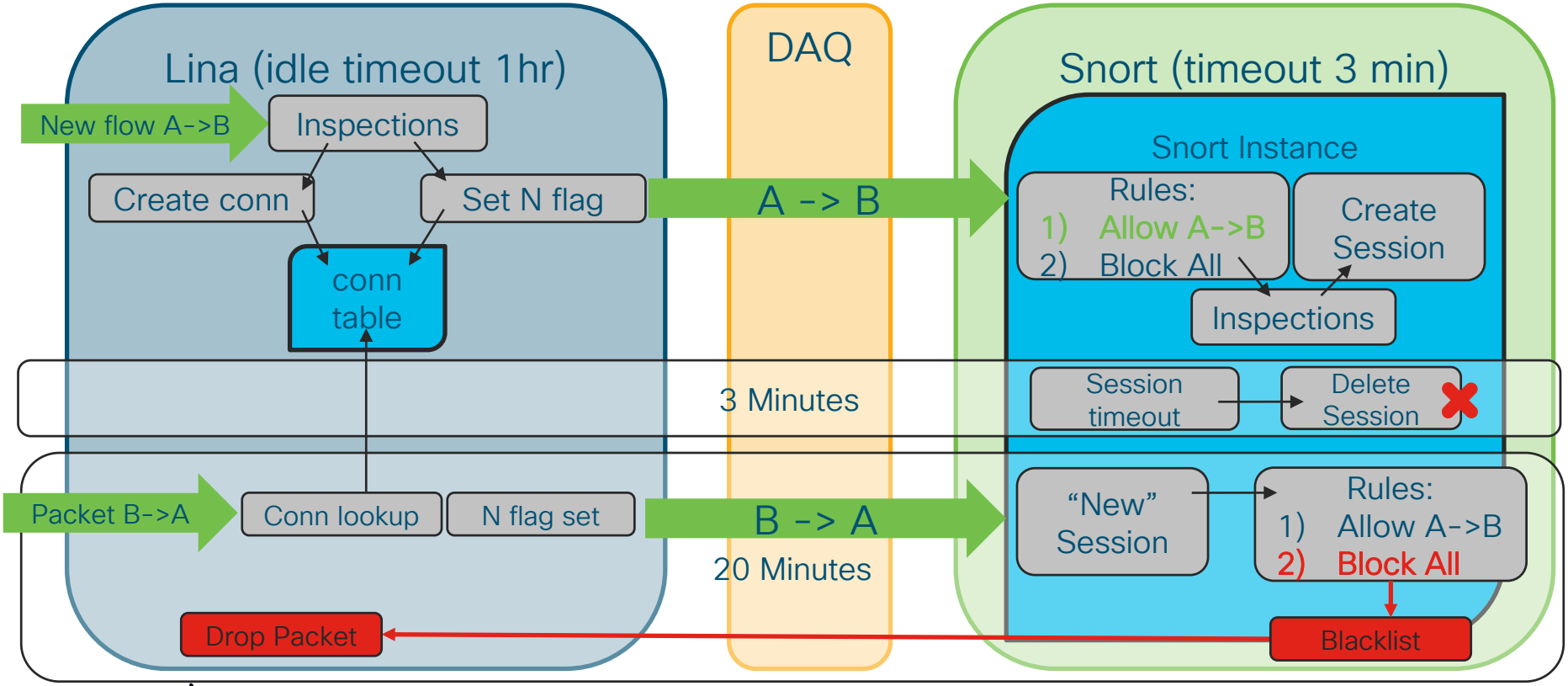
Session Tracking



Example flow – packet blocked by snort



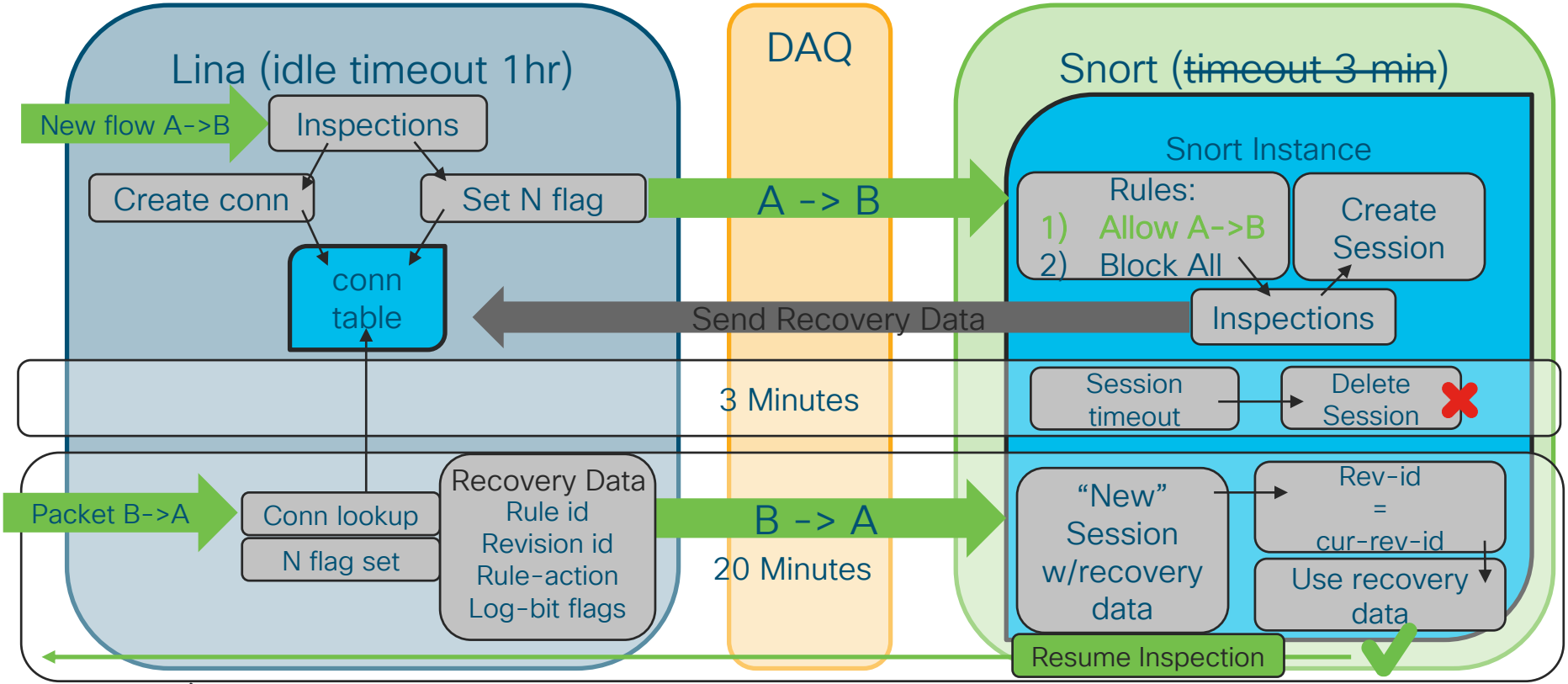
Example conn timeout (TCP) on version < 6.3



Changes in 6.3+ for session tracking lina/snort (TCP Only)

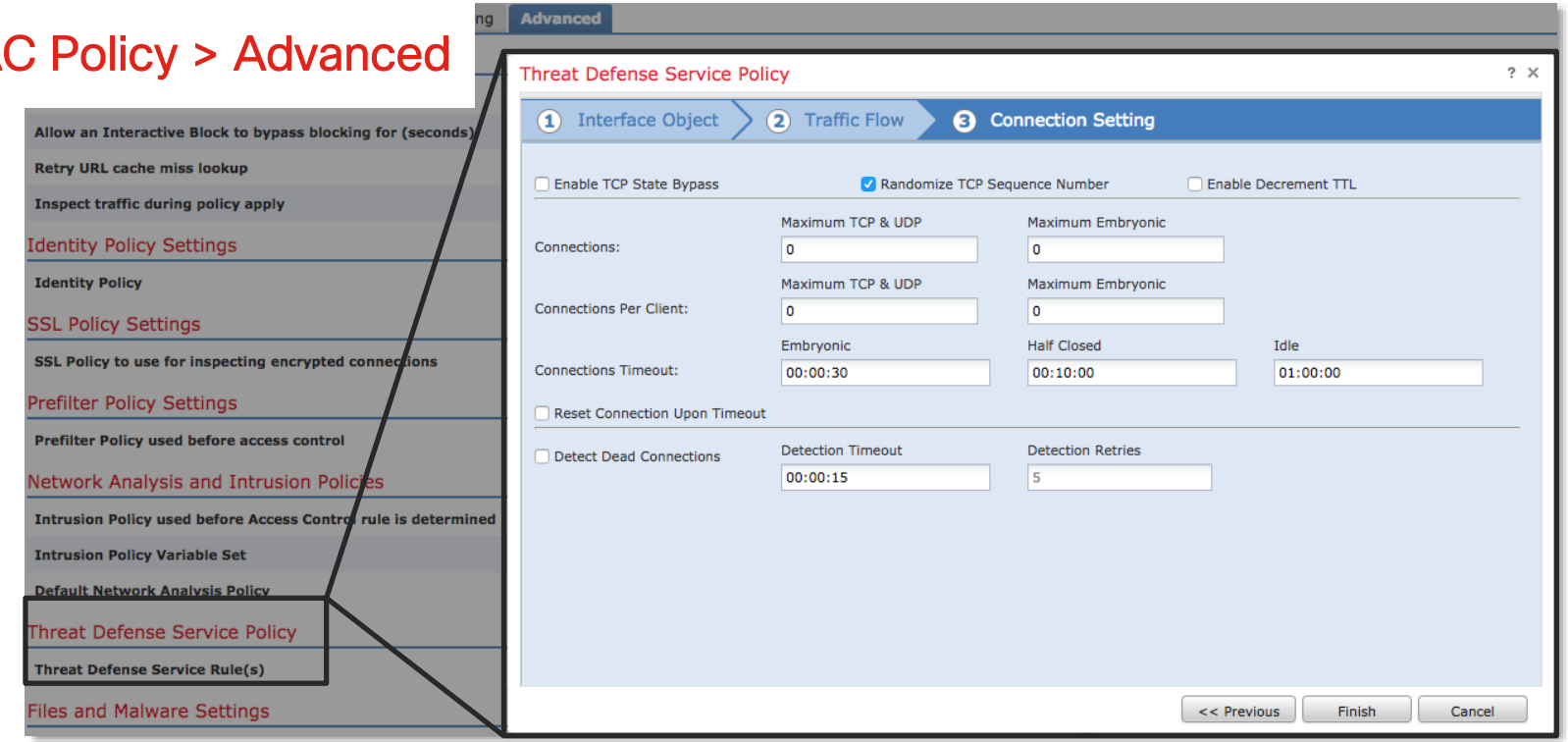
- Lina sets timeouts and syncs them to snort
- Snort sends lina recovery data (RD) for each session
- Lina stores RD in conn-meta
- Snort queries lina for RD if it doesn't know about a session
- Uses recovery data to match AC rule if revision hasn't changed
- When a conn times out in lina, it sends snort End of Flow (EOF) message

Example conn timeout (TCP) on version 6.3+

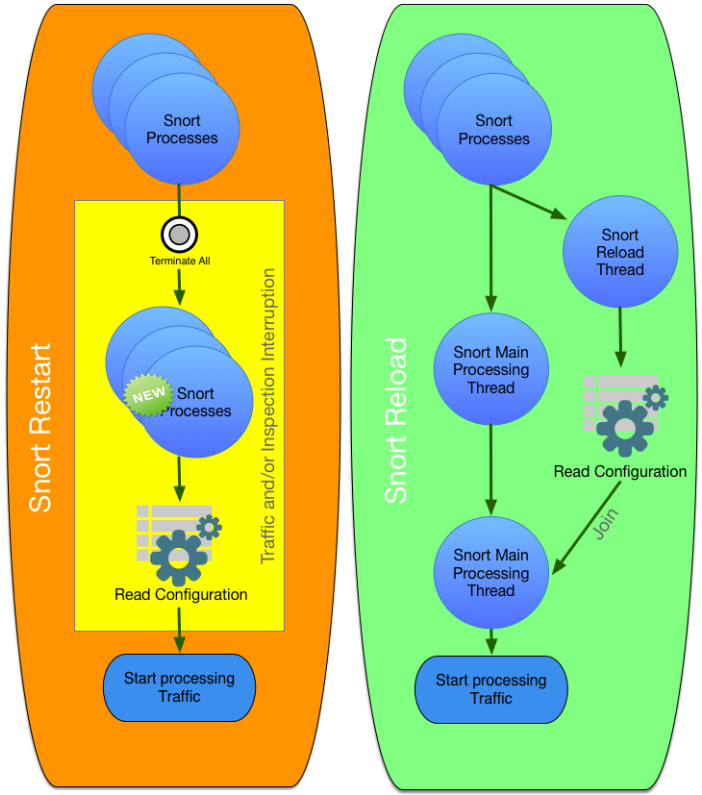


Configure timeouts in 6.3+

AC Policy > Advanced

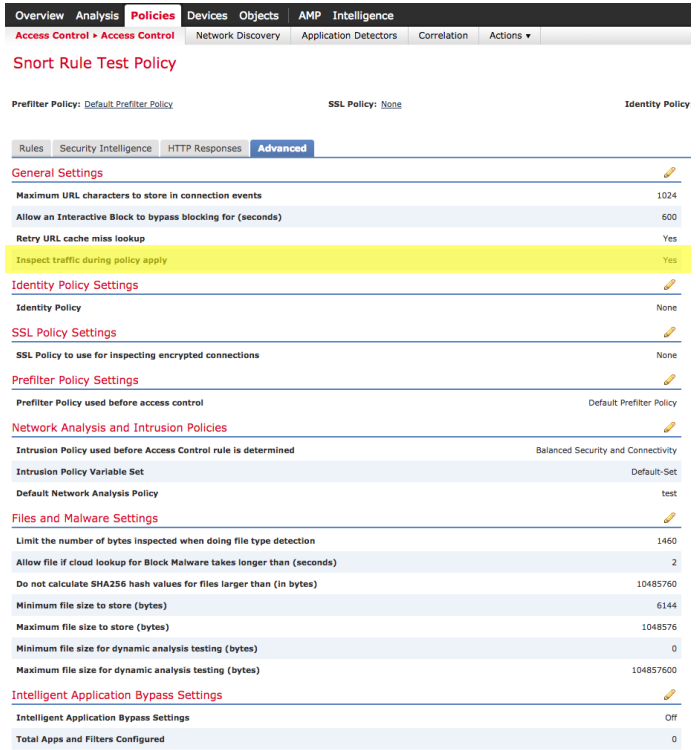


Snort Restart & Reload Architecture



Why does Snort have to restart?

- New version of Snort in policy deploy
- Reallocate memory for pre-processors/Security Intelligence
- Reload shared objects
- Pre-processor configuration changes
- Configured to restart instead of reload



Full listing of restart reasons

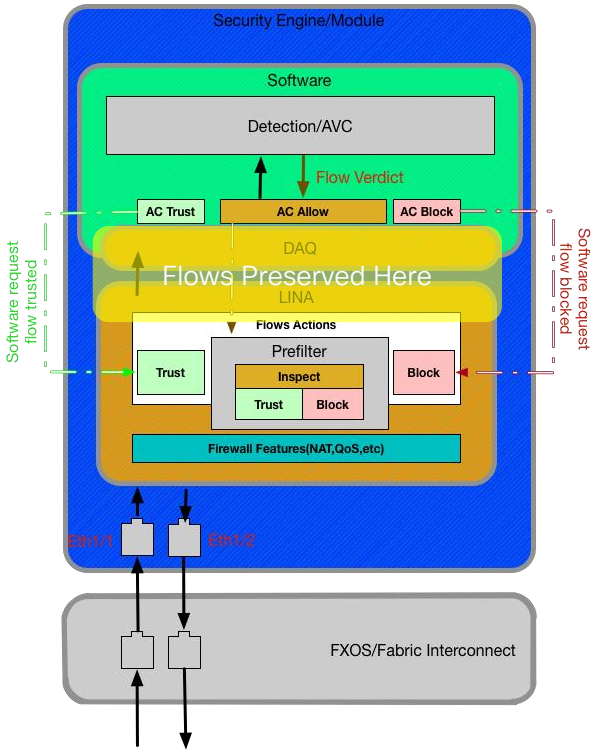
https://www.cisco.com/c/en/us/td/docs/security/firepower/622/configuration/guide/fpmc-config-guide-v622/policy_management.html#concept_33516C5D6B574B6888B1A05F956ABDF9

Mitigations

1. Snort Preserve-Connection
2. Software Bridge

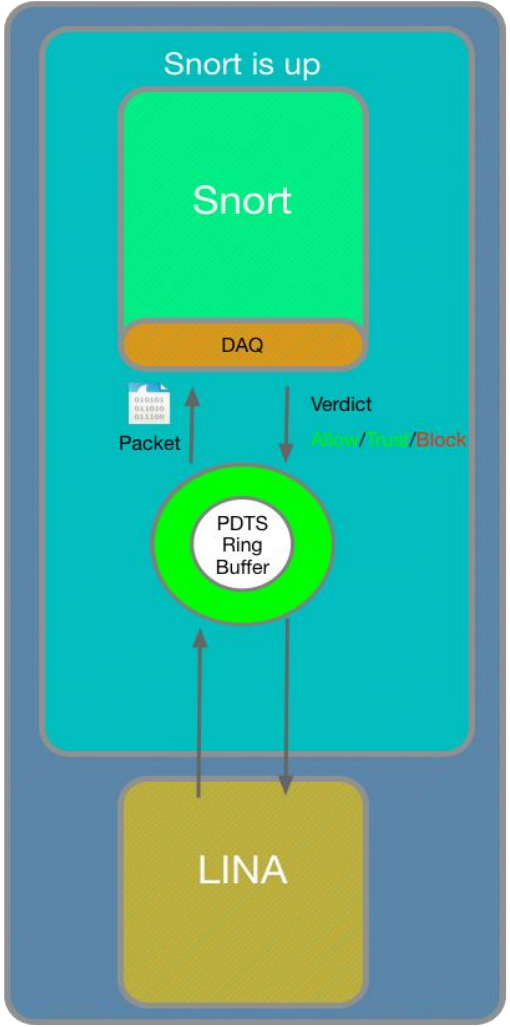
Snort Preserve-Connection

- When Snort goes down connections with Allow verdict are preserved in LINA
- Snort does NOT do a mid-session pickup on preserved flows on coming up
- Does NOT protect against new flows while Snort is down
- 6.2.0.2/6.2.3 Feature Introduction
- Can be enabled/disabled from CLISH: configure snort preserve-connection enable/disable



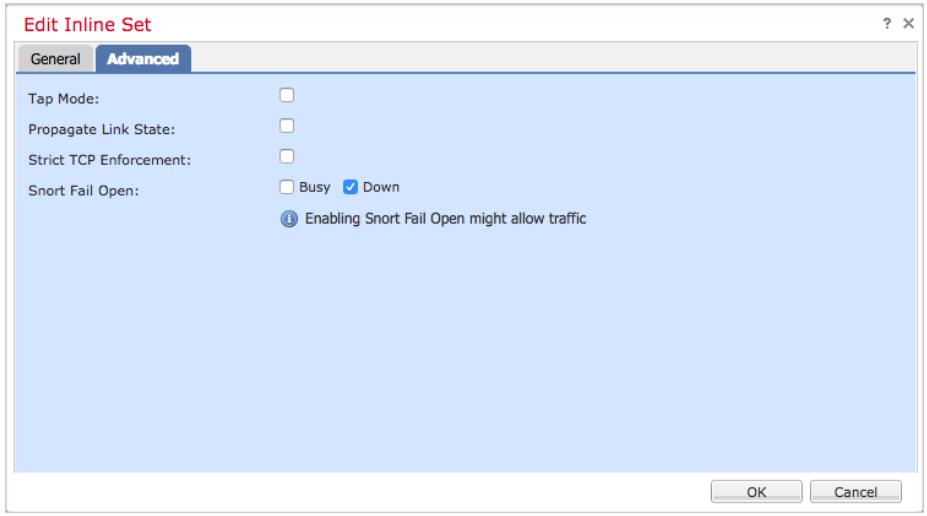
Software Bypass

- With inline Fail-Open deployments traffic is passed uninspected on the Software bridge when Snort is down.
- When Snort comes up, Snort does a mid-session pickup on traffic
- A.K.A Software Bypass
- CLISH Command: > pmtool disablebytype de



Snort Fail-Open when Busy / Down

- Snort fail-open when down means that all traffic will pass over software bridge when snort is down
- Snort fail-open when busy means traffic will be bypassed around Snort when the incoming buffer for snort reaches 85% full



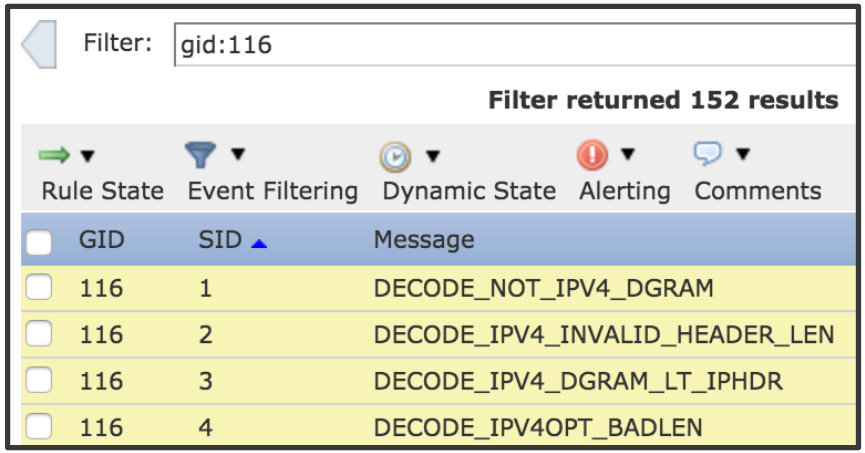
Packet Processing: Decode Preprocessor (GID:116)

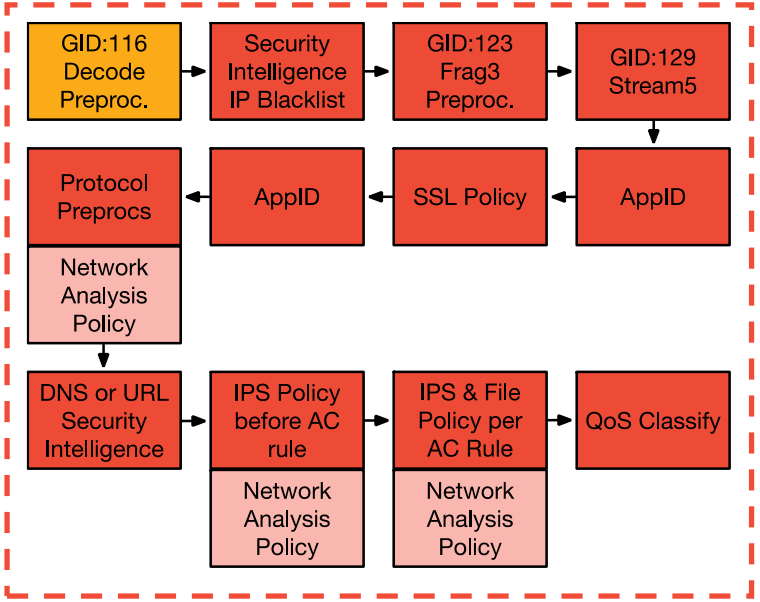
Decode performs basic checks on packets like:

- Confirm Ethernet protocol matches IPv4 or IPv6 value
- Verify IPv4 header is at least 20 bytes

Very rare for Decode to produce unexpected packet drops

Set GID:116 rules to “generate events” for visibility

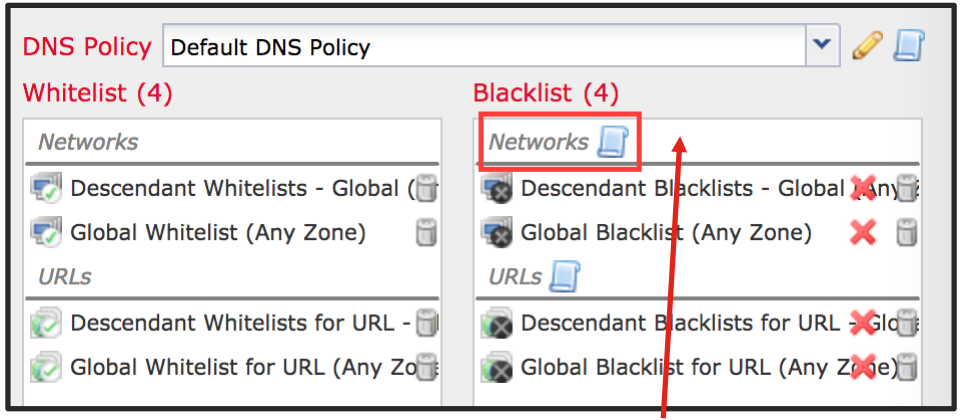




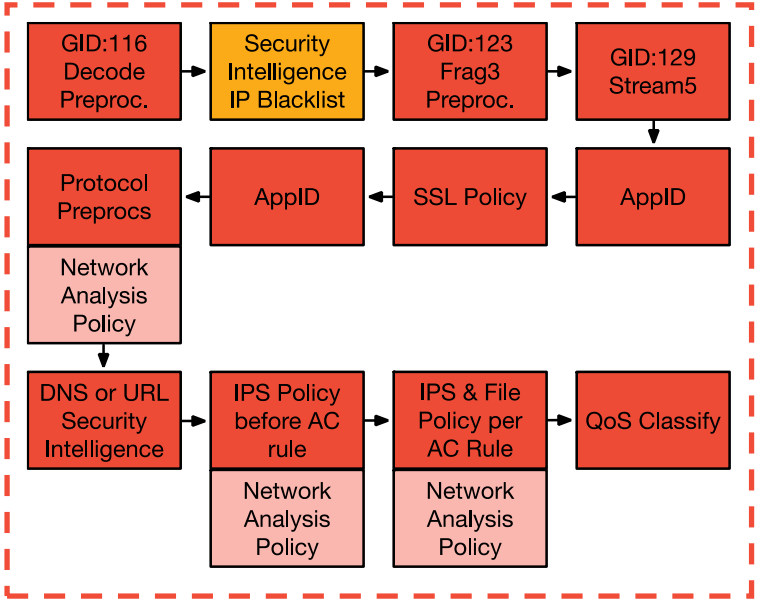
Snort Process Substeps

Packet Processing: IP Security Intelligence

- IP SI drops packets based on lists of malicious IP addresses
- SI drops packets at the IP-level without higher layer inspects
- Whitelist overrides Blacklist



Best Practice: Log SI blacklist events



Snort Process Substeps

Verify an IP is on a blacklist: **`grep -r [IP_ADDRESS] /var/sf/iprep_download`**

Packet Processing: Frag Preprocessor (GID:123)

Frag3 reassembles IP fragments before higher-level preprocs

Rare, but possible causes for drops:
- Zero-byte fragments
- Overlapping fragments

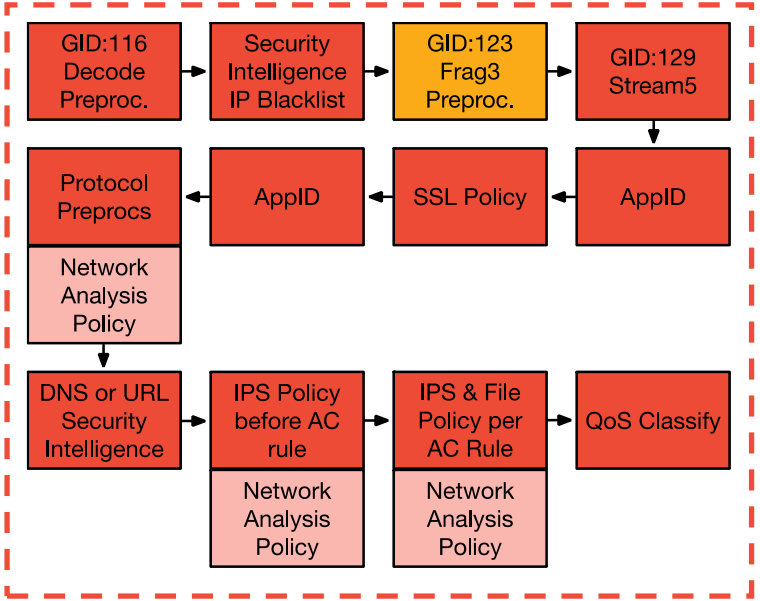
Set GID:123 rules to “generate events” for visibility

Filter: gid:123

Filter returned 11 results

Rule State | Event Filtering | Dynamic State | Alerting | Comments

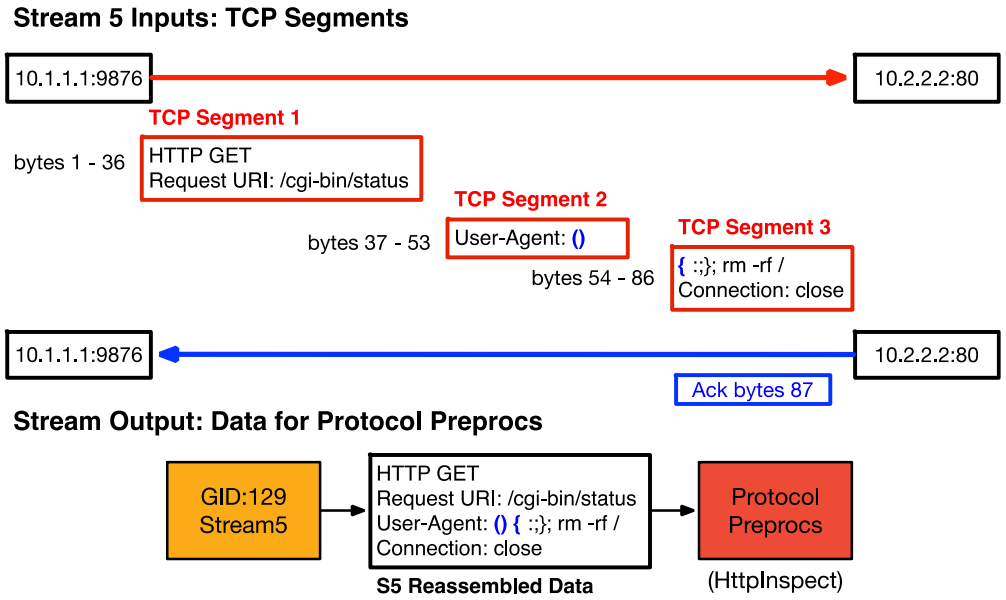
GID	SID	Message
123	13	FRAG3_TINY_FRAGMENT
123	12	FRAG3_EXCESSIVE_OVERLAP
123	11	FRAG3_MIN_TTL

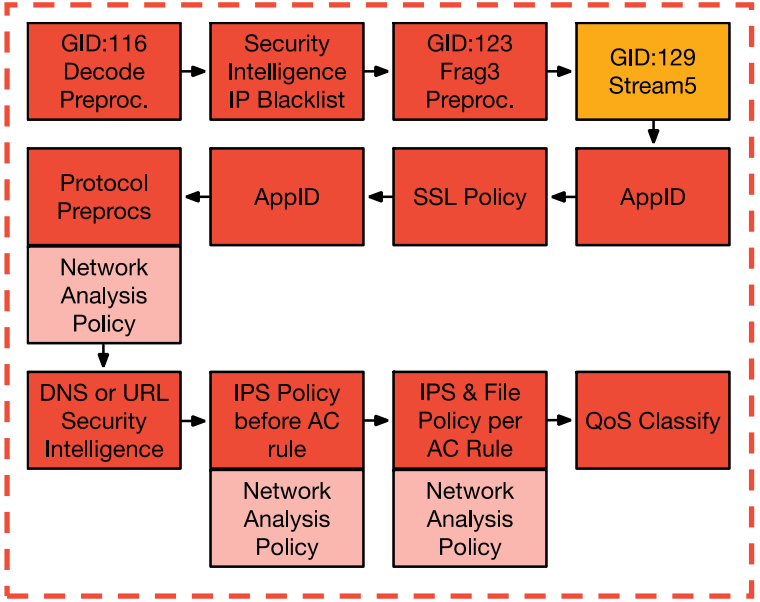


Snort Process Substeps

Packet Processing: Stream Preproc (GID: 129)

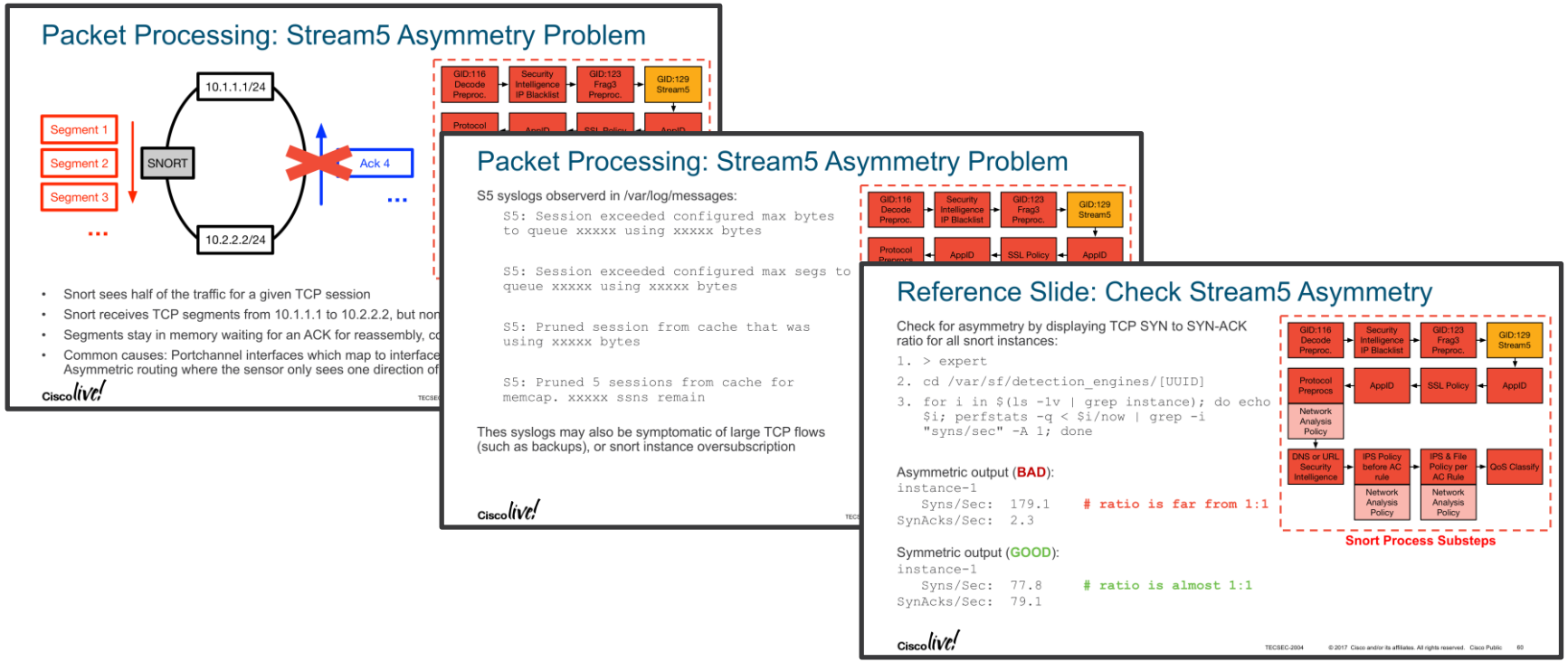
- S5 Reassembles TCP segments for Protocol preprocs
- TCP segments must be contiguous and acknowledged



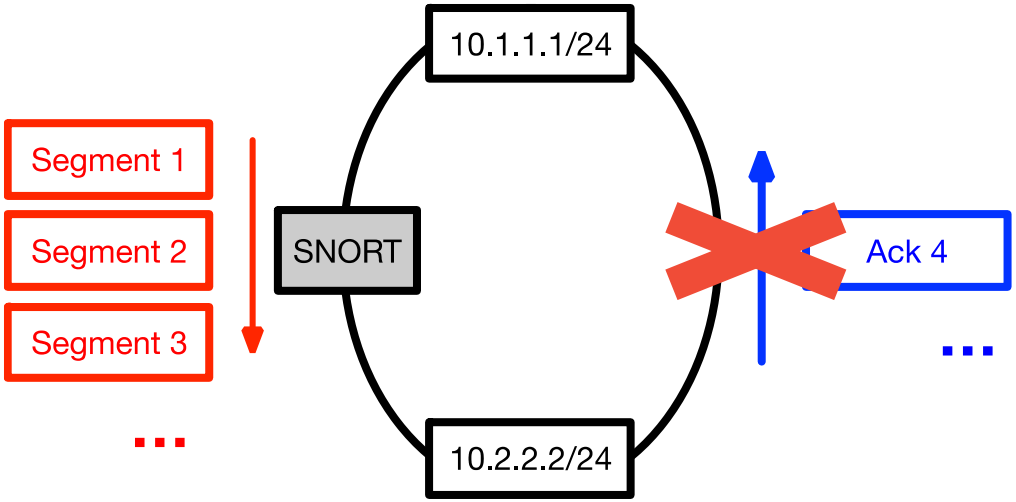


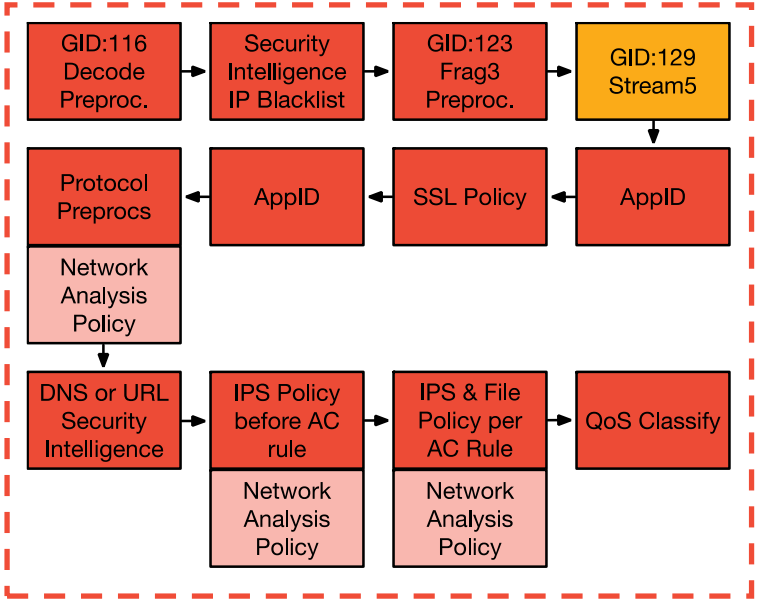
Snort Process Substeps

Stream5 Asymmetric Traffic Reference Slides



Packet Processing: Stream Asymmetry Problem





Snort Process Substeps

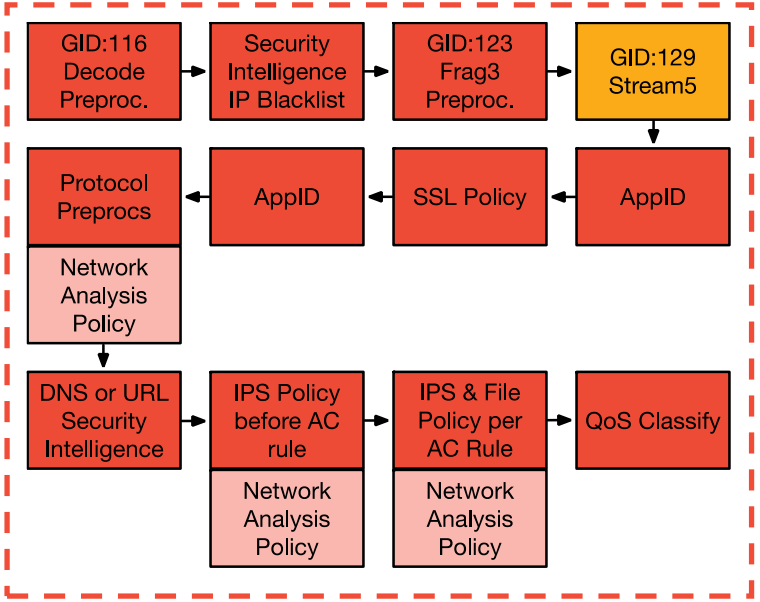
- Snort sees half of the traffic for a given TCP session
- Snort receives TCP segments from 10.1.1.1 to 10.2.2.2, but none of the reply traffic (TCP ACKs)
- Segments stay in memory waiting for an ACK for reassembly, consuming memory until an S5 threshold is hit
- Common causes: Portchannel interfaces which map to interface pairs are not in the same inline set, Asymmetric routing where the sensor only sees one direction of the traffic

Packet Processing: Stream Asymmetry Problem

S5 syslogs observerd in /var/log/messages:

```
S5: Session exceeded configured max bytes
to queue xxxxx using xxxxx bytes

S5: Session exceeded configured max segs
to queue xxxxx using xxxxx bytes

S5: Pruned session from cache that was
using xxxxx bytes

S5: Pruned 5 sessions from cache for
memcap. xxxxx ssns remain
```

Thes syslogs may also be symptomatic of large TCP flows (such as backups), or snort instance oversubscription



Snort Process Substeps

Reference Slide: Check Stream Asymmetry

Check for asymmetry by displaying TCP SYN to SYN-ACK ratio for all snort instances:

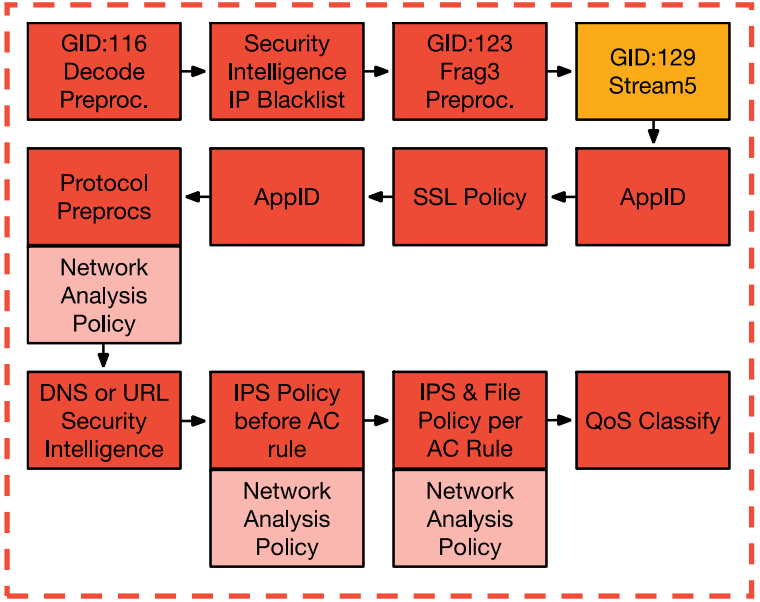
1. `> expert`
2. `cd /var/sf/detection_engines/[UUID]`
3. `for i in $(ls -1v | grep instance); do echo $i; perfstats -q < $i/now | grep -i "syns/sec" -A 1; done`

Asymmetric output (BAD):

```
instance-1
   Syns/Sec:   179.1     # ratio is far from 1:1
SynAcks/Sec:   2.3
```

Symmetric output (GOOD):

```
instance-1
   Syns/Sec:   77.8      # ratio is almost 1:1
SynAcks/Sec:   79.1
```



Snort Process Substeps

Asymmetric Traffic – TAC Script

Snort performance

getS5HostInfo

- Script developed by TAC to get information about asymmetric traffic
- Available currently at: https://github.com/johnjg12/snort-scripts
- Generates CSV files and report files using syslog files (/var/log/messages)
- Hidden slides with details available in presentation PDF

Asymmetric Traffic – TAC Script

Snort performance

Getting state and flag information from a session:

getS5HostInfo --LWstate 0x1 LWFlags 0x2001

LWstate 0x1

```
+==================+ Stream5 State Map +================-=====+

      hex: 0x1
   binary: 00000000 00000001

      ##) 00000000 00000001
          |||||||| ||||||||
      01) |||||||| |||||||+-- STREAM5_STATE_SYN
      02) |||||||| ||||||+--- STREAM5_STATE_SYN_ACK
      03) |||||||| |||||+---- STREAM5_STATE_ACK
      04) |||||||| ||||+----- STREAM5_STATE_ESTABLISHED
      05) |||||||| |||+------ STREAM5_STATE_DROP_CLIENT
      06) |||||||| ||+------- STREAM5_STATE_DROP_SERVER
      07) |||||||| |+-------- STREAM5_STATE_MIDSTREAM
      08) |||||||| +--------- STREAM5_STATE_TIMEDOUT
          ||||||||
      09) |||||||+---------- STREAM5_STATE_UNREACH
          ||||||+
          |||||+
      10) ||||+-------------- STREAM5_STATE_CLOSED
      11) 00000000 00000000-- STREAM5_STATE_NONE
```

LWFlags 0x2001

```
+==================+ Session Flag Map +==================+

      hex: 0x2001
   binary: 00000000 00000000 00100000 00000001

      ##) 00000000 00000000 00100000 00000001
          |||||||| |||||||| |||||||| ||||||||
      11) |||||||| |||||||| |||||||| |||||||+-- SSNFLAG_SEEN_CLIENT
      12) |||||||| |||||||| |||||||| ||||||+--- SSNFLAG_SEEN_SERVER
      13) |||||||| |||||||| |||||||| |||||+---- SSNFLAG_ESTABLISHED
      14) |||||||| |||||||| |||||||| ||||+----- SSNFLAG_NMAP
      15) |||||||| |||||||| |||||||| |||+------ SSNFLAG_ECN_CLIENT_QUERY
      16) |||||||| |||||||| |||||||| ||+------- SSNFLAG_ECN_SERVER_REPLY
      17) |||||||| |||||||| |||||||| |+-------- SSNFLAG_HTTP_1_1
      18) |||||||| |||||||| |||||||| +--------- SSNFLAG_SEEN_PMATCH
          |||||||| |||||||| ||||||||
      19) |||||||| |||||||| |||||||+---------- SSNFLAG_MIDSTREAM
      11) |||||||| |||||||| ||||||+----------- SSNFLAG_CLIENT_FIN
      11) |||||||| |||||||| |||||+------------ SSNFLAG_SERVER_FIN
      12) |||||||| |||||||| ||||+------------- SSNFLAG_CLIENT_PKT
      13) |||||||| |||||||| |||+-------------- SSNFLAG_SERVER_PKT
      14) |||||||| |||||||| ||+--------------- SSNFLAG_COUNTED_INITIALIZE
      15) |||||||| |||||||| |+---------------- SSNFLAG_COUNTED_ESTABLISH
      16) |||||||| |||||||| +----------------- SSNFLAG_COUNTED_CLOSING
          |||||||| ||||||||
      17) |||||||| |||||||+------------------- SSNFLAG_TIMEDOUT
      18) |||||||| ||||||+-------------------- SSNFLAG_PRUNED
      19) |||||||| |||||+--------------------- SSNFLAG_RESET
      20) |||||||| ||||+---------------------- SSNFLAG_DROP_CLIENT
      21) |||||||| |||+----------------------- SSNFLAG_DROP_SERVER
      22) |||||||| ||+------------------------ SSNFLAG_LOGGED_QUEUE_FULL
      23) |||||||| |+------------------------- SSNFLAG_STREAM_ORDER_BAD
      22) |||||||| +-------------------------- SSNFLAG_FORCE_BLOCK
          ||||||||
      23) |||||||+--------------------------- SSNFLAG_CLIENT_SWAP
      22) ||||||+---------------------------- SSNFLAG_CLIENT_SWAPPED
      27) 11111111 11111111 11111111 11111111-- SSNFLAG_ALL
      28) 00000000 00000000 00000000 00000000-- SSNFLAG_NONE
```

cisco Live!

Asymmetric Traffic – TAC Script

Snort performance

Generate CSV file
  Single File
  getS5HostInfo --csv messages

  Aggregating data from all messages files
  egrep –h “S5:” messages* >> s5Mes.out
  getS5HostInfo --csv s5Mes.out

Generate summary file
getS5HostInfo --summary <output_csv_file>

Asymmetric Traffic

getS5HostInfo --summary myfile.csv

```
Top 5 sessions for syn_no_syn_ack:
1 ) 10.0.0.154      -> 172.16.235.229 : 4101
2 ) 10.0.0.154      -> 172.16.234.48  : 4041
3 ) 10.0.1.14       -> 172.16.235.67  : 4031
4 ) 10.0.8.19       -> 172.16.235.67  : 3882
5 ) 10.0.0.15       -> 172.16.235.67  : 3881

Top 5 sessions for syn_ack_no_syn:
1 ) 192.168.1.50    -> 172.16.235.229 : 3267
2 ) 192.168.3.5     -> 172.16.234.48  : 2653
3 ) 192.168.2.105   -> 172.16.235.67  : 2452
4 ) 192.168.12.107  -> 172.16.235.67  : 2445
5 ) 192.168.12.106  -> 172.16.235.67  : 2221


Top 5 sessions for client_no_server_count:
1 ) 10.0.0.154      -> 172.16.235.229 : 4116
2 ) 10.0.0.154      -> 172.16.234.48  : 4051
3 ) 10.0.1.14       -> 172.16.235.67  : 4047
4 ) 10.0.8.19       -> 172.16.235.67  : 3894
5 ) 10.0.0.15       -> 172.16.235.67  : 3892

Top 50 sessions for server_no_client_count:
1 ) 192.168.1.50    -> 172.16.235.229 : 4443
2 ) 192.168.3.5     -> 172.16.234.48  : 3875
3 ) 192.168.2.105   -> 172.16.235.67  : 2532
4 ) 192.168.12.107  -> 172.16.235.67  : 430
5 ) 192.168.12.106  -> 172.16.235.67  : 417
```

```
Top Source ports seen:
Port : Times Seen
1 ) 20       : 8351
2 ) 60020    : 202
3 ) 38887    : 59
4 ) 39792    : 27
5 ) 33376    : 25
6 ) 33920    : 24
7 ) 34034    : 23
8 ) 41337    : 23
9 ) 42414    : 22
10) 57264    : 22
11) 46752    : 22
12) 46778    : 22
13) 47648    : 22
14) 33956    : 22
15) 42369    : 22


Top Dest ports seen:
Port : Times Seen
1 ) 25       : 87349
2 ) 60020    : 41244
3 ) 10051    : 7422
4 ) 8443     : 3855
5 ) 2058     : 3573
6 ) 2003     : 2942
7 ) 80       : 2269
8 ) 4242     : 1980
9 ) 1044     : 1472
10) 22       : 1386
11) 8200     : 1126
12) 1935     : 1057
13) 9997     : 934
14) 1031     : 876
15) 445      : 722


Top Application Protocols Seen:
AppID : Times Seen
1 ) 8        : 76845
2 ) 25       : 42301
3 ) 5        : 13566
4 ) 169      : 7529
5 ) 9        : 1443
6 ) 34       : 1232
7 ) 28       : 1126
8 ) 174      : 771
9 ) 13       : 716
10) 115      : 462
11) 101      : 138
12) 184      : 7
13) 114      : 3
14) 170      : 3
```

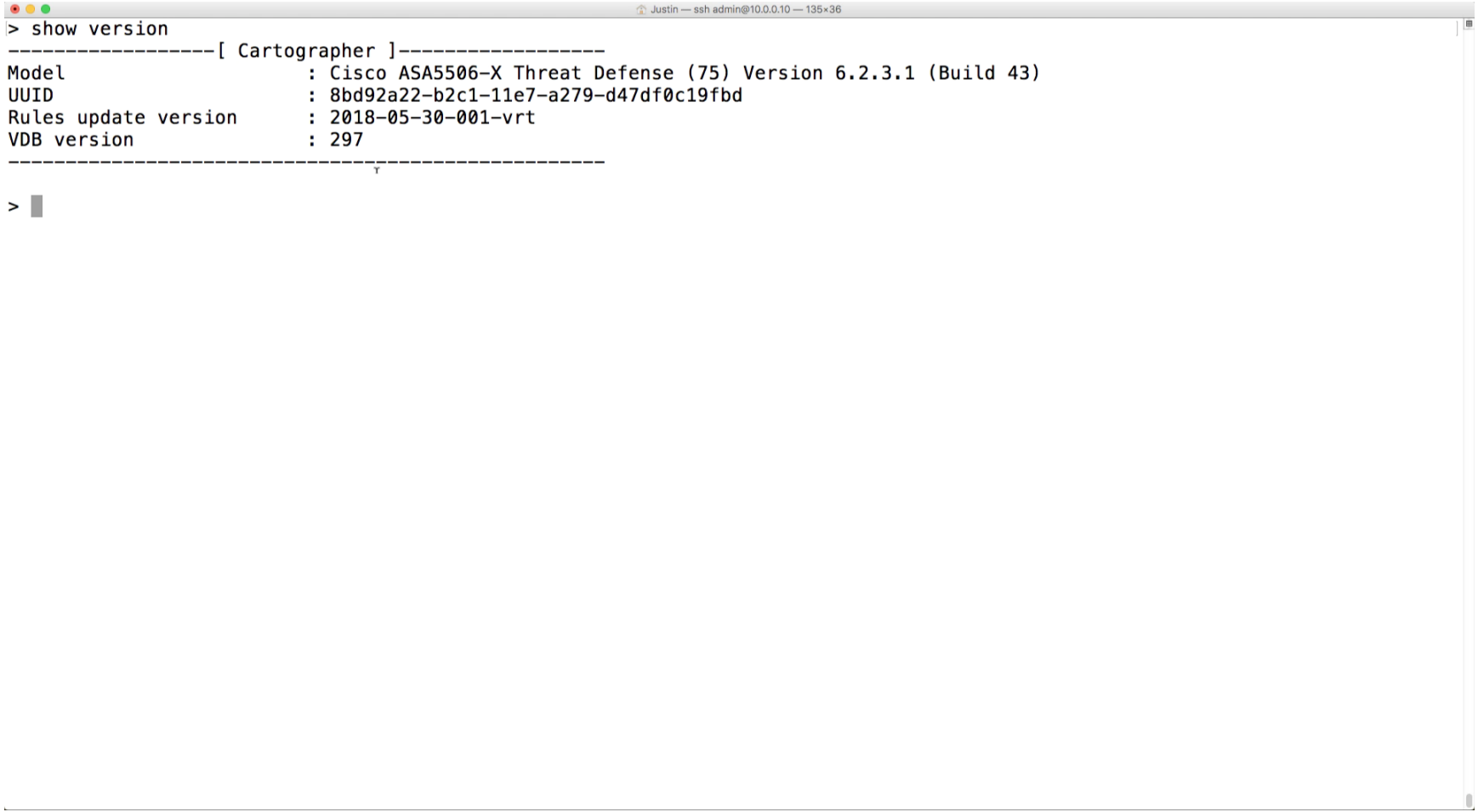
```
                  Stream5 State Summary

Time: 2015-05-31 04:02:03 - 2015-06-11 16:15:03
Total Sessions           :  258866
Saw Client but not Server:  229927 (88%)
Saw Server but not Client:  18544 (7%)
Saw Client and Server    :  10395 (4%)
STREAM5_STATE_ESTABLISHED:  27691 (10.70%)
STREAM5_STATE_DROP_CLIENT:  0 (0.00%)
STREAM5_STATE_DROP_SERVER:  0 (0.00%)
STREAM5_STATE_MIDSTREAM  :  2466 (0.95%)
STREAM5_STATE_TIMEDOUT   :  729 (0.28%)
STREAM5_STATE_UNREACH    :  0 (0.00%)
STREAM5_STATE_CLOSED     :  0 (0.00%)


                  Session Flags Summary

Time: 2015-05-31 04:02:03 - 2015-06-11 16:15:03
Total Sessions           :  258866
Saw SYN but not SYN_ACK  :  228813 (88%)
Saw SYN_ACK but not SYN  :  19524 (7%)
Saw SYN and SYN_ACK      :  8126 (3%)
Saw 3-Way Handshake      :  8126 (3%)
SSNFLAG_ESTABLISHED      :  44 (0.02%)
SSNFLAG_NMAP             :  0 (0.00%)
SSNFLAG_ECN_CLIENT_QUERY :  1 (0.00%)
SSNFLAG_ECN_SERVER_REPLY :  0 (0.00%)
SSNFLAG_HTTP_1_1         :  0 (0.00%)
SSNFLAG_SEEN_PMATCH      :  0 (0.00%)
SSNFLAG_MIDSTREAM        :  2466 (0.95%)
SSNFLAG_CLIENT_FIN       :  0 (0.00%)
SSNFLAG_SERVER_FIN       :  0 (0.00%)
SSNFLAG_CLIENT_PKT       :  0 (0.00%)
SSNFLAG_SERVER_PKT       :  0 (0.00%)
SSNFLAG_COUNTED_INITIALIZE:  258776 (99.97%)
SSNFLAG_COUNTED_ESTABLISH :  27690 (10.70%)
SSNFLAG_COUNTED_CLOSING   :  88557 (34.21%)
SSNFLAG_TIMEDOUT          :  256338 (99.02%)
SSNFLAG_PRUNED            :  2483 (0.96%)
SSNFLAG_RESET             :  63762 (24.63%)
SSNFLAG_DROP_CLIENT       :  0 (0.00%)
SSNFLAG_DROP_SERVER       :  0 (0.00%)
SSNFLAG_LOGGED_QUEUE_FULL :  249859 (96.52%)
SSNFLAG_STREAM_ORDER_BAD  :  81368 (31.43%)
SSNFLAG_FORCE_BLOCK       :  0 (0.00%)
SSNFLAG_CLIENT_SWAP       :  68 (0.03%)
SSNFLAG_CLIENT_SWAPPED    :  68 (0.03%)
```

cisco Live!

Story Time!

	First Packet	Last Packet	Action	Initiator IP	Responder IP	Ingress Security Zone	Source Port / ICMP Type	Destination Port / ICMP Code
	2018-06-05 19:33:32	2018-06-05 19:35:33	Allow	192.168.0.4	8.8.8.8	Passive	12755 / udp	53 (domain) / udp
	2018-06-05 19:33:32	2018-06-05 19:35:33	Allow	8.8.8.8	192.168.0.4	Passive	53 (domain) / udp	12755 / udp
	2018-06-05 19:33:32	2018-06-05 19:35:33	Allow	8.8.8.8	192.168.0.4	Passive	53 (domain) / udp	12434 / udp
	2018-06-05 19:33:32	2018-06-05 19:35:33	Allow	192.168.0.4	8.8.8.8	Passive	12434 / udp	53 (domain) / udp

Page 1 of 1 Displaying rows 1–4 of 4 rows



Asymmetric Traffic – Common Problems

Configuration options

Problem:

Different VLANs on each side of session

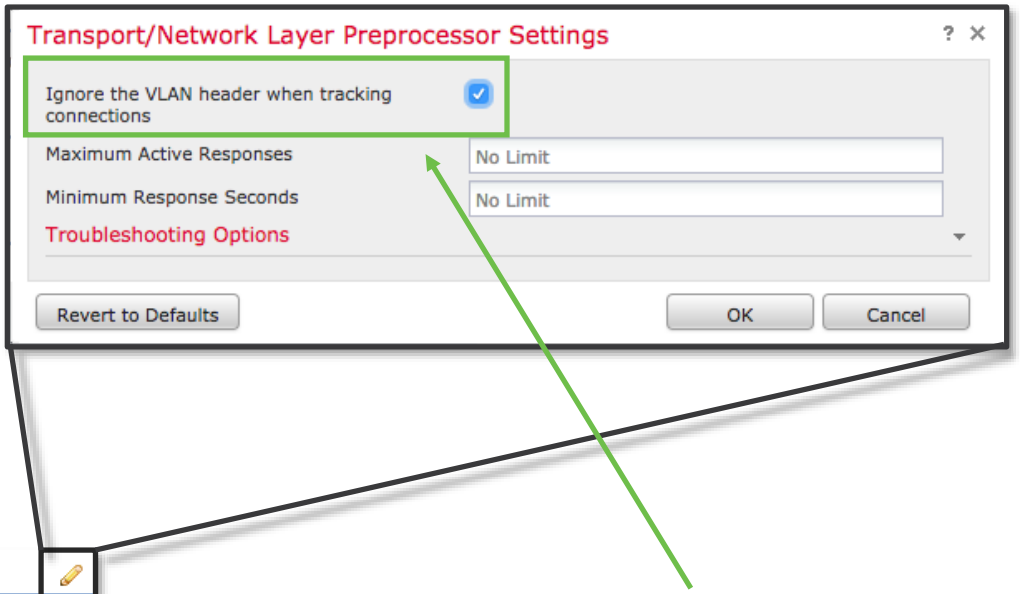
Example:

(VLAN50) 192.168.1.2 -> 10.8.0.2

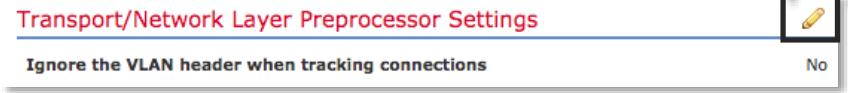
(VLAN51) 10.8.0.2 -> 192.168.1.2

Fix:

Enable VLAN agnostic mode



Access Control Policy Advanced tab



Check this box to ignore VLANs when identifying unique sessions

Asymmetric Traffic – Common Problems

Configuration options

Problem:

Traffic from same session traversing multiple Inline sets

Example:

Inline set A 192.168.1.2 -> 10.8.0.2

Inline set B 10.8.0.2 -> 192.168.1.2

Fix:

Combine pairs into single inline set

Devices > Device Management [Edit device]

Separate inline sets

Device	Interfaces	Inline Sets

Name	Interface Pairs
Inline Set A	s1p1 ↔ s1p2
Inline Set B	s1p3 ↔ s1p4

Single inline set

Device	Interfaces	Inline Sets

Name	Interface Pairs
Single Inline Set	s1p1 ↔ s1p2, s1p3 ↔ s1p4

Asymmetric Traffic – Common Problems

Configuration options

Problem:

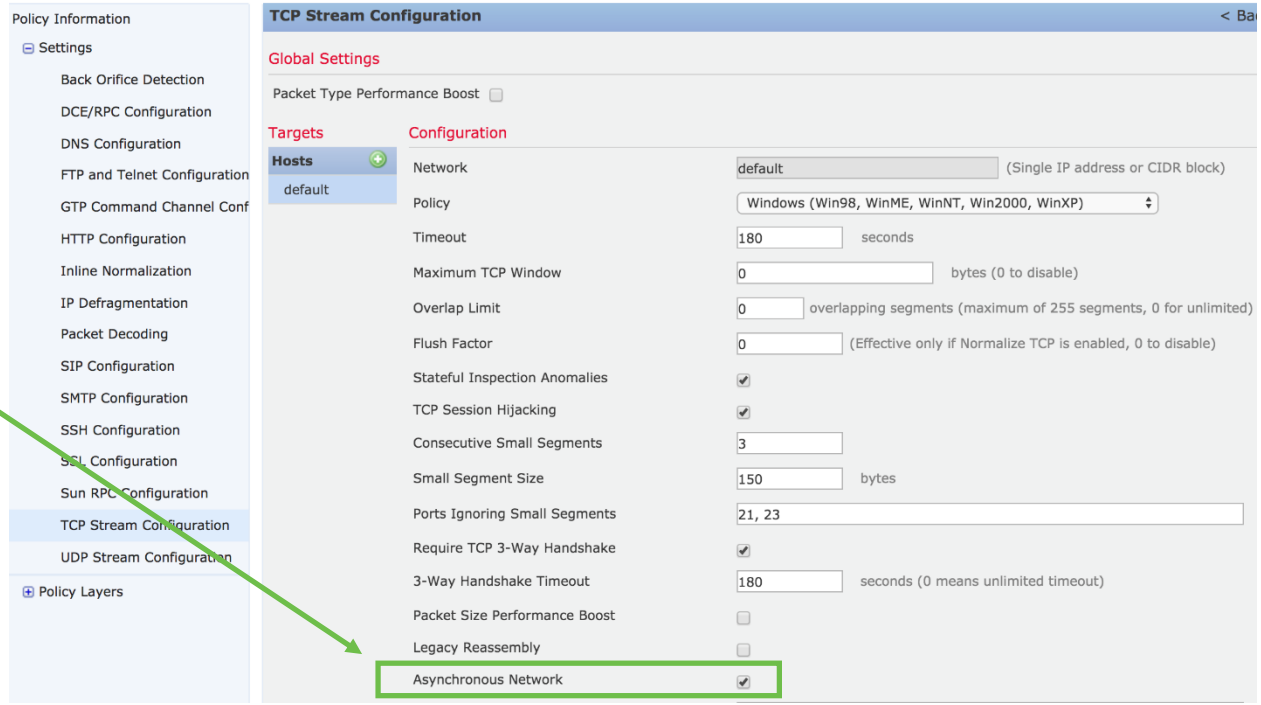
Traffic is actually asymmetric

Fix:

Configure network or move device so that there is no asymmetric traffic

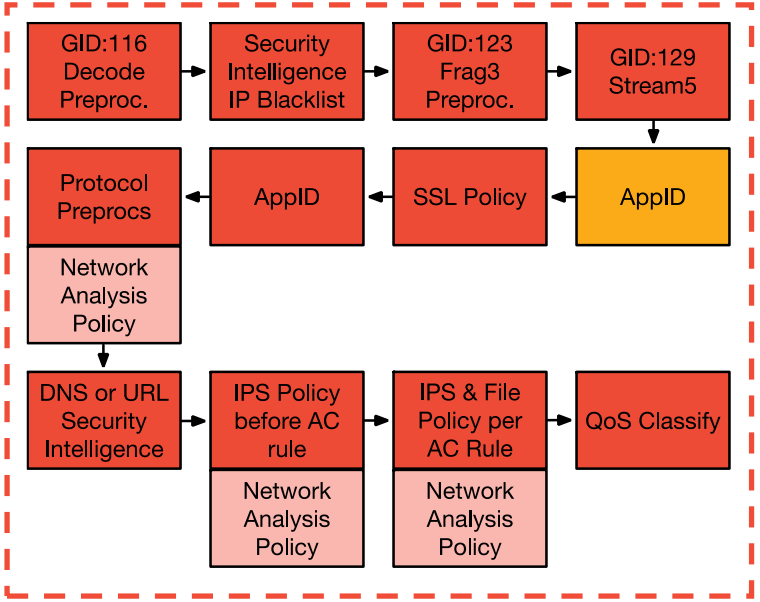
Mitigation:

Enable Asynchronous Network option in NAP > TCP Stream Configuration



Packet Processing: AppID

- AppID identifies over 3,000 layer 7 network applications:
  - Facebook, Facebook chat, Facebook games, etc
- AppID runs both before and after SSL decryption
- AppID does not drop traffic
- An incorrect AppID disposition can cause traffic to match the wrong access control rule



Packet Processing: AppID Debugging

- Specify flow 5-tuple of a flow to see application matching:

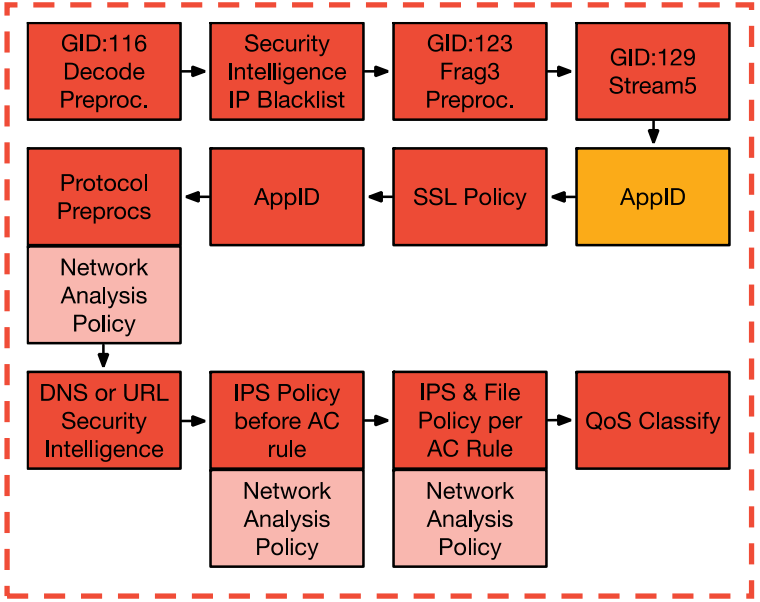
```
> system support application-identification-debug
```

Output:

AS address space
I snort instance number
R 1st packet reversed (server to client)

- Specify flow 5-tuple to show access control rule matching:

```
> system support firewall-engine-debug
```



Snort Process Substeps

Packet Processing: SSL Policy

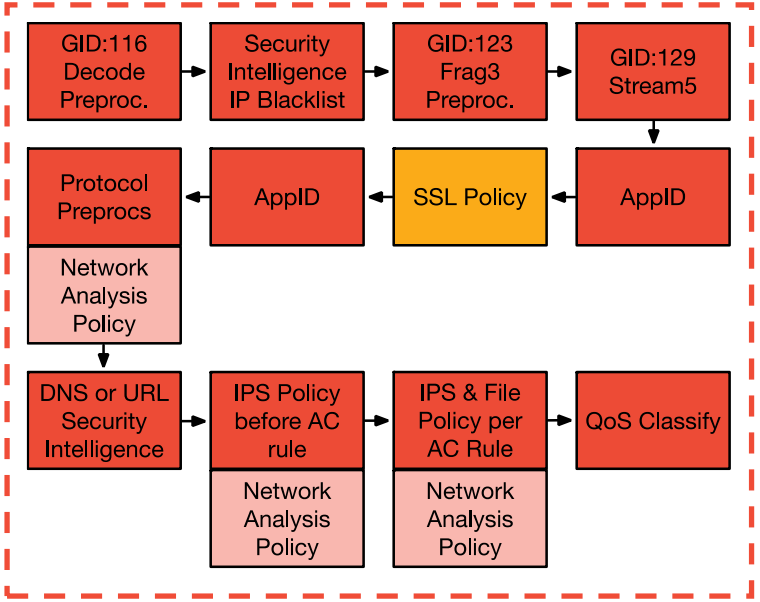
An authorized man-in-the-middle of TLS/SSL traffic

For servers you own:
Decrypt: Known Key - Requires private key and certificate

For clients navigating to 3rd party sites:
Decrypt: Resign - Resign certificate with an intermediate CA

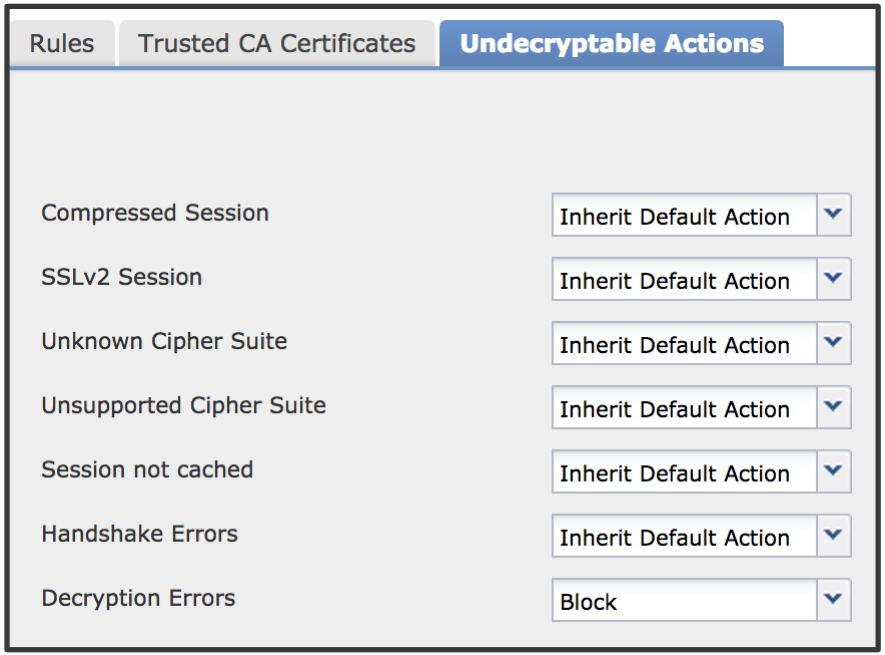
Two options for new certificates to be trusted:
1. The client must trust the FMC as a Certificate Authority
2. The client must trust a CA which signs the FMC's CSR (Certificate Signing Request)

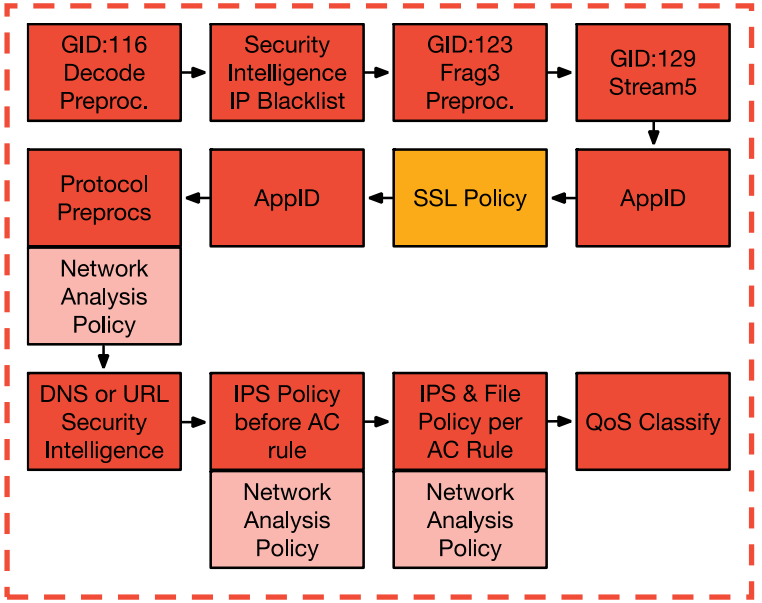
Traffic is TCP and SSL proxied in a DAQ extension which sends cleartext traffic to snort for IPS inspection



Packet Processing: SSL Policy Debugging

Be careful with “Undecryptable Actions,” especially if your default action in the SSL Policy rules is “Block”





Snort Process Substeps

Packet Processing: SSL Policy Debugging

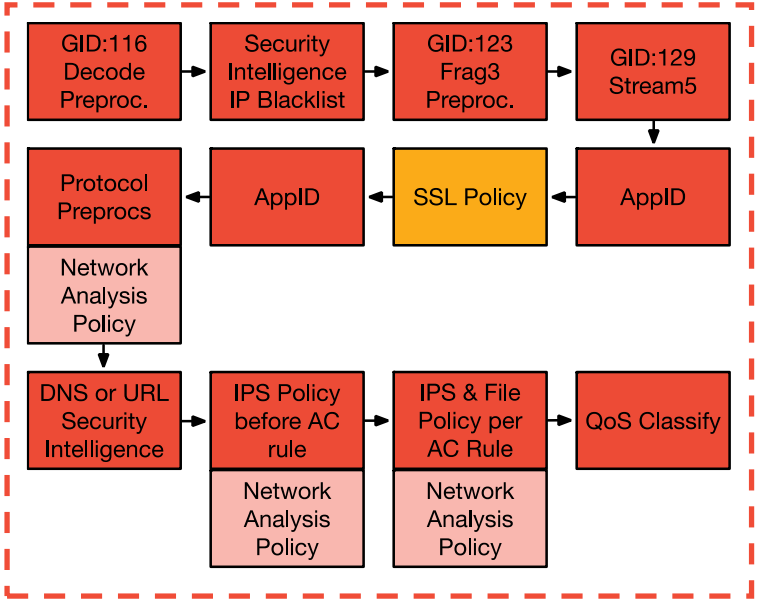
Troubleshooting Best Practices:

1) Take note of browser side errors!

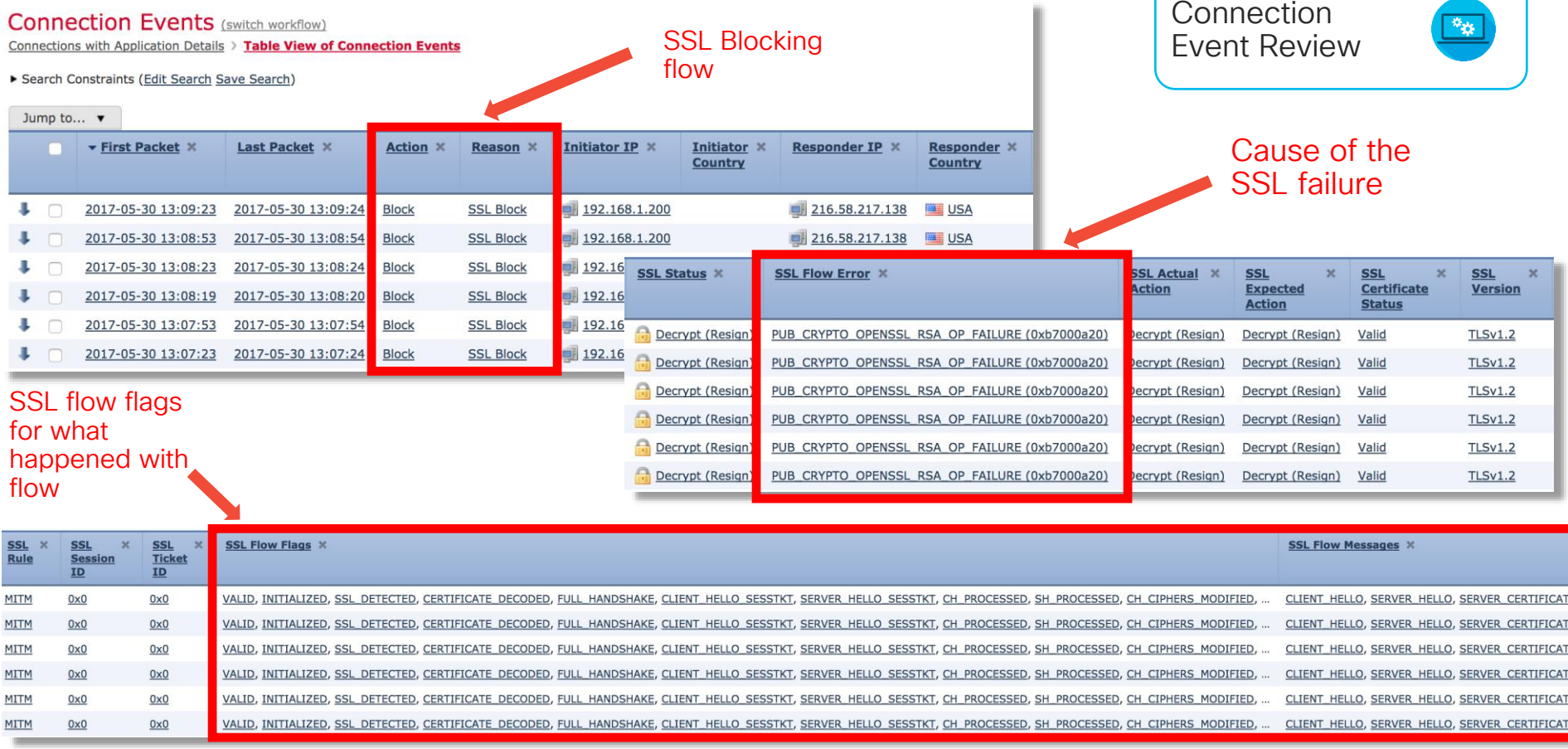
2) View SSL decryption event columns in Connection Events:

- Navigate to “Analysis > Connections > Events”
- Click “Table View of Connection Events”
- Click “X” next to any column and select 13 SSL columns

3) Columns in connection events explain decryption errors

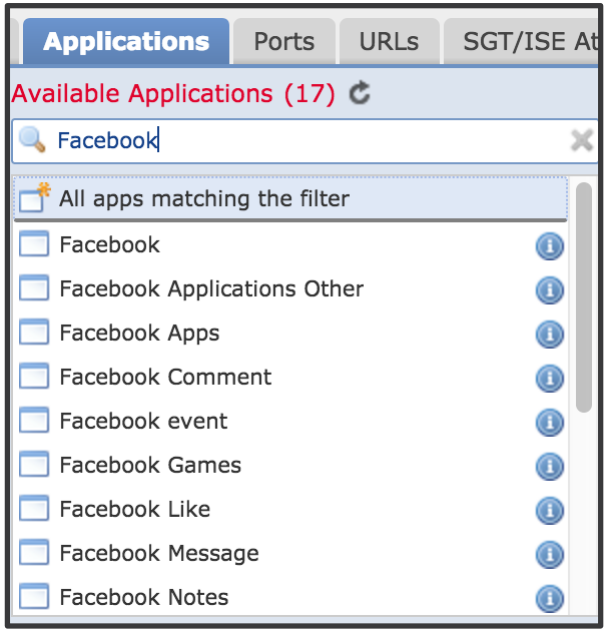


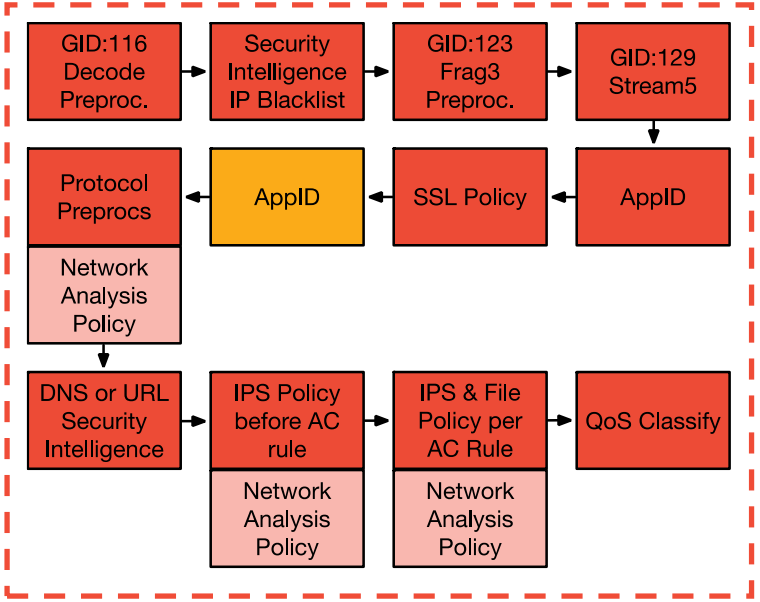
Connection Event Review



Packet Processing: AppID (Post SSL Decryption)

- Some apps require SSL decryption for further differentiation

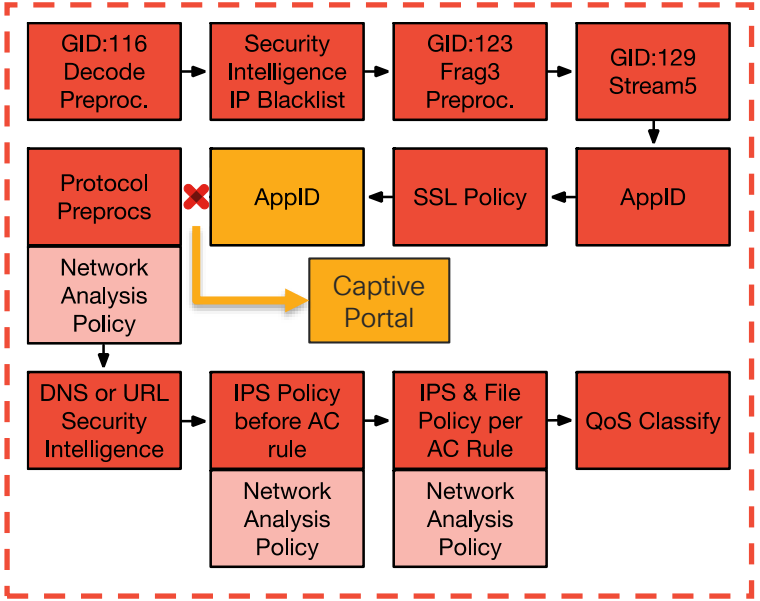




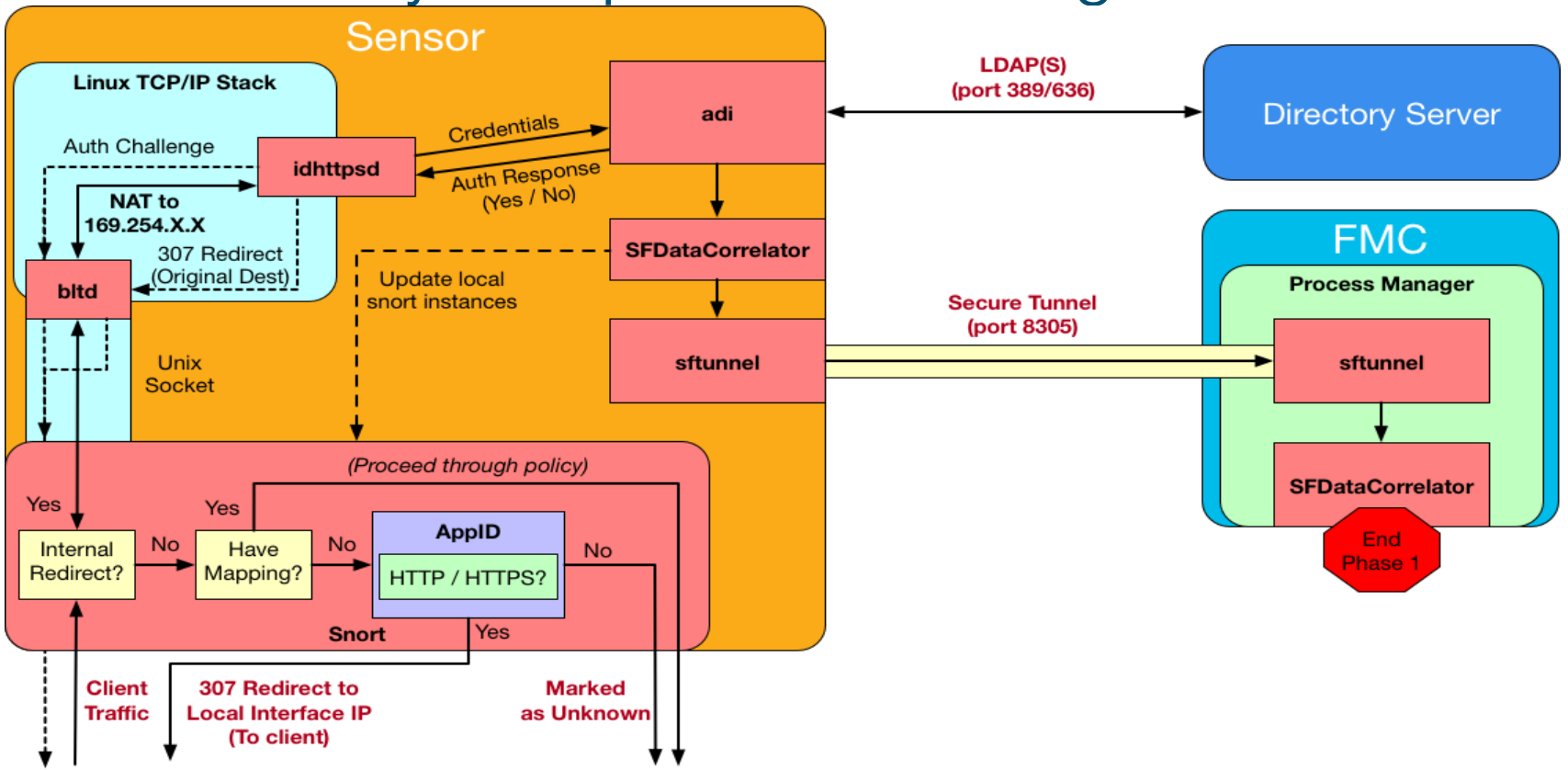
Snort Process Substeps

Packet Processing: Captive Portal

- Will only act if traffic is identified as HTTP or HTTPS
- Evaluation point to see if a user mapping currently exists for this IP address
- Intercepts client traffic and forces them to authenticate if there is no active mapping



User Identity – Captive Portal Diagram



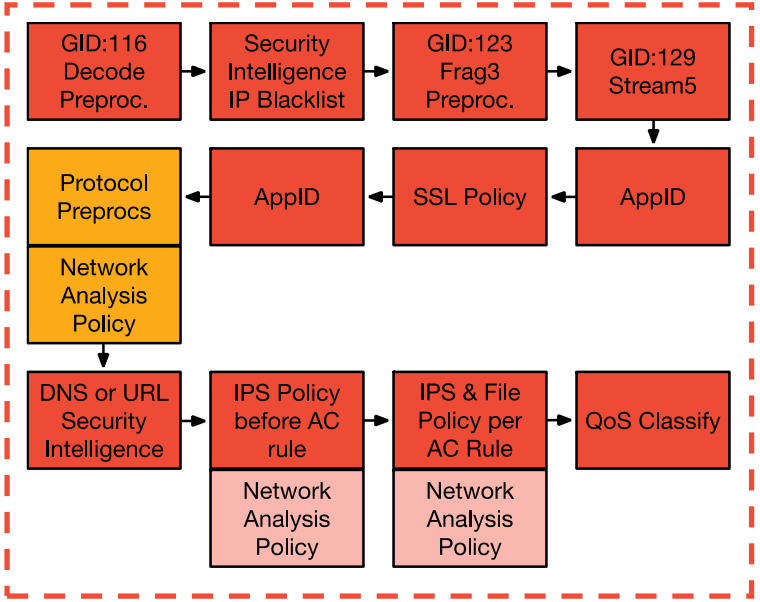
Captive Portal new session walkthrough

1. Client traffic (after coming from the data plane) makes its way to Snort
2. Check for current mappings for the requesting IP address
3. If no mapping, traffic eventually makes it into AppID portion of Snort
4. Traffic is identified as HTTP/HTTPS snort injects a 307 response to client, redirecting them to the sensors interface IP
5. Traffic destined to the sensors local IP forces a flag to be set on the packet that instructs Snort to send this over to bltd
6. The response from the client is sent over to the bltd process via a Unix socket
7. bltd NATs the traffic to a 169.254.X.X IP address to be able to talk to the idhttpsd process
8. idhttpsd receives the GET request from the client (post bltd NAT)
9. idhttpsd challenges the clients authentication (method varies depending on configured authentication mechanism)
10. The challenge response from idhttpsd gets un-natted (by bltd) and sent back to the client (through snort)
11. Client responds to the authentication challenge
12. Response from client comes back through snort, gets re-natted by the bltd process and sent over to idhttpsd
13. idhttpsd passes the credentials it received (from clients response) to the adi process
14. adi tests authentication directly against the configured directory server
    1. adi gets a YES or NO
    2. Regardless of response, adi tells idhttpsd the verdict
    3. Assuming YES, adi will also tell SFDataCorrelator to create a mapping
15. SFDataCorrelator creates the mapping and updates snort with the mappings
16. SFDataCorrelator also sends this information to the FMC to propagate the mappings to other sensors
17. At the same time, idhttpsd will send the client another 307 redirect, redirecting the client to their original destination

Packet Processing: Protocol Preprocessors

Default Application Layer (L7) Preprocessors in a “Balanced Security and Connectivity” Network Analysis Policy (NAP):

Enabled	GID	Disabled	GID
DCE/RPC	133	SIP	140
DNS	131	POP	142
FTP & Telnet	125, 126	IMAP	141
HTTP	119		
Sun RPC	106		
GTP Command Channel	143		
SMTP	124		
SSH	128		
SSL	137		



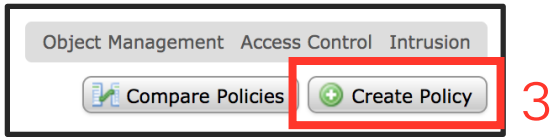
Snort Process Substeps

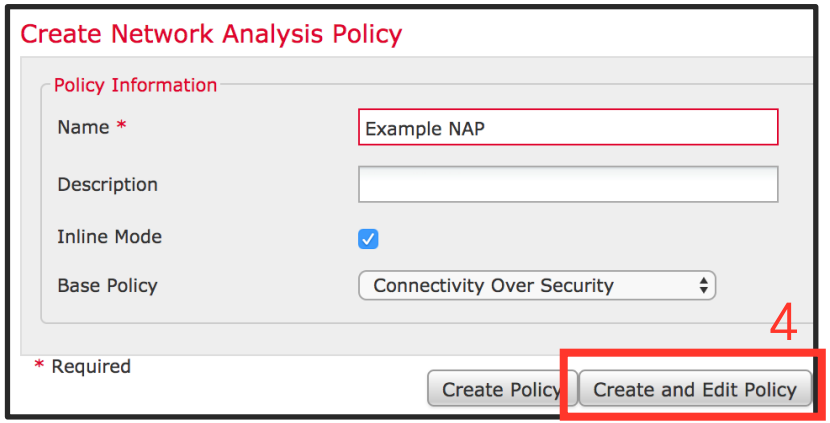
Not shown:

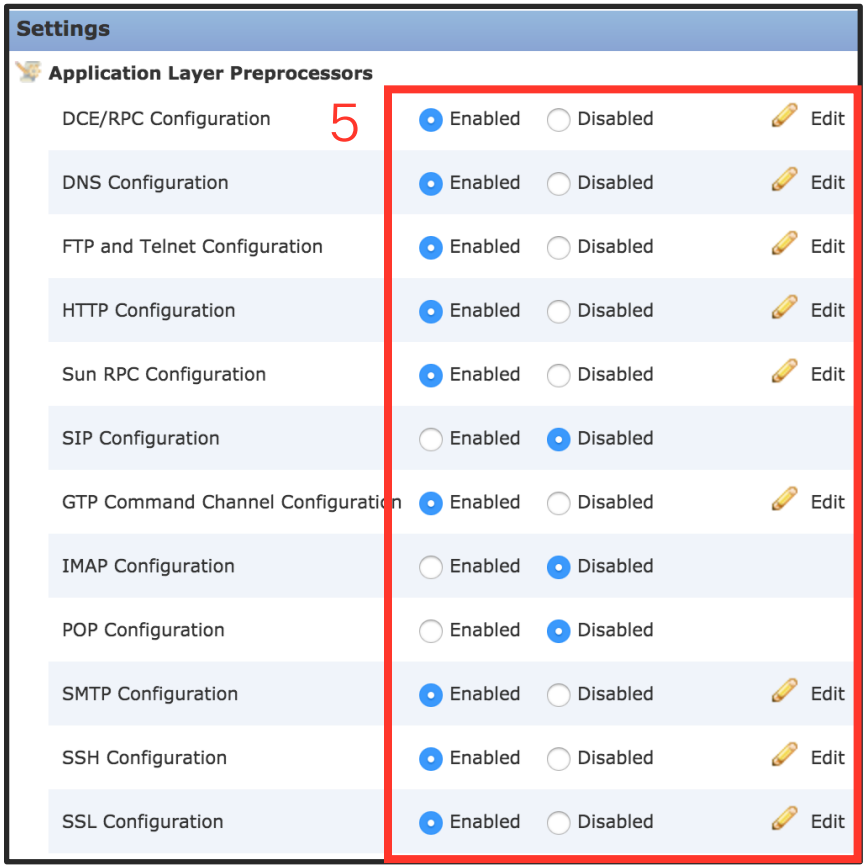
Transport and Network Layer, SCADA, Specific Threat preprocessors

Packet Processing: Build a Network Analysis Policy

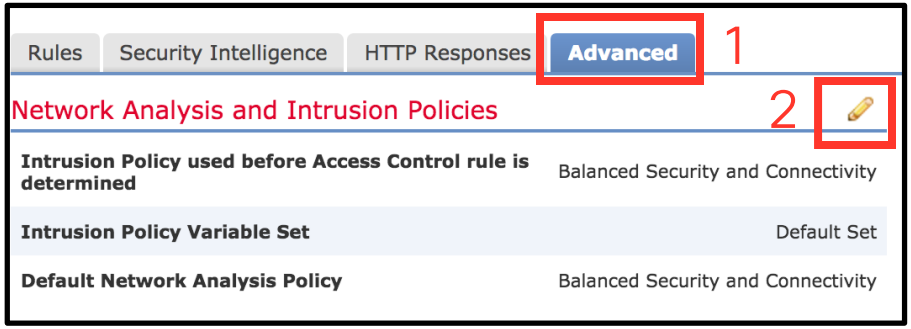
Create a Network Analysis Policy



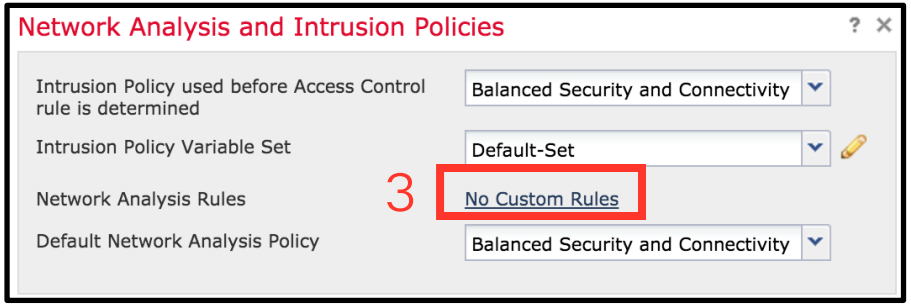


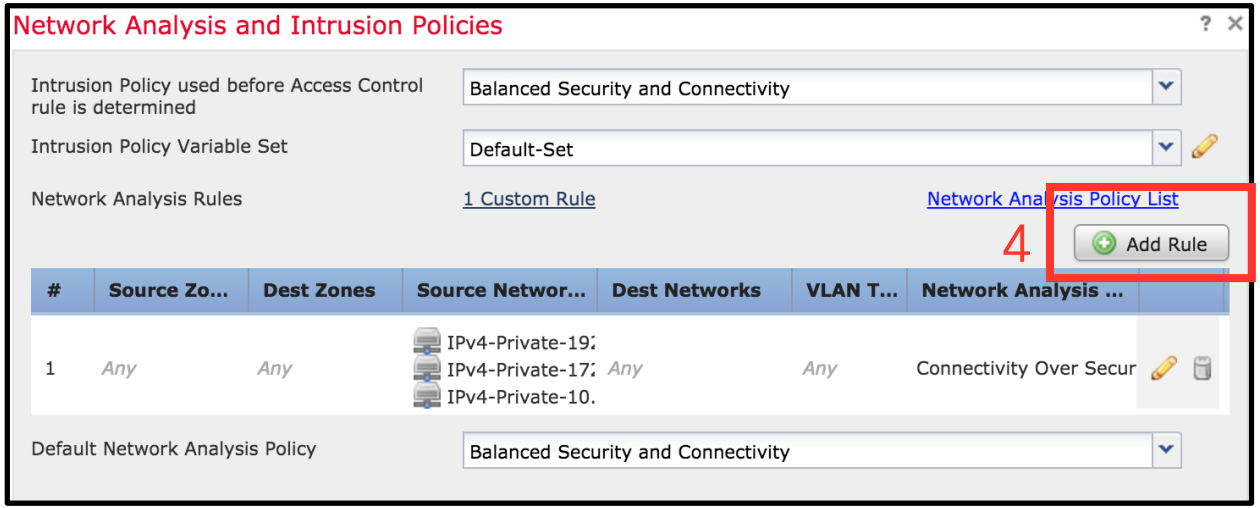


Packet Processing: Apply a Network Analysis Policy



The NAP provides preprocessor settings



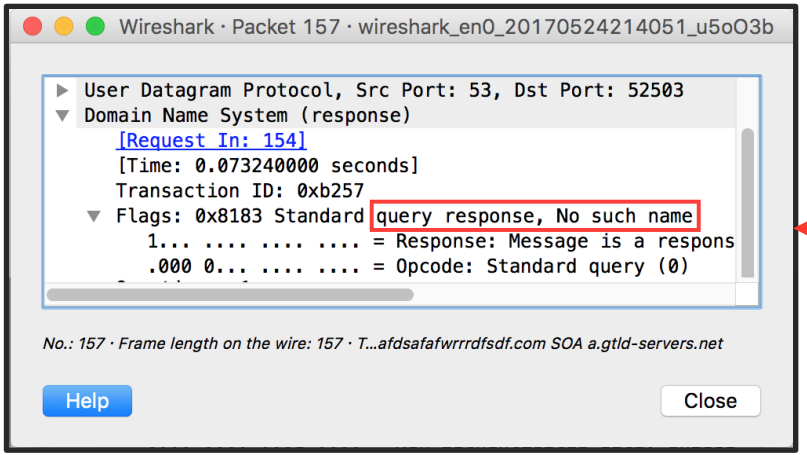


Packet Processing: DNS Security Intelligence

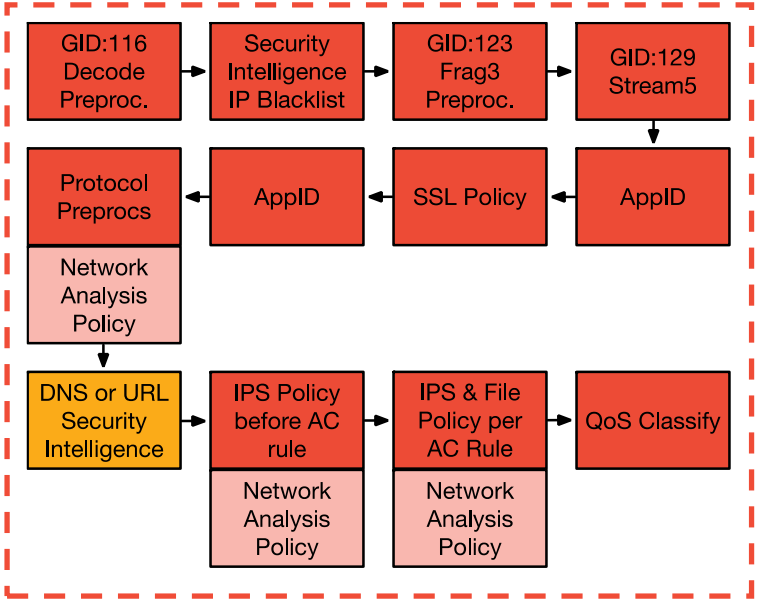
DNS SI performs a “man in the middle” of DNS queries

Option 1:

Alter DNS response to NXDOMAIN (domain not found)



NXDOMAIN Response



Snort Process Substeps

Packet Processing: DNS Security Intelligence

DNS Security Intelligence NXDomain – Firewall Engine Debug

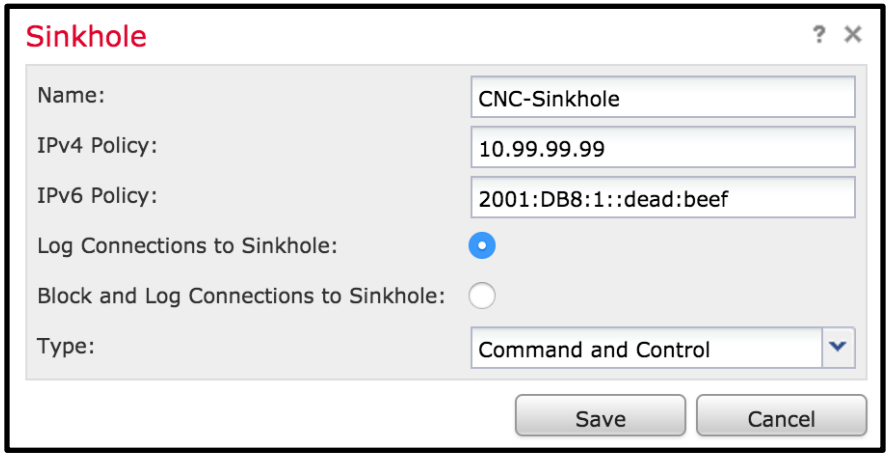
```
> system support firewall-engine-debug

[lines removed]

10.1.1.2-54821 and 172.18.108.34-53 17 AS 1 I 1 no session DNS SI shared mem lookup
returned 1 for example.com

[lines removed]
10.1.1.2-54821 and 172.18.108.34-53 17 AS 1 I 1 no session Got DNS list match. si list
1048587
10.1.1.2-54821 and 172.18.108.34-53 17 AS 1 I 1 no session Firing DNS action DNS NXDomain
10.1.1.2-54821 and 172.18.108.34-53 17 AS 1 I 1 no session DNS SI: Matched rule order 3,
Id 5, si list id 1048587, action 22, reason 2048, SI Categories 1048587,0
```

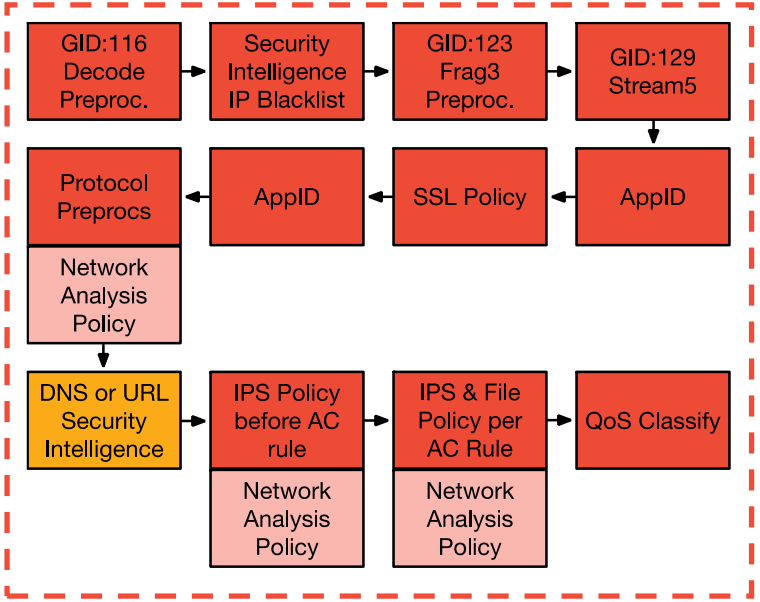
Packet Processing: DNS Security Intelligence

DNS SI performs a “man in the middle” of DNS queries

Option 2:

Alter DNS response to inject a Sinkhole server IP address





Snort Process Substeps

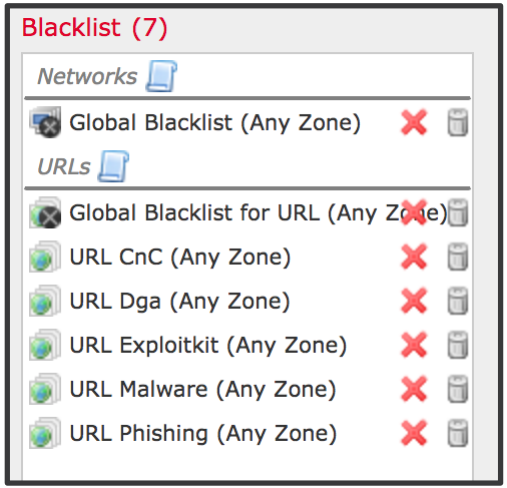
Packet Processing: DNS Security Intelligence

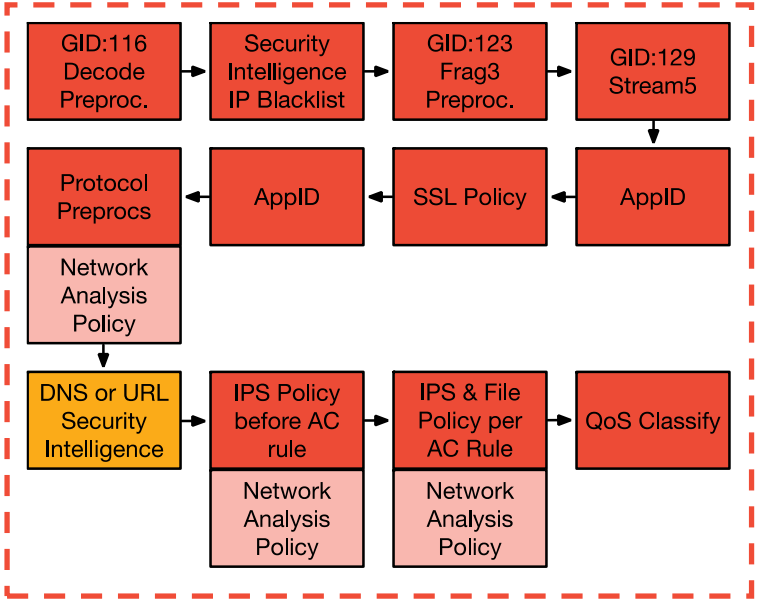
DNS Security Intelligence Sinkhole - Firewall Engine Debug

```
> system support firewall-engine-debug

[lines removed]

10.1.1.2-42818 and 172.18.108.34-53 17 AS 1 I 2 no session DNS SI shared mem lookup
returned 1 for example.com

[lines removed]
10.1.1.2-42818 and 172.18.108.34-53 17 AS 1 I 2 no session Got DNS list match. si list
1048587
10.1.1.2-42818 and 172.18.108.34-53 17 AS 1 I 2 no session Firing DNS action DNS Sinkhole
10.1.1.2-42818 and 172.18.108.34-53 17 AS 1 I 2 no session DNS SI: Matched rule order 3,
Id 5, si list id 1048587, action 23, reason 2048, SI Categories 1048587,0
```

Packet Processing: URL Security Intelligence

- URL SI is independent from Access Control URL rules
- Blocks lists of malicious domains
- Matches the HTTP GET or TLS Client Hello





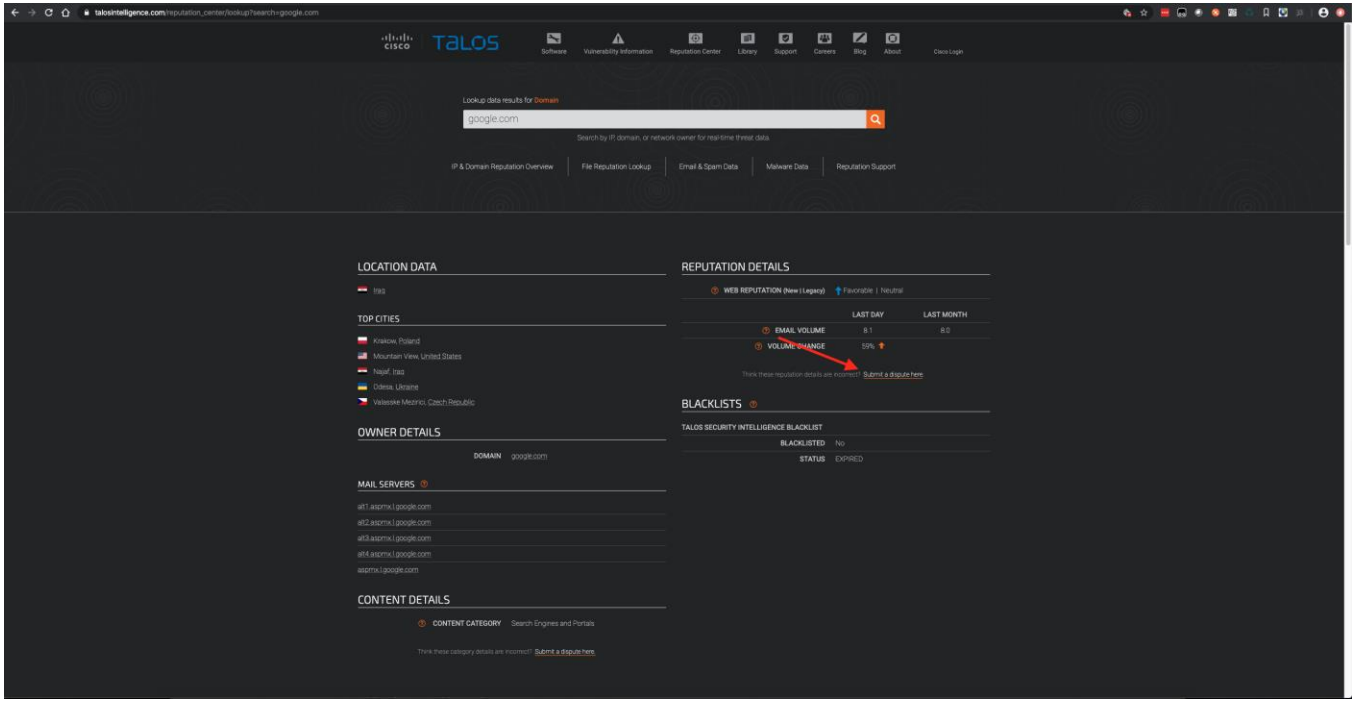
Snort Process Substeps

Packet Processing: URL Security Intelligence

URL Security Intelligence Block (Deny) - Firewall Engine Debug

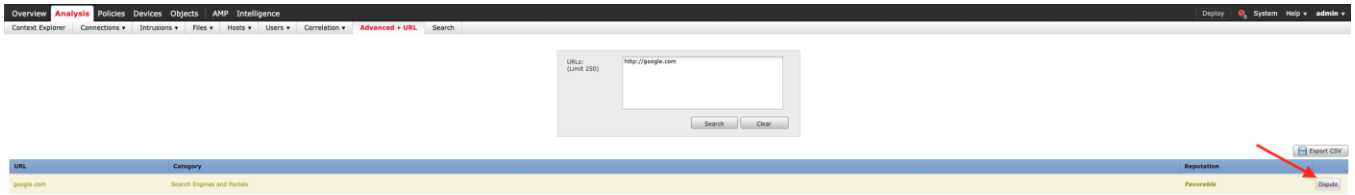
```
> system support firewall-engine-debug

[lines removed]

10.1.1.2-35316 > 10.9.9.9-80 6 AS 1 I 21 URL SI:
ShmDBLookupURL("http://example.com/") returned 1
10.1.1.2-35316 > 10.9.9.9-80 6 AS 1 I 21 matched non-allow rule order 33, id 33
10.1.1.2-35316 > 10.9.9.9-80 6 AS 1 I 21 URL SI: Matched rule order 33, Id 33,
si list id 1048584, action 4
10.1.1.2-35316 > 10.9.9.9-80 6 AS 1 I 21 deny action
```

Packet Processing: URL Security Intelligence



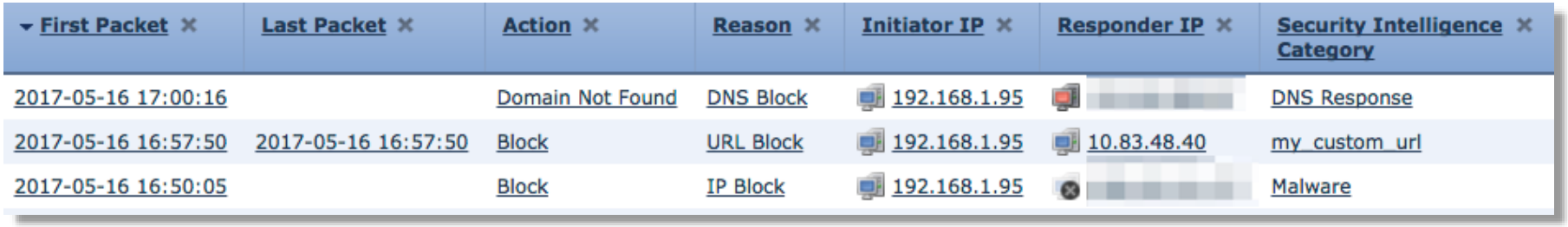
Dispute Reputations

6.5



Packet Processing: URL Security Intelligence

Analysis > Connections > Security Intelligence Events

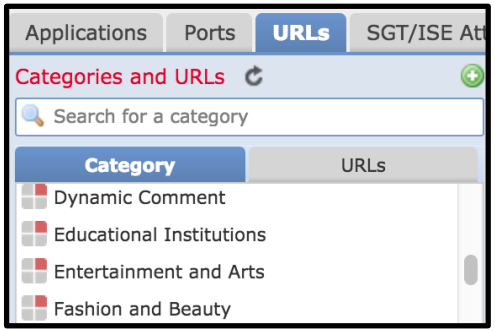


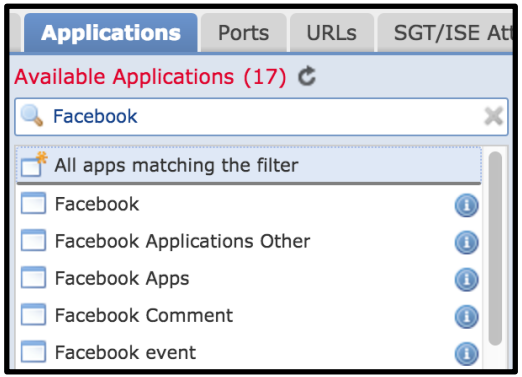
First Packet	Last Packet	Action	Reason	Initiator IP	Responder IP	Security Intelligence Category
2017-05-16 17:00:16		Domain Not Found	DNS Block	192.168.1.95		DNS Response
2017-05-16 16:57:50	2017-05-16 16:57:50	Block	URL Block	192.168.1.95	10.83.48.40	my_custom_url
2017-05-16 16:50:05		Block	IP Block	192.168.1.95		Malware

With logging enabled for all SI types you should be able to easily see what is being blocked by SI.

Packet Processing: IPS Policy before Access Rules

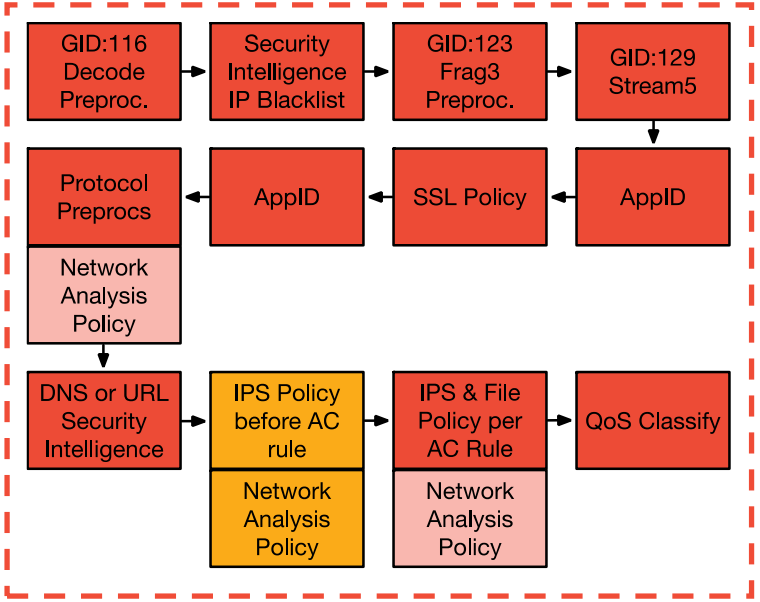
- Access Control rules can match URLs or Applications





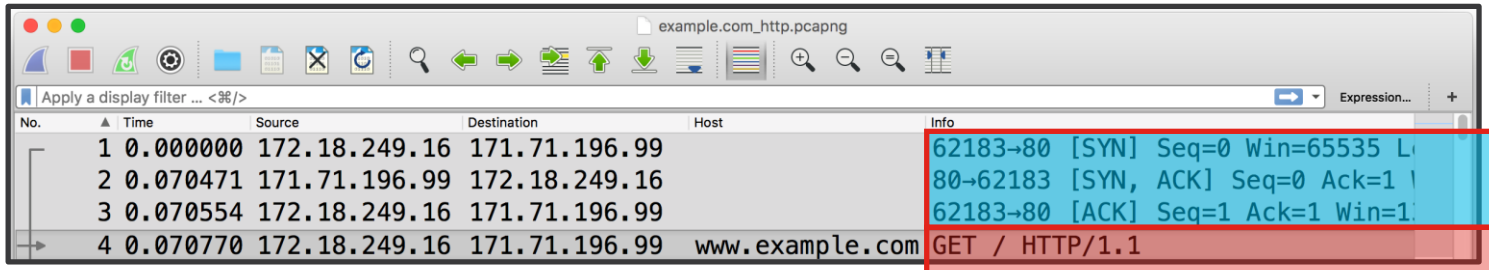
- To match a URL or App rule, Snort often needs the TLS Client Hello or HTTP GET

- Packets sent in a flow before matching an AC rule hit the "Intrusion Policy used before Access Control rule is determined"



Snort Process Substeps

Packet Processing: IPS Policy before Access Rules

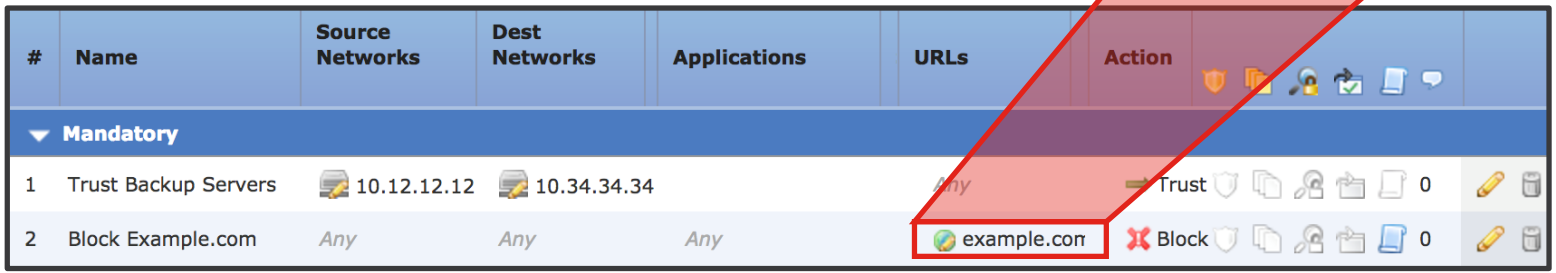


TCP 3-way Handshake

HTTP GET

Packets before HTTP GET

Packets after HTTP GET



Packet Processing: Access Control Policy Rules

Access Control Policy rules are evaluated from top to bottom

Allow – Permit unless prohibited by an IPS or File Policy

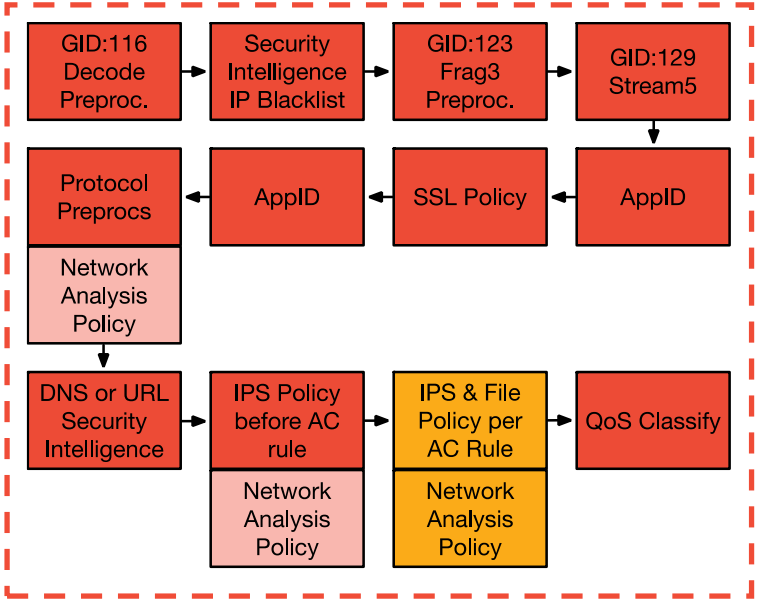
Trust – Pass the traffic without IPS or File inspection

Block – Silently drop the flow

Block with Reset – Send a TCP Reset or ICMP Unreachable

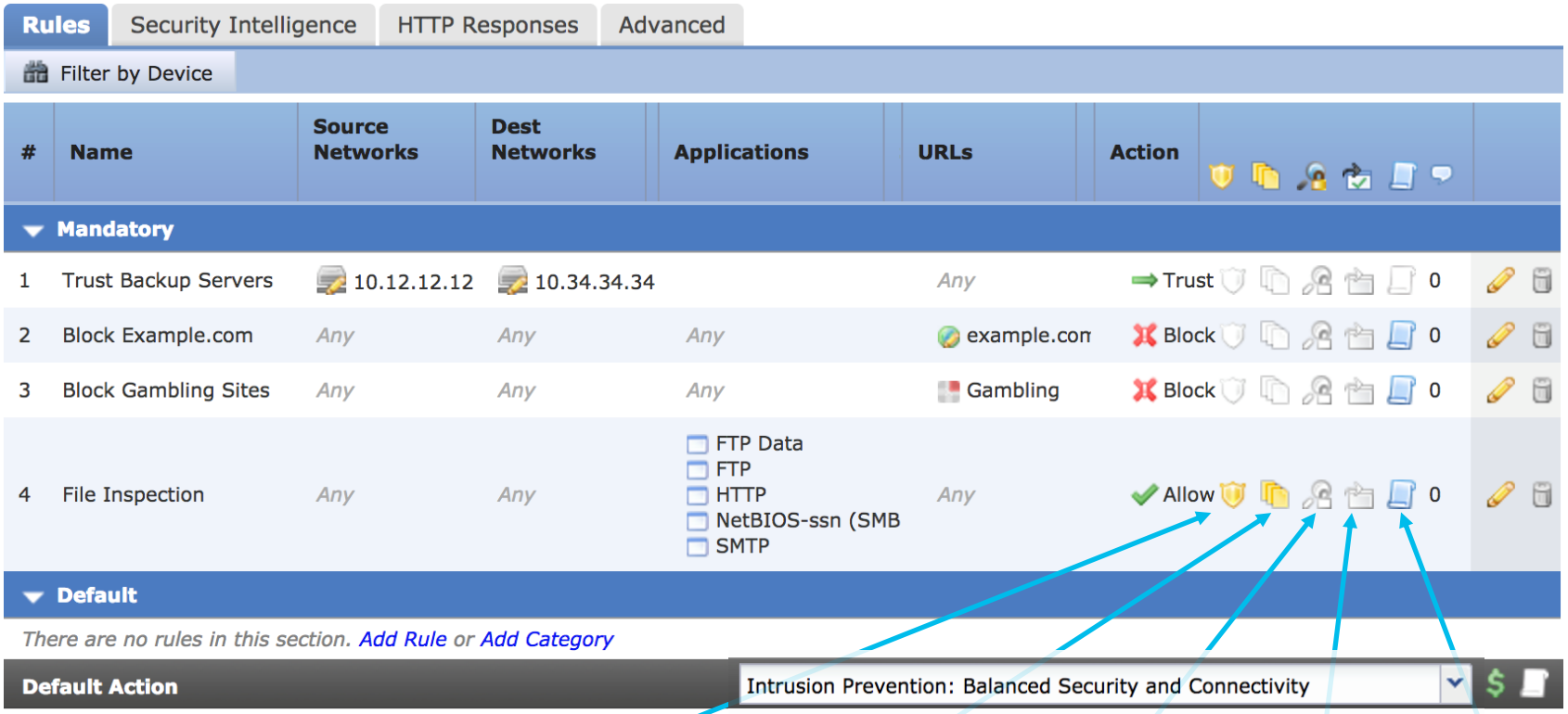
Interactive Block with Reset – Inject an HTTP 403 Forbidden

Monitor – Log the traffic and continue rule evaluation

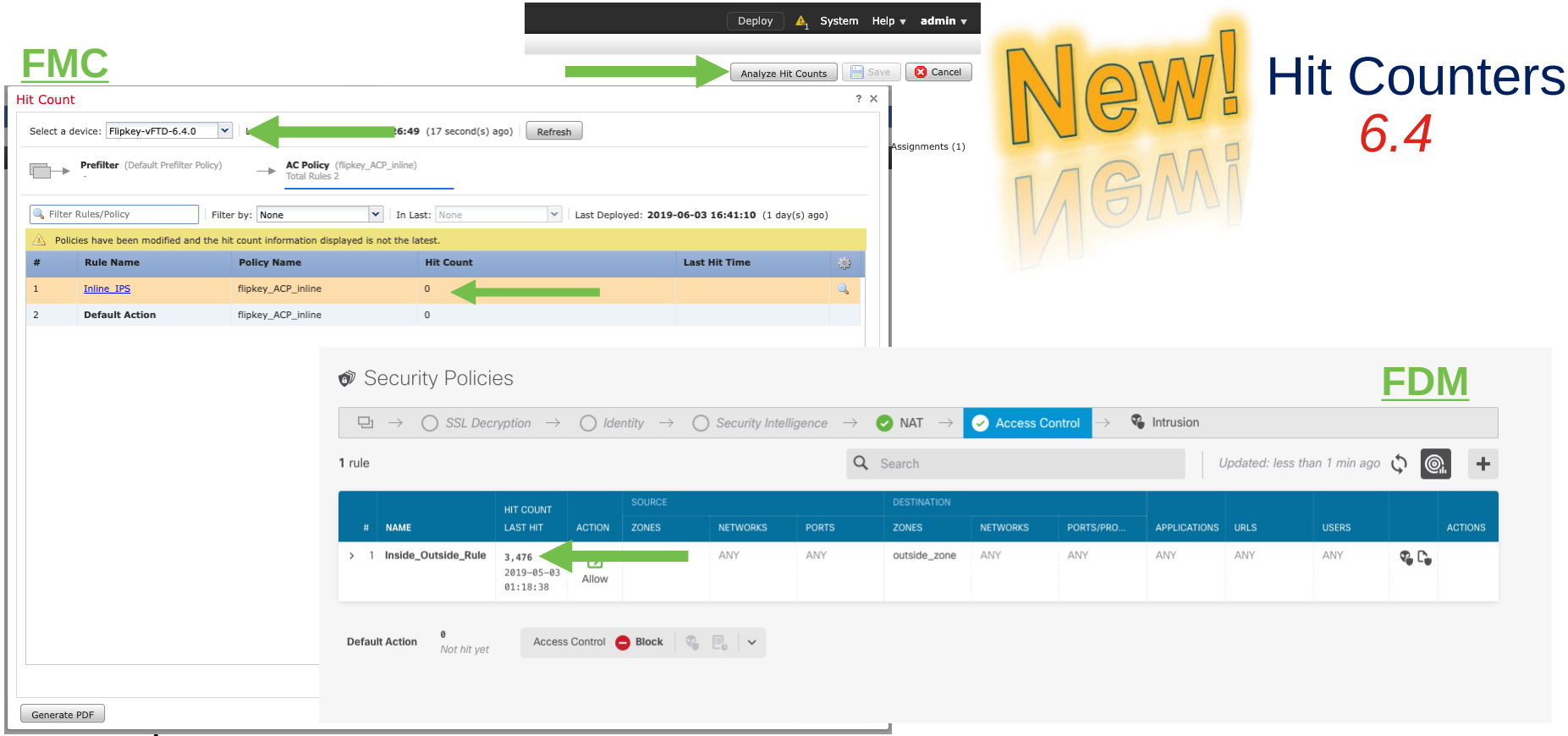


Snort Process Substeps

Packet Processing: Access Control Policy Rules



Packet Processing: Access Control Policy Rules



New!

Hit Counters

6.4

Packet Processing: Access Control Rule Evaluation

#	Name	Source Zones	Dest Zones	Source Networks	Dest Networks	VLAN ...	Users	Applic...	Sourc...	Dest P...	URLs	ISE/S... Attrib...	Acti...
Mandatory - JG AC (all) (1-6)													
1	Trust ssh for host	Any	Any	192.168.0.7	Any	Any	Any	Any	Any	SSH	Any	Any	Trust
2	inspect	Any	Any	10.0.0.0/8	Any	Any	Any	Any	Any	Any	Any	Any	Allow
3	trust server backup	Any	Any	192.168.62.3	10.123.175.22	Any	Any	Any	Any	Any	Any	Any	Trust

SSH Connection from 192.168.62.3 to 10.123.175.22

1. SYN 192.168.62.3 → 10.123.175.22
2. SYN,ACK 10.123.175.22 → 192.168.62.
3. ACK 192.168.62.3 → 10.123.175.22
4. SSH 192.168.62.3 → 10.123.175.22

Starts evaluation at 'inspect' rule

Service identified as SSH

No match 'inspect' rule (non-http)

Match 'trust server backup' rule and Trust flow

Packet Processing: Rule Evaluation

firewall-engine-debug

Example: SSH Connection from 192.168.62.3 to 10.123.175.22

SYN → SYN,ACK → ACK → First SSH Packet (client to server)

```
192.168.62.3-46594 > 10.123.175.22-22 6 AS 1 I 0 New session
192.168.62.3-46594 > 10.123.175.22-22 6 AS 1 I 0 Starting with minimum 4, 'inspect', and IPProto first
with zones 1 -> 2, geo 0 -> 0, vlan 0, inline sgt tag: untagged, ISE sgt id: 0, svc 0, payload 0, client
0, misc 0, user 9999997, icmpType 0, icmpCode 0
192.168.62.3-46594 > 10.123.175.22-22 6 AS 1 I 0 pending rule order 4, 'inspect', XFF wait for AppId
```

```
192.168.62.3-46594 > 10.123.175.22-22 6 AS 1 I 0 Starting with minimum 4, 'inspect', and IPProto first
with zones 1 -> 2, geo 0 -> 0, vlan 0, inline sgt tag: untagged, ISE sgt id: 0, svc 0, payload 0, client
0, misc 0, user 9999997, icmpType 0, icmpCode 0
192.168.62.3-46594 > 10.123.175.22-22 6 AS 1 I 0 pending rule order 4, 'inspect', XFF wait for AppId
```

```
192.168.62.3-46594 > 10.123.175.22-22 6 AS 1 I 0 Starting with minimum 4, 'inspect', and IPProto first
with zones 1 -> 2, geo 0 -> 0, vlan 0, inline sgt tag: untagged, ISE sgt id: 0, svc 0, payload 0, client
0, misc 0, user 9999997, icmpType 0, icmpCode 0
192.168.62.3-46594 > 10.123.175.22-22 6 AS 1 I 0 pending rule order 4, 'inspect', XFF wait for AppId
```

```
192.168.62.3-46594 > 10.123.175.22-22 6 AS 1 I 0 Starting with minimum 4, 'inspect', and IPProto first
with zones 1 -> 2, geo 0 -> 0, vlan 0, inline sgt tag: untagged, ISE sgt id: 0, svc 846, payload -1,
client 2000000846, misc 0, user 9999997, icmpType 0, icmpCode 0
192.168.62.3-46594 > 10.123.175.22-22 6 AS 1 I 0 no match rule order 4, 'inspect', XFF non-http
192.168.62.3-46594 > 10.123.175.22-22 6 AS 1 I 0 match rule order 5, 'trust server backup', action Trust
```

Packet Processing: Rule Evaluation

firewall-engine-debug

SSH Connection from 192.168.62.3 to 10.123.175.22 (truncated)

```
192.168.62.3-46594 > 10.123.175.22-22 6 AS 1 I 0 New session
192.168.62.3-46594 > 10.123.175.22-22 6 AS 1 I 0 Starting with minimum 4, 'inspect', and IPProto first
with zones 1 -> 2, geo 0 -> 0, vlan 0, inline sgt tag: untagged, ISE sgt id: 0, svc 0, payload 0, client
0, misc 0, user 9999997, icmpType 0, icmpCode 0
192.168.62.3-46594 > 10.123.175.22-22 6 AS 1 I 0 pending rule order 4, 'inspect', XFF wait for AppId

[…omitted for brevity]

192.168.62.3-46594 > 10.123.175.22-22 6 AS 1 I 0 Starting with minimum 4, 'inspect', and IPProto first
with zones 1 -> 2, geo 0 -> 0, vlan 0, inline sgt tag: untagged, ISE sgt id: 0, svc 846, payload -1,
client 2000000846, misc 0, user 9999997, icmpType 0, icmpCode 0
192.168.62.3-46594 > 10.123.175.22-22 6 AS 1 I 0 no match rule order 4, 'inspect', XFF non-http
192.168.62.3-46594 > 10.123.175.22-22 6 AS 1 I 0 match rule order 5, 'trust server backup', action Trust


[! How to map service/application ID to name]
> expert
$ grep "^846[^0-9]" /var/sf/appid/odp/appMapping.data
846 SSH 32 0 0 ssh
```

Packet Processing: Rule Evaluation

SSH Connection from 192.168.62.3 to 10.123.175.22

(Blocked/Ended before matching an AC rule)

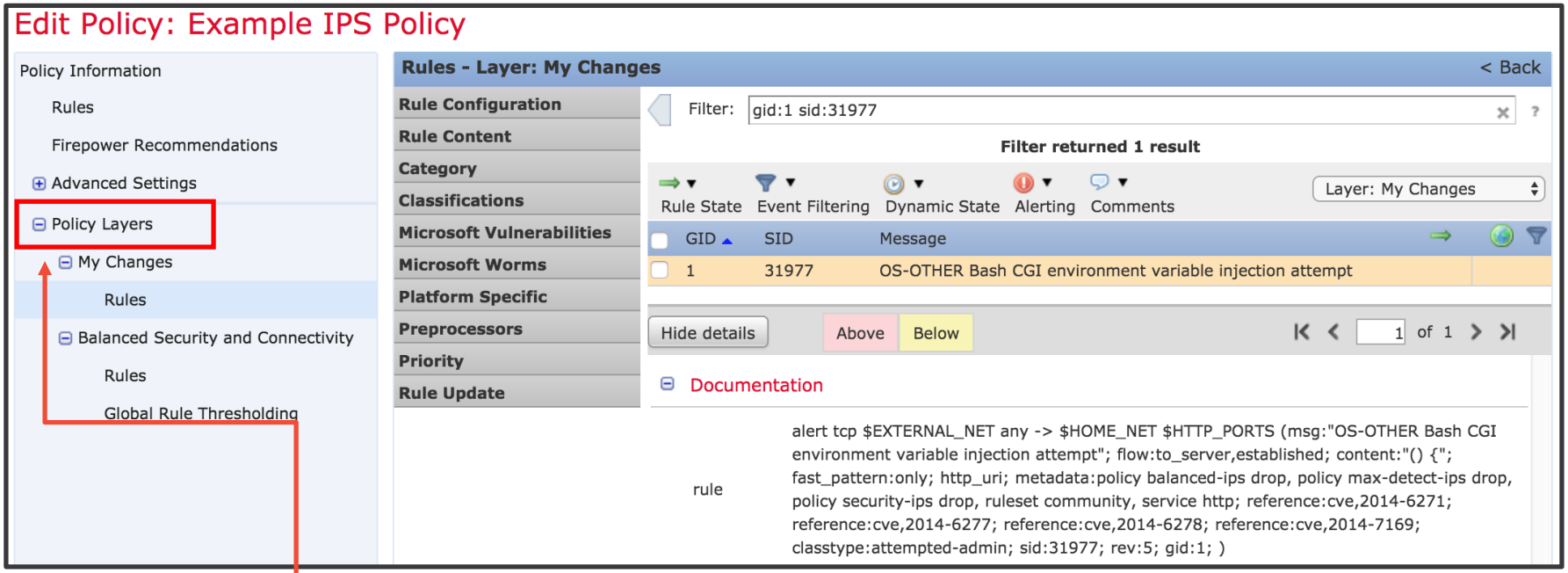
firewall-engine-debug

```
192.168.62.3-54650 > 10.123.175.22-22 6 AS 1 I 0 New session
192.168.62.3-54650 > 10.123.175.22-22 6 AS 1 I 0 Starting with minimum 4, 'inspect', and IPProto first with
zones 1 -> 2, geo 0 -> 0, vlan 0, inline sgt tag: untagged, ISE sgt id: 0, svc 0, payload 0, client 0, misc
0, user 9999997, icmpType 0, icmpCode 0
192.168.62.3-54650 > 10.123.175.22-22 6 AS 1 I 0 pending rule order 4, 'inspect', XFF wait for AppId
192.168.62.3-54650 > 10.123.175.22-22 6 AS 1 I 0 Deleting session

[!Session was deleted because we hit a drop IPS rule and blacklisted the flow.
This happened before AC rule was matched (Intrusion policy before AC rule match dropped).
Firewall engine will re-evaluate from top of AC policy to find a rule for logging decision]

192.168.62.3-54650 > 10.123.175.22-22 6 AS 1 I 0 Starting with minimum 0, id 0 and IPProto first with zones
1 -> 2, geo 0 -> 0, vlan 0, inline sgt tag: 0, ISE sgt id: 0, svc -1, payload -1, client -1, misc -1, user
9999997, icmpType 102, icmpCode 22
192.168.62.3-54650 > 10.123.175.22-22 6 AS 1 I 0 no match rule order 3, 'Trust ssh for host', src network
and GEO
192.168.62.3-54650 > 10.123.175.22-22 6 AS 1 I 0 no match rule order 4, 'inspect', XFF non-http
192.168.62.3-54650 > 10.123.175.22-22 6 AS 1 I 0 match rule order 5, 'trust server backup', action Trust
```

Action	Reason	Initiator IP	Responder IP	Source Port / ICMP Type	Destination Port / ICMP Code	Application Protocol	Client	Intrusion Events	Access Control Policy	Access Control Rule
Block	Intrusion Block	192.168.62.3	10.123.175.22	55654 / tcp	22 (ssh) / tcp				JG AC (all)	trust server backup

AC Rule has “Trust” action but connection event action shows “Block”

Packet Processing: Access Control with IPS

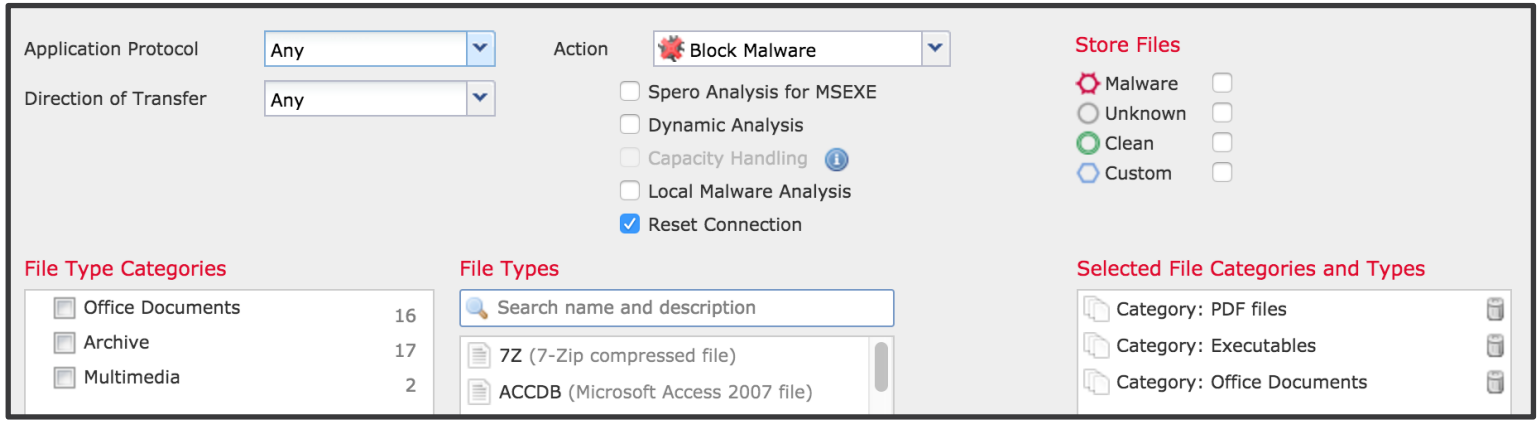


Intrusion Policies are built on layers

Prebuilt base layers from Cisco TALOS:

- Connectivity over Security (~500 rules)
- Balanced Security & Connectivity (~9,000 rules)
- Security over Connectivity (~13,000 rules)

Packet Processing: Access Control with File



- Like Intrusion Policies, a File Policy is tied to an Access Control Rule
- Checks multiple file types by looking up a SHA256 checksum for known malware
- Can submit unknown files to the AMP cloud or AMP ThreatGrid appliance

```
> system support firewall-engine-debug
10.1.1.2-16969 > 10.9.9.9-80 6 AS 0 I 1 File malware event for
275a021bbfb6489e54d471899f7db9d1663fc695ec2fe2a2c4538aabf651fd0f named eicar.com with
disposition Malware and action Block Malware
```

Packet Processing: QoS Classification in Snort

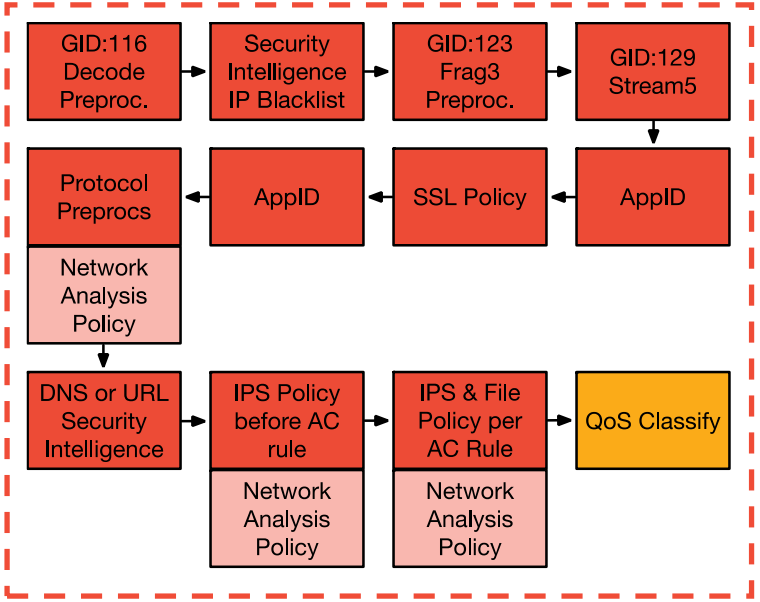
Eligible traffic for rate-liming:

- Allowed or Trusted

Ineligible traffic for rate-liming:

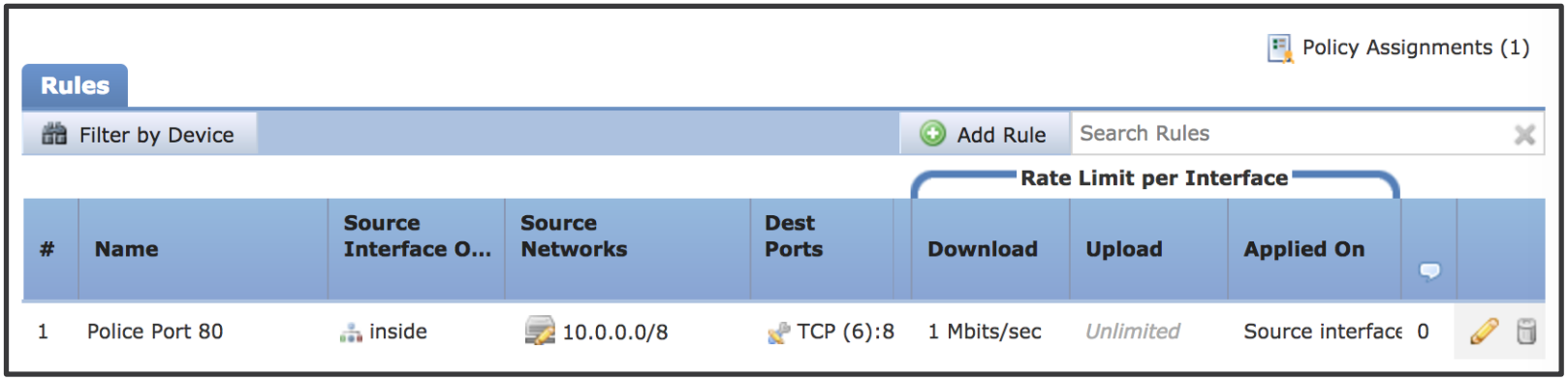
- Blocked or Prefilter Fastpath (Snort exempt)

- Snort classifies traffic by matching it to a QoS rule
- Snort tells Lina the Flow-rule QoS id for each flow
- On the Lina interface, the Rule ID matches a traffic class



Snort Process Substeps

Packet Processing: QoS Classification in Snort




```
> expert
$ cat /ngfw/var/sf/detection_engines/[UUID]/qos.rules
[lines removed]
268435467 ratelimit 2 10.0.0.0 8 any any any  80 any 6

> system support firewall-engine-debug
[lines removed]
10.1.1.2-59831 > 10.9.9.9-80 6 AS 1 I 19 match rule order 1, id 268435467 action Rate Limit
10.1.1.2-59831 > 10.9.9.9-80 6 AS 1 I 19 QoS policy match status ((null)), match action
(Rate Limit), QoS rule id (268435467)
```

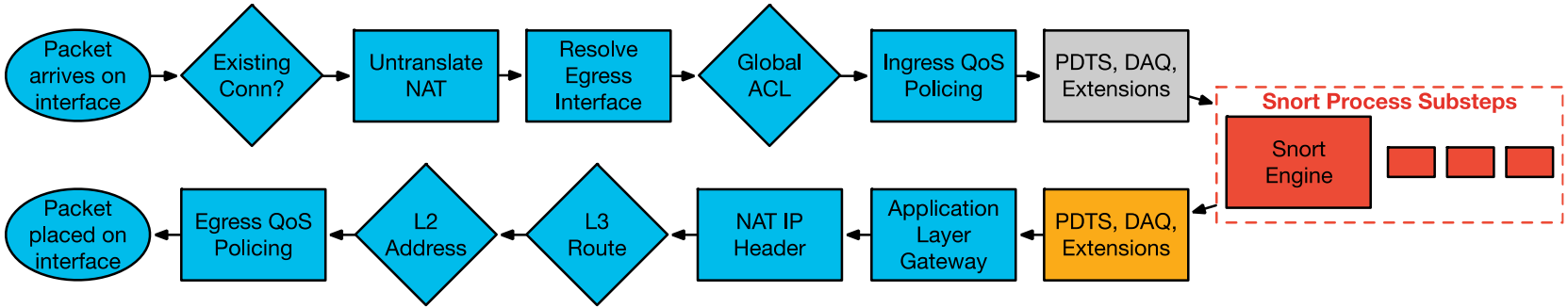
QoS Rule ID

Packet Processing: QoS Interface Policing in LINA

```
> system support diagnostic-cli

firepower# show run service-policy
service-policy global_policy global
service-policy policy_map_inside interface inside

firepower# show service-policy interface inside
Interface inside:
  Service-policy: policy_map_inside
    Flow-rule QoS id: 268435467
      Output police Interface inside:
        cir 1000000 bps, bc 31250 bytes

firepower# show conn detail
TCP outside: 10.9.9.9/80 inside: 10.1.1.2/59831,
    flags UxIO N, qos-rule-id 268435467, idle 0s, uptime 4m5s, timeout 1h0m, bytes
15542738, xlate id 0x7f05a30260c0
```

QoS Rule ID

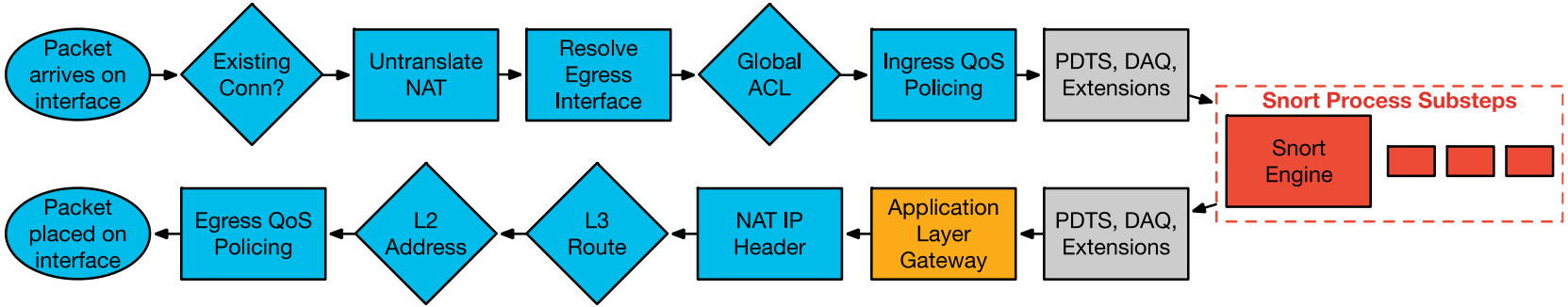
Packet Processing: Packet Data Transport System



The Packet Data Transport System sends packets back to Lina after Snort processing.

Note: It is extremely rare for any packets to be dropped at this stage.

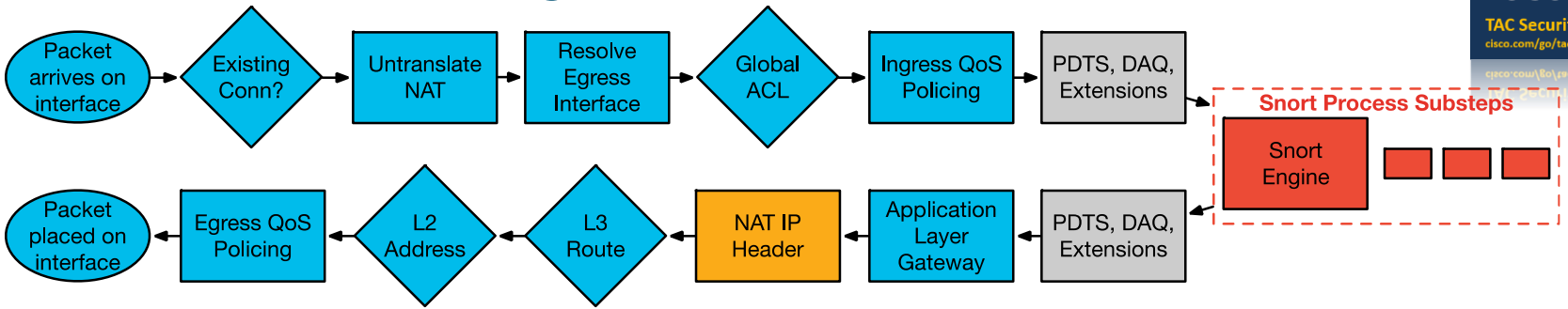
Packet Processing: Application Layer Gateway



- Stateful inspection ensures protocol compliance at TCP/UDP/ICMP level
- (Optional) Customizable application inspection up to Layer 7 (FTP, SIP, and so on)
  - Rewrite embedded IP addresses, open up ACL pinholes for secondary connections
  - Additional security checks are applied to the application payload

```
ASA-4-406002: FTP port command different address: 10.2.252.21(192.168.1.21) to
209.165.202.130 on interface inside
ASA-4-405104: H225 message received from outside_address/outside_port to
inside_address/inside_port before SETUP
```

Packet Processing: NAT IP Header



- Translate the source and destination IP addresses in the IP header
- Translate the port if performing PAT
- Update header checksums

Auto NAT (Object NAT)

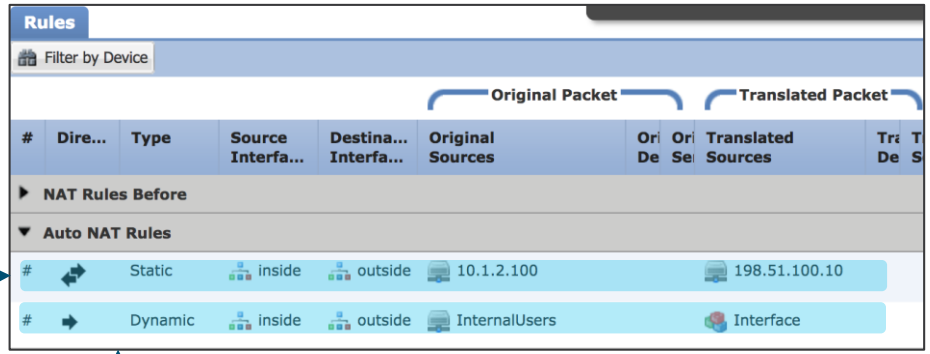
- Auto NAT is the simplest form of NAT, and is defined *within an object*

Static Host NAT

```
object network obj-WebServer
 host 10.1.2.100
 nat (inside,outside) static 198.51.100.10
```

Dynamic PAT (interface overload)

```
object network InternalUsers
 subnet 192.168.2.0 255.255.255.0
 nat (inside,outside) dynamic interface
```



Manual NAT (Twice NAT)

- Manual NAT can specify the source and the destination translations

Network Objects

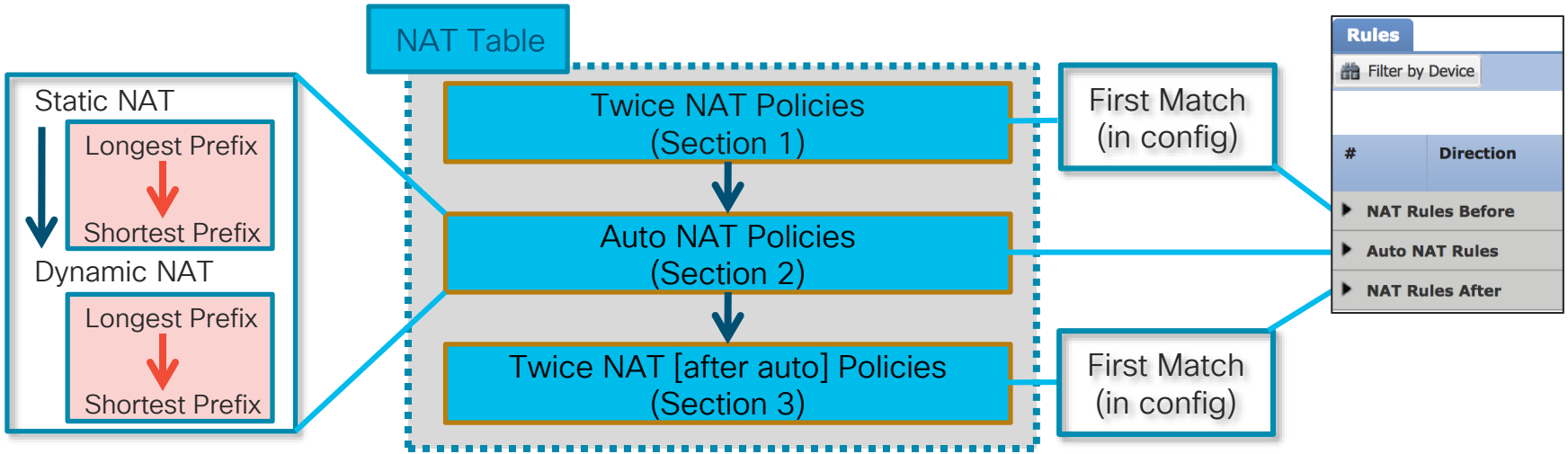
```
object network 10.10.10.0-net
 subnet 10.10.10.0 255.255.255.0
!
object network 192.168.1.0-net
 subnet 192.168.1.0 255.255.255.0
```

Twice NAT Config

```
nat (inside,outside) source static 10.10.10.0-net 10.10.10.0-net
destination static 192.168.1.0-net 192.168.1.0-net
```

NAT Order of Operation

- FTD configuration is built into the **NAT table**
- The NAT Table is based on **First Match** (top to bottom)
- The show nat command will display the NAT table in order

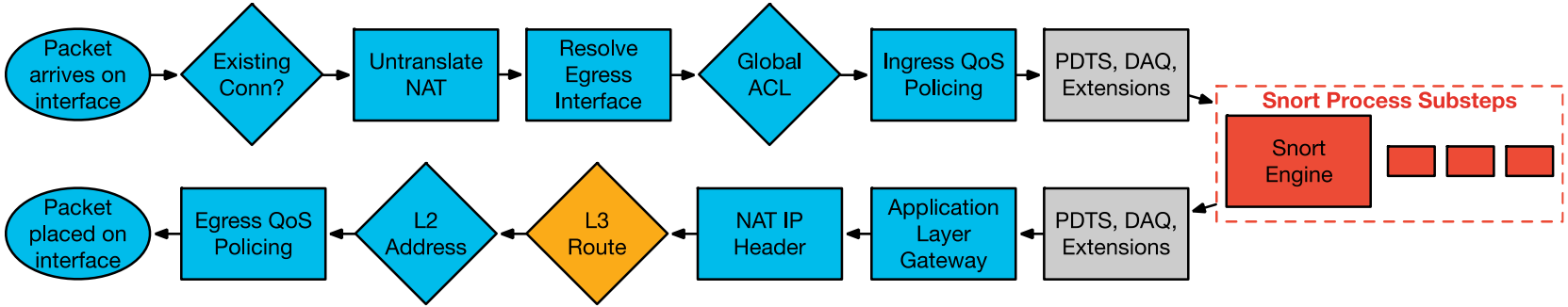


Carrier Grade NAT (CGNAT)

- In 2019, version 6.5 added CGNAT functionality to FTD
- CGNAT allocates xlates to source IPs in blocks instead of individually
- Goal is to improve performance and reduce logging overhead
- Troubleshooting methodology remains the same

```
> show local-host 10.0.0.1
.....
Port Block Allocation:
    Block 1: IP 192.0.2.107, UDP port range 53248-54271, in use 934
  Xlate:
    UDP PAT from inside:10.0.0.1/934 to outside:192.0.2.107/53736 flags ri idle 0:00:00
timeout 0:05:00
    UDP PAT from inside:10.0.0.1/933 to outside:192.0.2.107/53625 flags ri idle 0:00:00
timeout 0:05:00
```

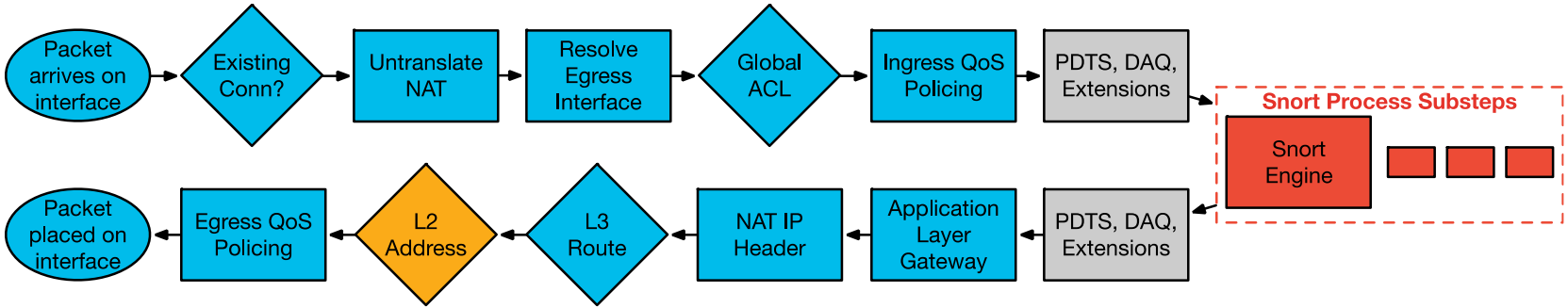
Packet Processing: L3 Route Lookup



- After the IP header translation an interface route lookup is performed
- Only routes pointing out the egress interface are eligible
- Remember: NAT rule can forward the packet to the egress interface, even though the routing table may point to a different interface
  - If the destination is not routable out of the identified egress interface, the packet is dropped

```
%ASA-6-110003: Routing failed to locate next hop for TCP from inside:192.168.103.220/59138
to dmz:172.15.124.76/23
```

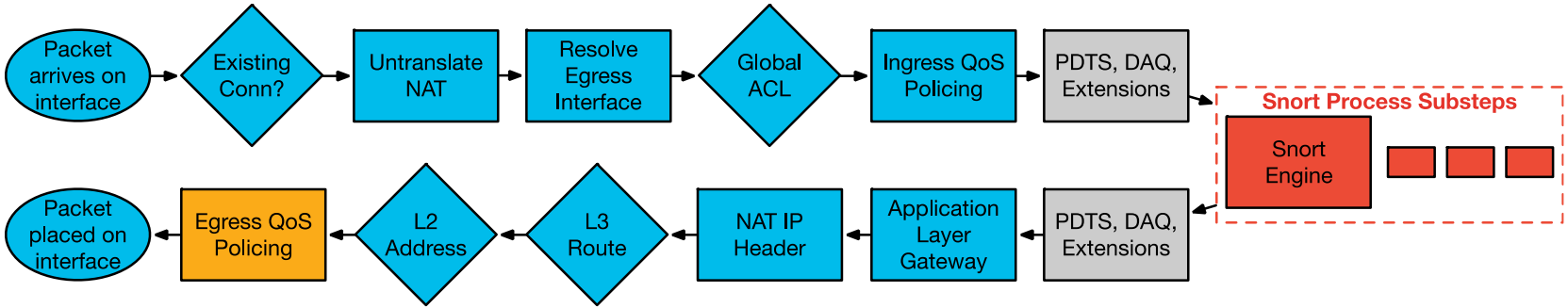
Packet Processing: L2 Address Lookup



- Once a Layer 3 route has been found, and next hop IP address identified, Layer 2 resolution is performed
  - Layer 2 rewrite of MAC header
- If Layer 2 resolution fails – no syslog
  - show arp will not display an entry for the L3 next hop
  - debug arp will indicate if we are not receiving an ARP reply

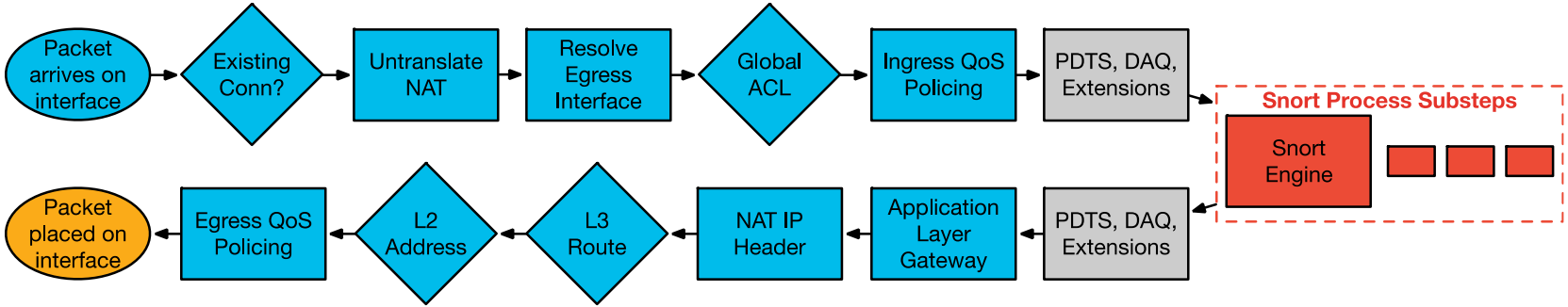
```
arp-req: generating request for 10.1.2.33 at interface outside
arp-req: request for 10.1.2.33 still  pending
```

Packet Processing: Egress QoS Policing




```
> system support diagnostic-cli

firepower# show service-policy interface outside
Interface outside:
  Service-policy: policy_map_outside
    Flow-rule QoS id: 268435467
      Output police Interface outside:
        cir 1000000 bps, bc 31250 bytes
```

Packet Processing: Transmit Packet



- Packet is transmitted on wire
- Interface counters will increment on interface
- Underrun counter indicates drops due to egress interface oversubscription
  - TX ring is full

```
> show interface outside
Interface GigabitEthernet0/1 "outside", is up, line protocol is up
        …
        273399 packets output, 115316725 bytes, 80 underruns
        …
        input queue (blocks free curr/low): hardware (485/441)
        output queue (blocks free curr/low): hardware (463/0)
```

Packet Processing: Other FTD Modes

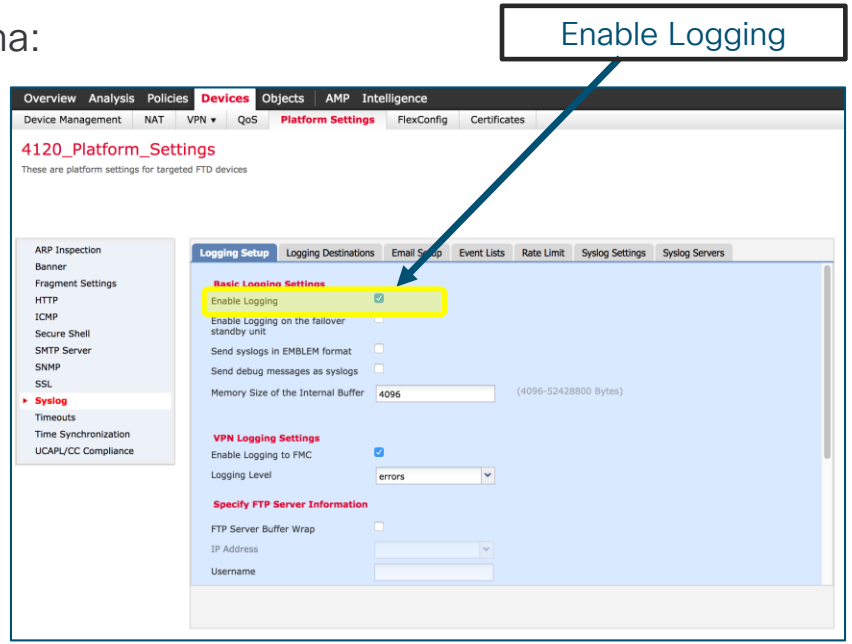
- Transparent Mode
  - Functions as an L2 bridge, re-writes VLAN tags in trunk mode
  - Traffic is processed by Lina and Snort
- Inline Sets
  - Functions as an L1 “bump in the wire”, no L2/L3 packet re-writing
  - Snort processing only (Lina sees the packet but only redirects to Snort)
  - Can be used in conjunction with both transparent and routed mode
- Flow Offload
  - Enabled by the Prefilter Fastpath option on 4100/9300 platforms*
  - Bypasses Lina and Snort completely
  - L2/L3 re-writing is handled by special network adapter in the security engine blade
  - View offloaded flows via the ‘show flow-offload flow detail’ command in Lina CLI

Troubleshooting Tools

Tools – Syslogs

- Syslogs remain the primary mechanism for recording connections to and through the firewall
- Should be the first troubleshooting tool to use for most issues
- Most syslogs in FTD are still generated from Lina:
  - Health of Lina resources and processes
  - Lina CPU, memory, block depletion
  - Failover events
  - NAT translation builds/teardowns

Note: Lina syslog config is defined under 'Platform Settings' in FMC

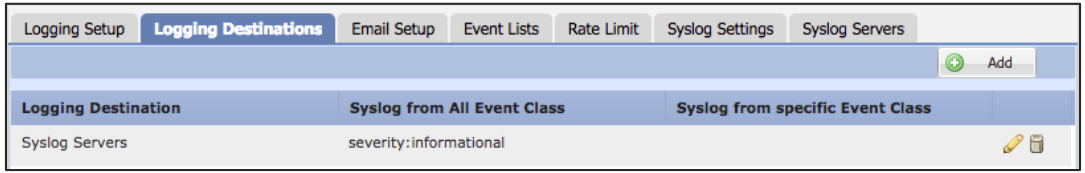


Tools – Syslogs – FMC vs. CLI configuration

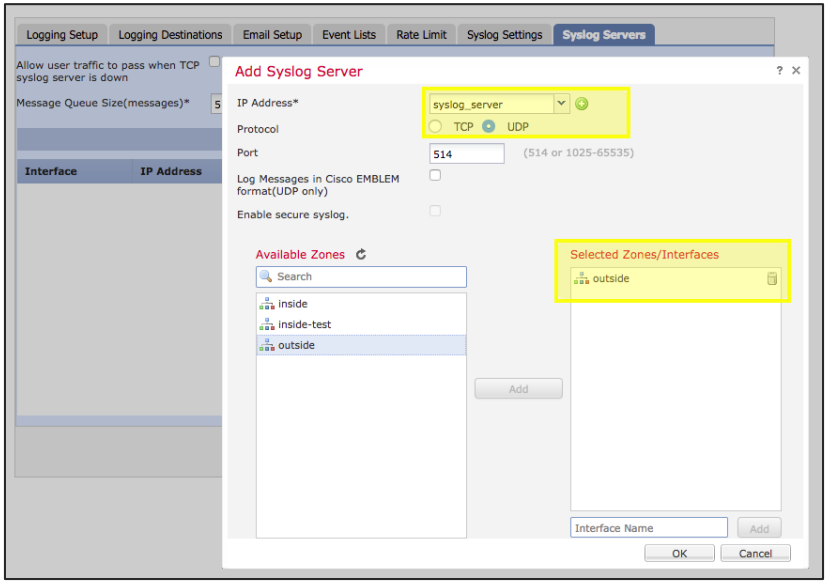
- FMC screenshots and corresponding Lina CLI configuration:

1

2



3




```
firepower# show run logging
1 logging enable
2 logging trap informational
3 logging host outside 10.1.0.1
```

Note: The syslog_server object is defined as 10.1.0.1

Tools – Syslogs – Connection Logging

- Lina connection logging and packet deny logs are disabled by default in FTD

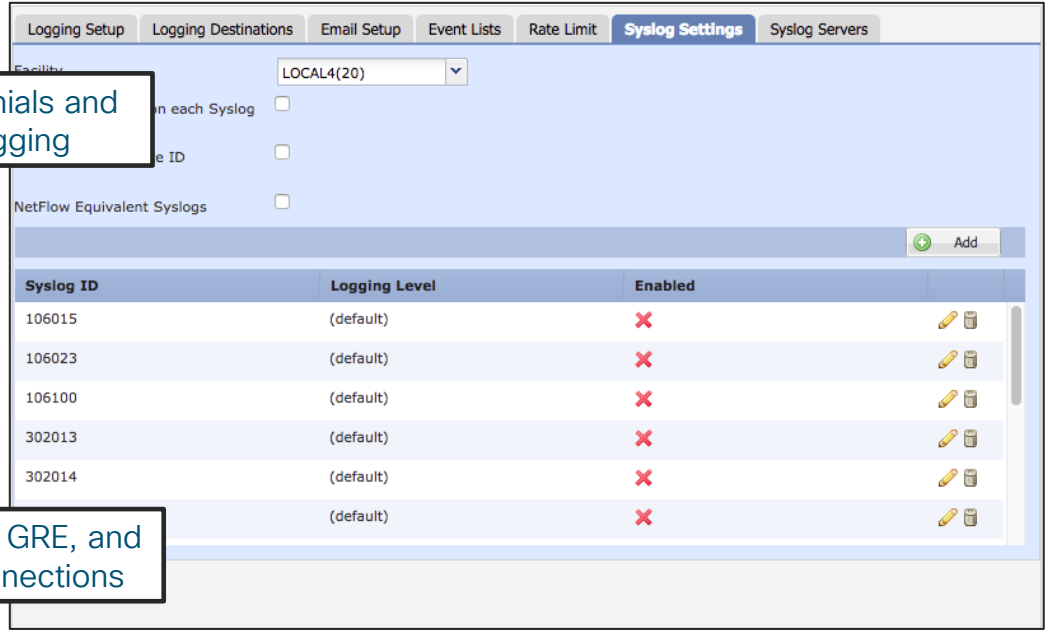
CLI:

```
firepower# show run logging
…
no logging message 106015
no logging message 313001
no logging message 313008
no logging message 106023
no logging message 710003
no logging message 106100
no logging message 302015
no logging message 302014
no logging message 302013
no logging message 302019
no logging message 302017
no logging message 302016
no logging message 302021
no logging message 302020
```

Packet denials and ACL logging

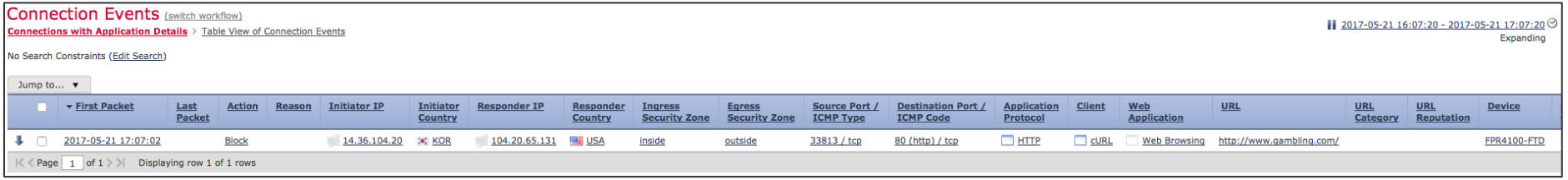
UDP, TCP, GRE, and ICMP connections

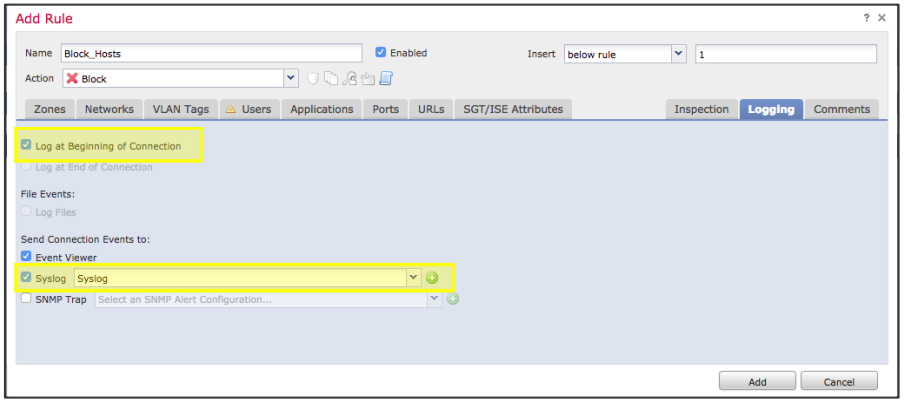
FMC:



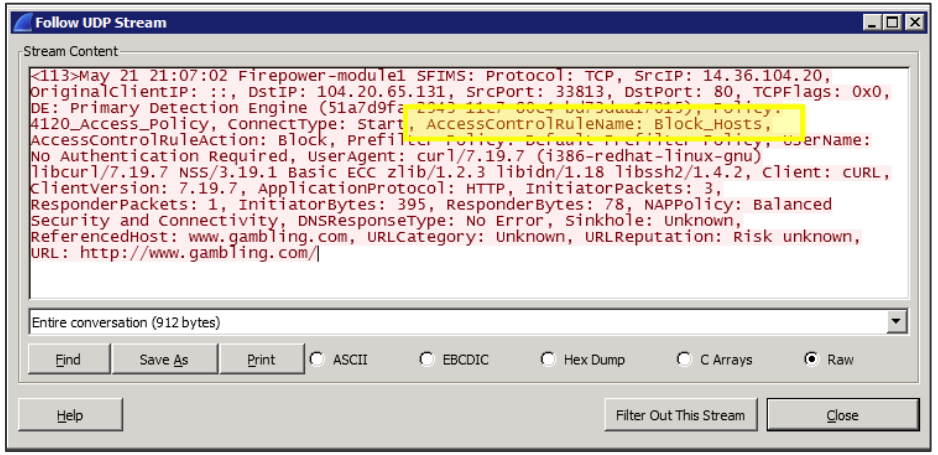
Tools – Syslogs – Connection Log Example

- Snort can also generate syslog messages for connection events when configured in the Access Policy.





Syslog generated (Wireshark view)



Tools – Syslogs – Snort vs. Lina

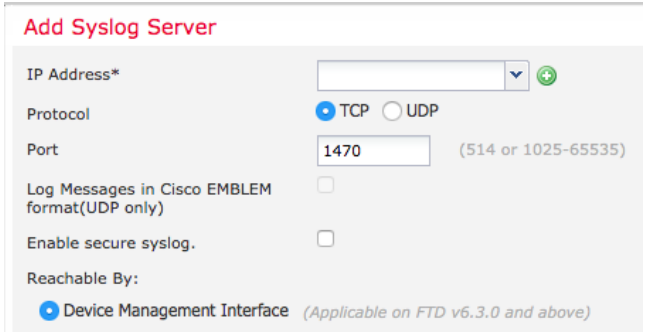
- Example: Logging at beginning AND end of connection AND syslog options for AC rule with Lina connection logging messages enabled in Syslog settings.

Date	Time	Priority	Hostname	Message
5/24/17	17:30:24	System4.Alert	10.1.1.79	May 24 21:30:22 FPR4100 SFIMS: Protocol: TCP, SrcIP: 10.1.1.20, OriginalClientIP: ::, DstIP: 172.18.124.145, SrcPort: 50072, DstPort: 21, TCPFlags: 0x0, DE: Primary Detection Engine (51a7d9fa-2943-11e7-80c4-bd73daa17015), Policy: 4120_Access_Policy, ConnectType: End, AccessControlRuleName: Allow_Hosts, AccessControlRuleAction: Allow, UserName: No Authentication Required, Client: FTP client, ApplicationProtocol: FTP, InitiatorPackets: 6, ResponderPackets: 6, InitiatorBytes: 434, ResponderBytes: 462, DNSResponseType: No Error, Sinkhole: Unknown, URLCategory: Unknown, URLReputation: Risk unknown
5/24/17	17:30:24	System4.Alert	10.1.1.79	May 24 21:30:17 FPR4100 SFIMS: Protocol: TCP, SrcIP: 10.1.1.20, OriginalClientIP: ::, DstIP: 172.18.124.145, SrcPort: 50072, DstPort: 21, TCPFlags: 0x0, DE: Primary Detection Engine (51a7d9fa-2943-11e7-80c4-bd73daa17015), Policy: 4120_Access_Policy, ConnectType: Start, AccessControlRuleName: Allow_Hosts, AccessControlRuleAction: Allow, UserName: No Authentication Required, InitiatorPackets: 2, ResponderPackets: 1, InitiatorBytes: 148, ResponderBytes: 78, DNSResponseType: No Error, Sinkhole: Unknown, URLCategory: Unknown, URLReputation: Risk unknown
5/24/17	17:30:24	Local4.Info	10.1.1.80	%ASA-6-302014: Teardown TCP connection 14704 for inside:10.1.1.20/50072 to outside:172.18.124.145/21 duration 0:00:05 bytes 40 Flow closed by inspection
5/24/17	17:30:18	Local4.Info	10.1.1.80	%ASA-6-302013: Built inbound TCP connection 14704 for inside:10.1.1.20/50072 (10.2.104.80/50072) to outside:172.18.124.145/21 (172.18.124.145/21)

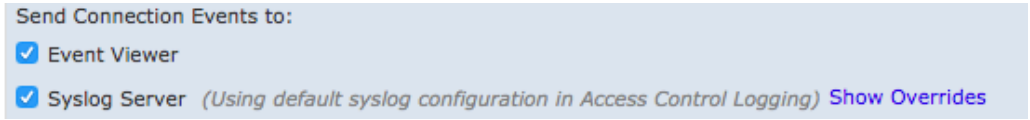
Build Snort Policy Snort Action Teardown

Tools – FTD Unified Syslogging

- In FTD 6.3 and later, syslogs can be generated from a single IP address (FTD management interface)



- %ASA- prefix changed to %FTD- and is also prepended to Snort logs
- Logging configuration in Platform Settings can be propagated to Access Control Policy



https://www.cisco.com/c/en/us/td/docs/security/firepower/630/relnotes/firepower-release-notes-630/new_features.html

Custom Syslog Levels

- Assign any syslog message to any available level
- Problem:

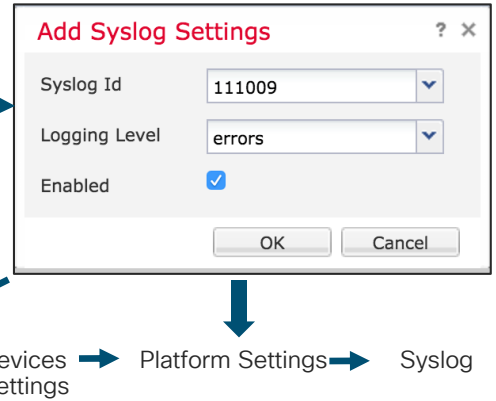
  You want to record what exec commands are being executed on the firewall; syslog ID 111009 records this information, but by default it is at level 7 (debug)

```
ASA-7-111009: User 'johndoe' executed cmd: show run
```

  The problem is we don't want to log all 1775 other syslogs that are generated at debug level

```
ASA-3-111009: User 'johndoe' executed cmd: show run
```

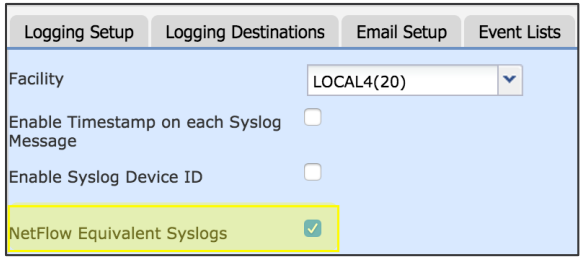
Levels	
0—Emergency	4—Warning
1—Alert	5—Notifications
2—Critical	6—Informational
3—Errors	7—Debugging



NetFlow Secure Event Logging (NSEL)

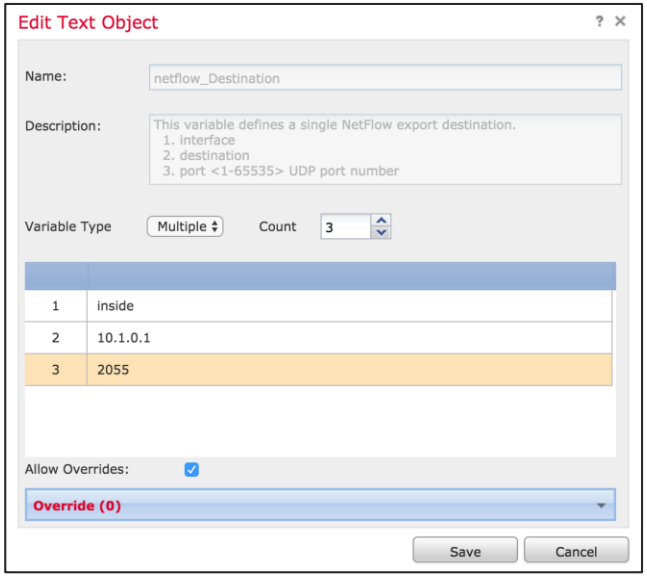
- NetFlow v9 support is available in FTD 6.2+ via FlexConfig
  - Provides a method to deliver binary logs at high speeds
  - Reduce processing overhead in printing logs
  - Combine multiple events into one NetFlow record
- FlowSets Supported:
  - Flow Creation
  - Flow Teardown
  - Flow Denied
  - Flow Update
- Remove redundant syslog messages:

Enabled by default on FTD but destinations must be configured via FlexConfig



NSEL – Netflow Destination Configuration Example

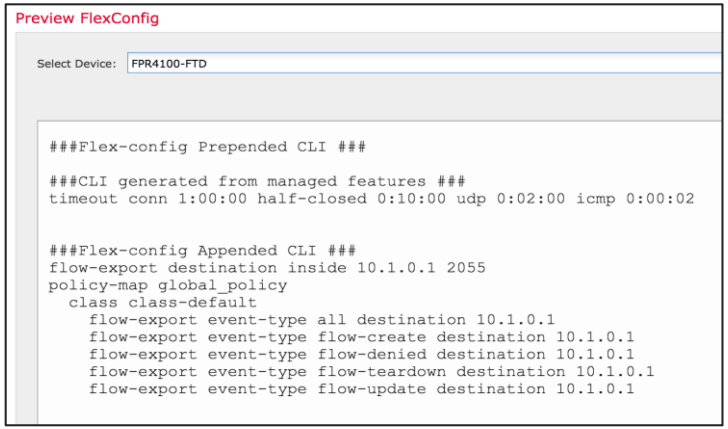
1) Define destination



Objects → Object Management → FlexConfig → Text Object

2) Add FlexConfig object to a FlexConfig policy

3) Save and preview config before deployment



Devices → FlexConfig → Edit (Pencil icon)

Logging – Common Issues

- SNMP Trap as a logging destination should only be used when you really have an SNMP server that you want to receive all syslogs
- Logging to the console should only be enabled while actively troubleshooting on the console
- Logging on the standby unit should only be used if you want to receive double the syslogs
- Allow user traffic to pass when TCP syslog server is down should nearly always be enabled with TCP syslogging

Debug Commands

- Debugs should not be the first choice to troubleshoot a problem
- Debugs can negatively impact the CPU complex and affect performance
- Most debugs are not conditional
- Know how much traffic of the matching type is passing through the firewall before enabling the respective debug

Show Output Filters

- Filters limit the output of **show** commands to only what you want to see
- Applies only to the '**system support diagnostic-cli**' interface
- Use the pipe character "**|**" at the end of **show <command>** followed by
  - **begin** Start displaying the output beginning at the first match of the RegEx, and continue to display the remaining output
  - **include** Display any line that matches the RegEx
  - **exclude** Display any line that does not match the RegEx
  - **redirect** Send output to a file (flash, tftp, ftp…)
  - **append** Append output to an existing file (flash, tftp, ftp…)

```
show <cmd> | begin|include|exclude|grep|redirect|append [-v] <regular_exp>
```

Traffic Rates

```
firepower# show traffic
[…]
TenGigabitEthernet5/1:
        received (in 2502.440 secs):
                99047659 packets        130449274327 bytes
                39580 pkts/sec  52128831 bytes/sec
        transmitted (in 2502.440 secs):
                51704620 packets        3581723093 bytes
                20661 pkts/sec  1431292 bytes/sec
      1 minute input rate 144028 pkts/sec,  25190735 bytes/sec
      1 minute output rate 74753 pkts/sec,  5145896 bytes/sec
      1 minute drop rate, 0 pkts/sec
      5 minute input rate 131339 pkts/sec, 115953675 bytes/sec
      5 minute output rate 68276 pkts/sec,  4748861 bytes/sec
      5 minute drop rate, 0 pkts/sec
```

Uptime statistics is useful to determine historical average packet size and rates:
52128831 B/sec / **39580 pkts/sec** = ~**1317 B/packet**

One-minute average is useful to detect bursts and small packets:
25190735 B/sec / **144028 pkts/sec** = ~**174 B/packet**

Xlate Table

- show xlate displays information about NAT translations through FTD
  - Second biggest memory consumer in Lina after conn table, no hardcoded size limit
- You can limit the output to just the local or global IP

```
firepower# show xlate local 10.2.1.2
5014 in use, 5772 most used
TCP PAT from inside:192.168.103.220/57762 to outside:10.2.1.2/43756 flags ri
idle 0:00:00 timeout 0:00:30
TCP PAT from inside:192.168.103.220/57761 to outside:10.2.1.2/54464 flags ri
idle 0:00:00 timeout 0:00:30
```

```
firepower# show nat pool
TCP PAT pool outside, address 10.2.1.2, range 1-511, allocated 1
TCP PAT pool outside, address 10.2.1.2, range 512-1023, allocated 0
TCP PAT pool outside, address 10.2.1.2, range 1024-65535, allocated 64102
```

Detailed NAT Information

TAC Tip

- **show nat** displays information about the NAT table
  - **detail** keyword will display object definitions
  - Watch the hit counts for policies that are not matching traffic

```
firepower# show nat detail
Manual NAT Policies (Section 1)
1 (inside) to (outside) source static science-obj science-obj destination static vpn-obj vpn-obj
    translate_hits = 0, untranslate_hits = 0
    Source - Origin: 192.168.0.0/16, Translated: 192.168.0.0/16
    Destination - Origin: 172.16.1.0/24, Translated: 172.16.1.0/24

Auto NAT Policies (Section 2)
1 (dmz) to (outside) source static webserver-obj 14.36.103.83
    translate_hits = 0, untranslate_hits = 3232
    Source - Origin: 192.168.22.32/32, Translated: 14.36.103.83/32
2 (inside) to (outside) source dynamic science-obj interface
    translate_hits = 37723, untranslate_hits = 0
    Source - Origin: 192.168.0.0/16, Translated: 14.36.103.96/16
```

Check specific translation policies in the applied order.

Translate hits indicate connections from real to mapped interfaces

Untranslate hits indicate connections from mapped to real interfaces

Connection Table

```
firepower# show conn detail
2 in use, 64511 most used
Flags: A - awaiting responder ACK to SYN, a - awaiting initiator ACK to SYN,
       b - TCP state-bypass or nailed,
       C - CTIQBE media, c - cluster centralized,
       D - DNS, d - dump, E - outside back connection, e - semi-distributed,
       F - initiator FIN, f - responder FIN,
       G - group, g - MGCP, H - H.323, h - H.225.0, I - initiator data,
       i - incomplete, J - GTP, j - GTP data, K - GTP t3-response
       k - Skinny media, M - SMTP data, m - SIP media, N - inspected by Snort, n - GUP
       O - responder data, P - inside back connection,
       q - SQL*Net data, R - initiator acknowledged FIN,
       R - UDP SUNRPC, r - responder acknowledged FIN,
       T - SIP, t - SIP transient, U - up,
       V - VPN orphan, v - M3UA W - WAAS,
       w - secondary domain backup,
       X - inspected by service module,
       x - per session, Y - director stub flow, y - backup stub flow,
       Z - Scansafe redirection, z - forwarding stub flow
TCP outside:198.133.219.25/80 dmz:10.9.9.3/4101,
    flags UION, idle 8s, uptime 10s, timeout 1h, bytes 127
UDP outside:172.18.124.1/123 dmz:10.1.1.9/123,
    flags -, idle 15s, uptime 16s, timeout 2m, bytes 1431
```

N flag shows if connection is sent to snort

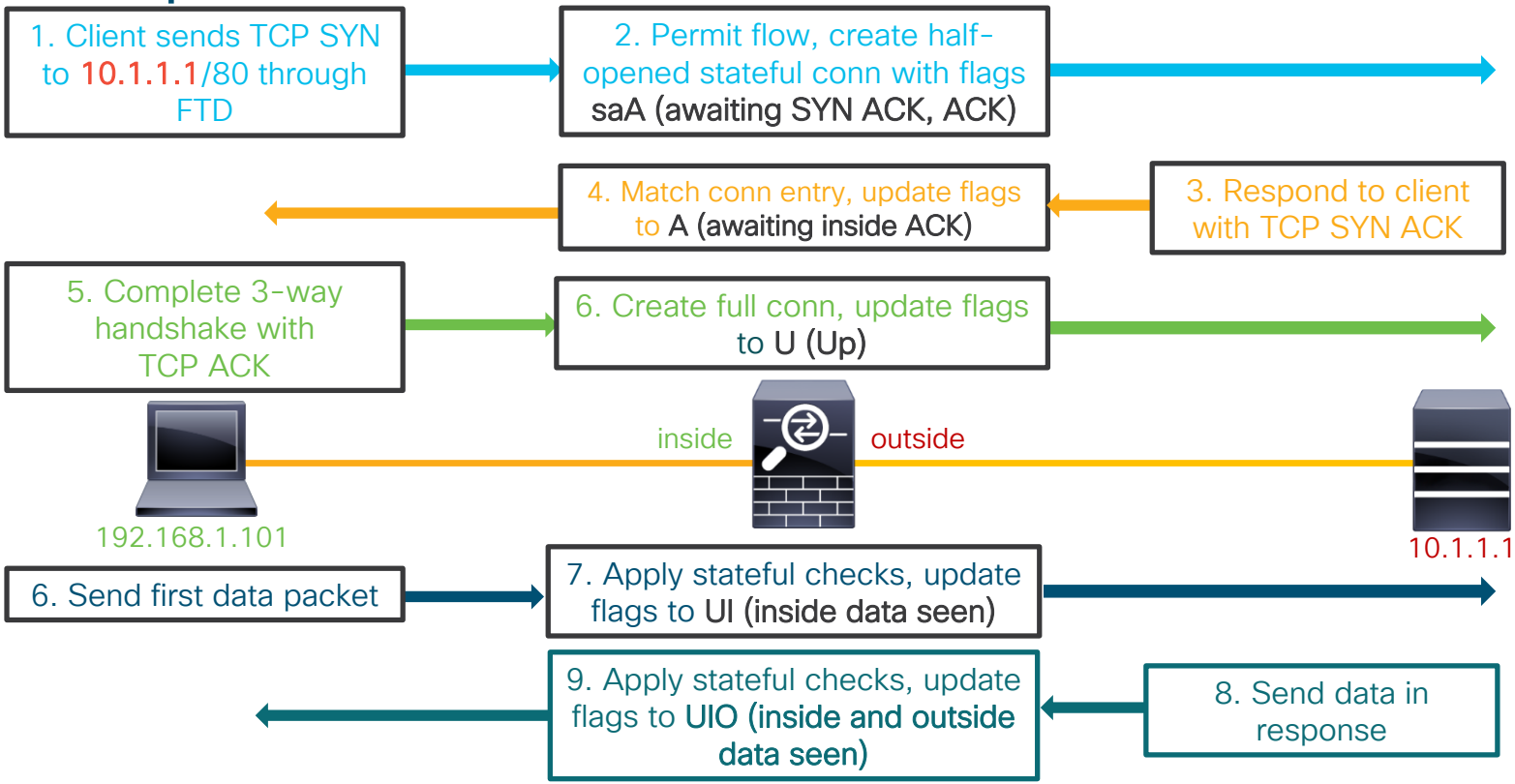
Narrow down the output with show conn address <ip>

Bidirectional byte count; use NSEL to report each direction separately.

Conn flags indicate current state

detail option adds uptime and timeout information

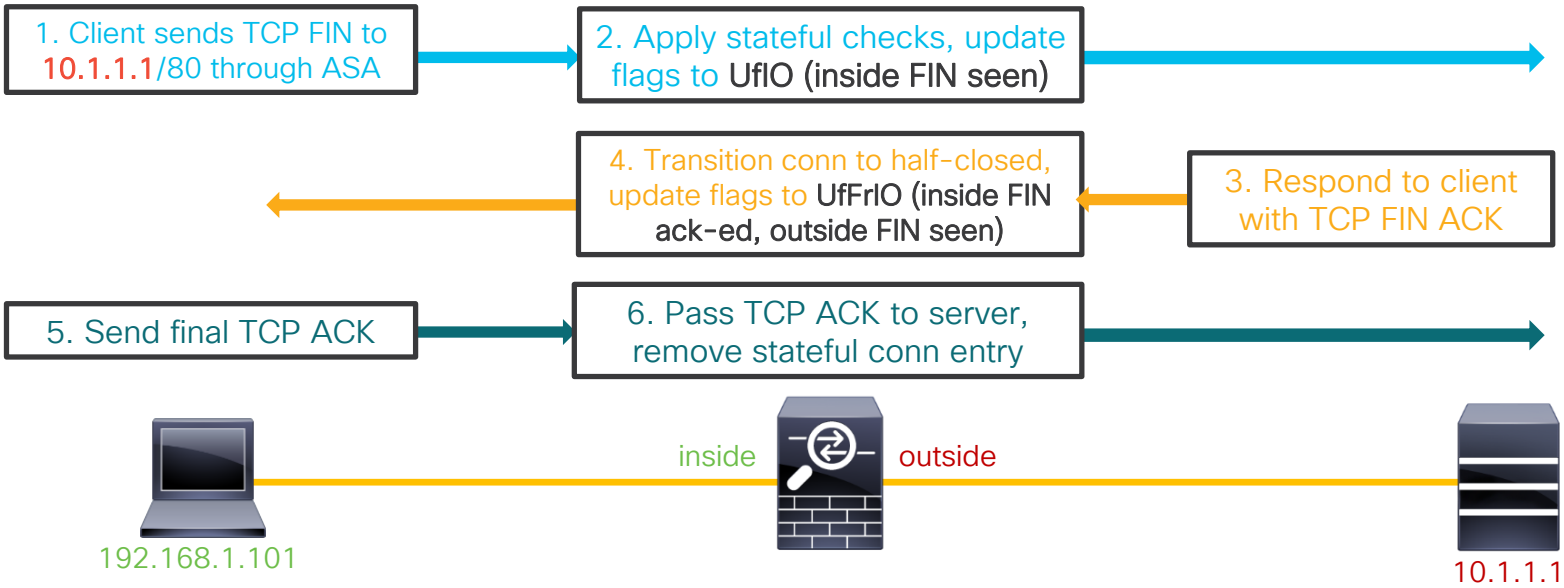
Example: TCP Connection Establishment




```
TCP outside 10.1.1.1:80 inside 192.168.1.101:50141, idle 0:00:00, bytes 153, flags UIO
```

Example: TCP Connection Termination

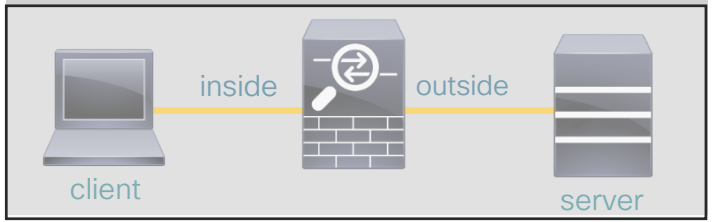
```
TCP outside 10.1.1.1:80 inside 192.168.1.101:50141, idle 0:00:00, bytes 153, flags UIO
```



TCP Connection Flags

Outbound Connection

TCP Flags		Conn Flags
SYN	→	saA
SYN+ACK	←	A
ACK	→	U
Inbound data	←	UI
Outbound data	→	UIO
FIN	→	UfIO
FIN+ACK	←	UfFrIO
ACK	→	~~UfFRrIO~~



Inbound Connection

TCP Flags		Conn Flags
SYN	←	saAB
SYN+ACK	→	aB
ACK	←	UB
Inbound data	←	UIB
Outbound data	→	UIOB
FIN	←	UFIOB
FIN+ACK	→	UfFRIOB
ACK	←	~~UBfFRrIOB~~

TCP Connection Termination Reasons

- If logging messages are enabled and a TCP flow was built through FTD, it will always log a teardown reason
- TCP teardown message is logged at level 6 (informational) by default
- For problems with abnormal connection termination, temporarily increase logging level and check the teardown reason

What do these termination reasons mean in the Teardown TCP connection syslog?

```
%ASA-6-302014: Teardown TCP connection 90 for outside:10.1.1.1/80 to
inside:192.168.1.101/1107 duration 0:00:30 bytes 0 SYN Timeout
```

```
%ASA-6-302014: Teardown TCP connection 3681 for DMZ:172.16.171.125/21 to
inside:192.168.1.110/24245 duration 0:01:03 bytes 12504 TCP Reset-O
```

TCP Connection Termination Reasons

Reason	Description
Conn-Timeout	Connection Ended Because It Was Idle Longer Than the Configured Idle Timeout
Deny Terminate	Flow Was Terminated by Application Inspection
Failover Primary Closed	The Standby Unit in a Failover Pair Deleted a Connection Because of a Message Received from the Active Unit
FIN Timeout	Force Termination After Ten Minutes Awaiting the Last ACK or After Half-Closed Timeout
Flow Closed by Inspection	Flow Was Terminated by Inspection Feature
Flow Terminated by IPS	Flow Was Terminated by IPS
Flow Reset by IPS	Flow Was Reset by IPS
Flow Terminated by TCP Intercept	Flow Was Terminated by TCP Intercept
Invalid SYN	SYN Packet Not Valid
Idle Timeout	Connection Timed Out Because It Was Idle Longer than the Timeout Value
IPS Fail-Close	Flow Was Terminated Due to IPS Card Down
SYN Control	Back Channel Initiation from Wrong Side

TCP Connection Termination Reasons

Reason	Description
SYN Timeout	Force Termination After Twenty Seconds Awaiting Three-Way Handshake Completion
TCP Bad Retransmission	Connection Terminated Because of Bad TCP Retransmission
TCP Fins	Normal Close Down Sequence
TCP Invalid SYN	Invalid TCP SYN Packet
TCP Reset-I	TCP Reset Was Sent From the Inside Host
TCP Reset-O	TCP Reset Was Sent From the Outside Host
TCP Segment Partial Overlap	Detected a Partially Overlapping Segment
TCP Unexpected Window Size Variation	Connection Terminated Due to a Variation in the TCP Window Size
Tunnel Has Been Torn Down	Flow Terminated Because Tunnel Is Down
Unauth Deny	Connection Denied by URL Filtering Server
Unknown	Catch-All Error
Xlate Clear	User Executed the 'Clear Xlate' Command

Local Host Table

- A local-host entry is created for every IP tracked by FTD
- It groups xlates, connections, and AAA information
- Useful for monitoring connections terminating on servers or offending clients

```
firepower# show local-host detail connection tcp 50 embryonic
Interface dmz: 0 active, 0 maximum active, 0 denied
Interface inside: 1 active, 1 maximum active, 0 denied
local host: <192.168.103.220>,
    TCP flow count/limit = 798/unlimited
    TCP embryonic count to host = 0
    TCP intercept watermark = unlimited
    UDP flow count/limit = 0/unlimited
 Conn:
    TCP outside:172.18.124.76/80 inside:192.168.103.220/34078,
        flags UO, idle 0s, uptime 0s, timeout 30s, bytes 0
    TCP outside:172.18.124.76/80 inside:192.168.103.220/34077,
        flags UO, idle 0s, uptime 0s, timeout 30s, bytes 0
   (output truncated)
```

Can be added to show only half-open connections

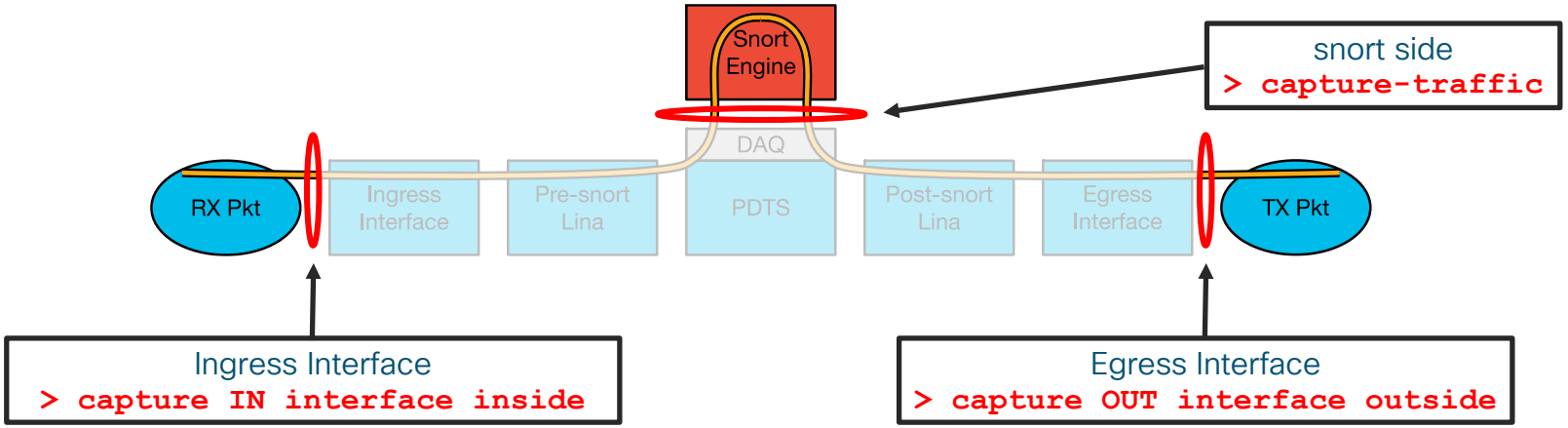
Only display hosts that have more than 50 active TCP connections.

Accelerated Security Path (ASP)

- Packets and flows dropped in the ASP will increment a counter
  - Frame drop counters are per packet
  - Flow drops are per flow
- See command reference under **show asp drop** for full list of counters

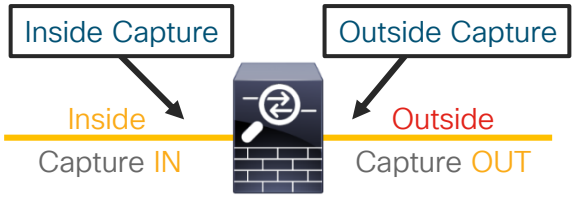
```
> show asp drop

Frame drop:
   Invalid encapsulation (invalid-encap)                  10897
   Invalid tcp length (invalid-tcp-hdr-length)             9382
   Invalid udp length (invalid-udp-length)                   10
   No valid adjacency (no-adjacency)                       5594
   No route to host (no-route)                             1009
   Reverse-path verify failed (rpf-violated)                 15
   Flow is denied by access rule (acl-drop)             25247101
   First TCP packet not SYN (tcp-not-syn)                 36888
   Bad TCP Checksum (bad-tcp-cksum)                         893

…
```

Where Packets Are Captured in Packet Flow



- Ingress packets are captured **before** most packet processing
- Egress packets are captured **after** all processing
- "`>capture-traffic`" is a capture in snort which shows packets read from the DAQ

Lina Packet Capture (CLI)



- Inline capability to record packets passing through FTD
- Apply capture under unique name to ingress and egress interfaces
  - Define the traffic that you want to capture, use pre-NAT “on the wire” information
  - Tcpdump-like format for displaying captured packets on the box

```
firepower# capture OUT interface outside match ip any host 172.18.124.1
firepower# capture IN interface inside match ip any host 172.18.124.1
firepower# show capture IN

4 packets captured

   1: 10:51:26.139046       802.1Q vlan#10 P0 172.18.254.46 > 172.18.124.1: icmp: echo request
   2: 10:51:26.139503       802.1Q vlan#10 P0 172.18.124.1 > 172.18.254.46: icmp: echo reply
   3: 10:51:27.140739       802.1Q vlan#10 P0 172.18.254.46 > 172.18.124.1: icmp: echo request
   4: 10:51:27.141182       802.1Q vlan#10 P0 172.18.124.1 > 172.18.254.46: icmp: echo reply
4 packets shown
firepower# no capture IN interface inside
firepower# no capture IN
```

Unlike ACL, **match** covers both directions of the flow

Removing the interface stops the capture but keeps contents in memory

Remember to remove the captures when done with troubleshooting

Lina Packet Capture (CLI)

- Capture buffer maintained in RAM (512KB by default, 33 MB max)
  - Stops capturing when full by default, **circular** option available
- Default recorded packet length is 1518 bytes
- May elevate CPU utilization when applied under very high packet rates
- Copy captures off via FTP, SCP, or TFTP (example below)

Much larger capture sizes coming soon!

```
firepower# capture OUT interface outside match ip any host 172.18.124.1
firepower# copy /pcap capture:OUT tftp://10.10.1.1/capout.pcap
```

Configured capture name

Save capture file under this name

Download binary PCAP to open in your favorite packet analyser (such as Wireshark)

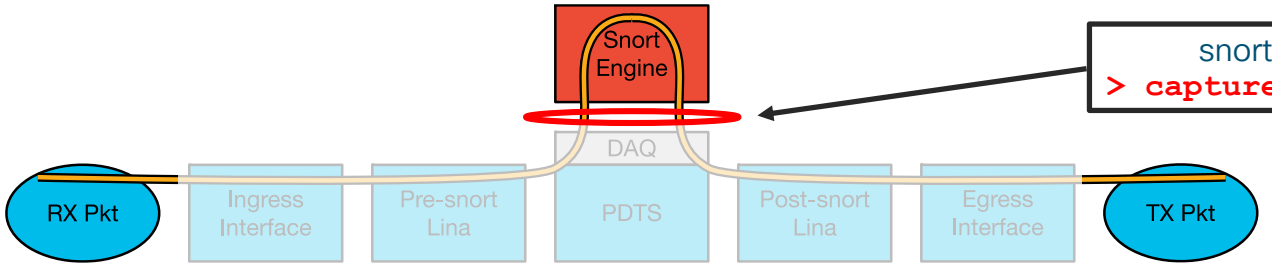
Packet Capture at time of Crash

- Before 6.2.2, Lina capture contents are lost if the device crashes
- New feature allows use of a circular buffer to capture all traffic just before a crash occurs
- Very useful for troubleshooting traffic-related crashes

Introduced in FTD 6.2.2

```
firepower# capture capin interface inside circular-buffer buffer 33000000
<<after forcing crash>>
firepower# show flash:
--#--  --length--  -----date/time------  path
  109  198         Dec 09 2017 00:59:00  lina_phase1.log
<<output truncated>>
  110  1761873     Jan 22 2019 10:36:34  capin.pcap
  111  502025      Jan 22 2019 10:36:42  crashinfo_20190122_103635_UTC
```

Snort-side captures with > capture-traffic



snort side
> capture-traffic

```
> capture-traffic

Please choose domain to capture traffic from:
  0 - br1
  1 - Router

Selection? 1

Please specify tcpdump options desired.
(or enter '?' for a list of supported options)
Options: -n -s 0 -w SNORTCAP.pcap -c 1000 host 192.168.1.2 and port 80
```

tcpdump -c 1000
Stop after 1000 packets

Standard BPF
(Berkeley Packet Filter) Options

tcpdump -n
Don't resolve hostnames

tcpdump -s 0
Capture the whole packet

tcpdump -w FILE.pcap
Write the capture to file

> capture-traffic
PCAPs are written to:
/ngfw/var/common/

Capturing ASP drops

- Capture all frames dropped in the ASP

```
firepower# capture drops type asp-drop all
```

- Capture all frames with a specific drop reason

```
firepower# capture drop type asp-drop ?

  acl-drop                        Flow is denied by configured
  rule
  all                             All packet drop reasons
  bad-crypto                      Bad crypto return in packet
  bad-ipsec-natt                  Bad IPSEC NATT packet
  bad-ipsec-prot                  IPSEC not AH or ESP
  bad-ipsec-udp                   Bad IPSEC UDP packet
  bad-tcp-cksum                   Bad TCP checksum
  bad-tcp-flags                   Bad TCP flags
```

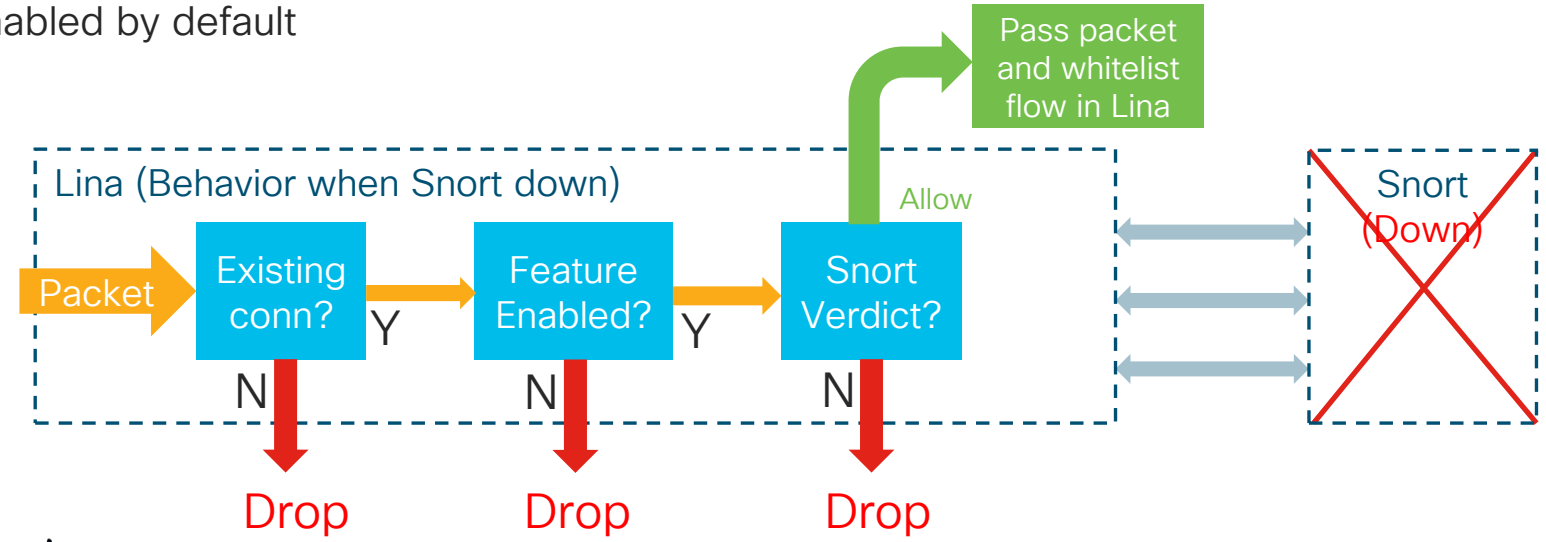
- ASP flow drops are non-atomic and cannot be captured

```
firepower# capture drops type asp-drop tcp-not-syn
```

In FTD you can filter ASP drops using an inline match statement like a normal packet capture

Snort Preserve-Connection

- Allows packets to pass while snort is down/restarting
- Flow must have reached an “Allow” verdict (AC policy)
- Added in 6.2.3
- Enabled by default



Snort Preserve-Connection: Enable/Disable

Show Current Setting

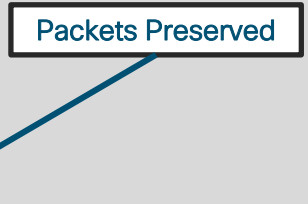
```
> show running-config snort
snort preserve-connection
```

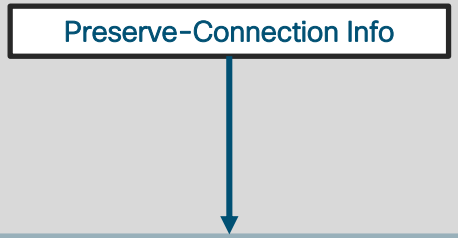
Change Setting

```
> configure snort preserve-connection disable
Building configuration...
Cryptochecksum: 4fd6de40 7bf66af6 b1836604 04f8496d

5745 bytes copied in 0.690 secs
[OK]
> show running-config snort
no snort preserve-connection
```

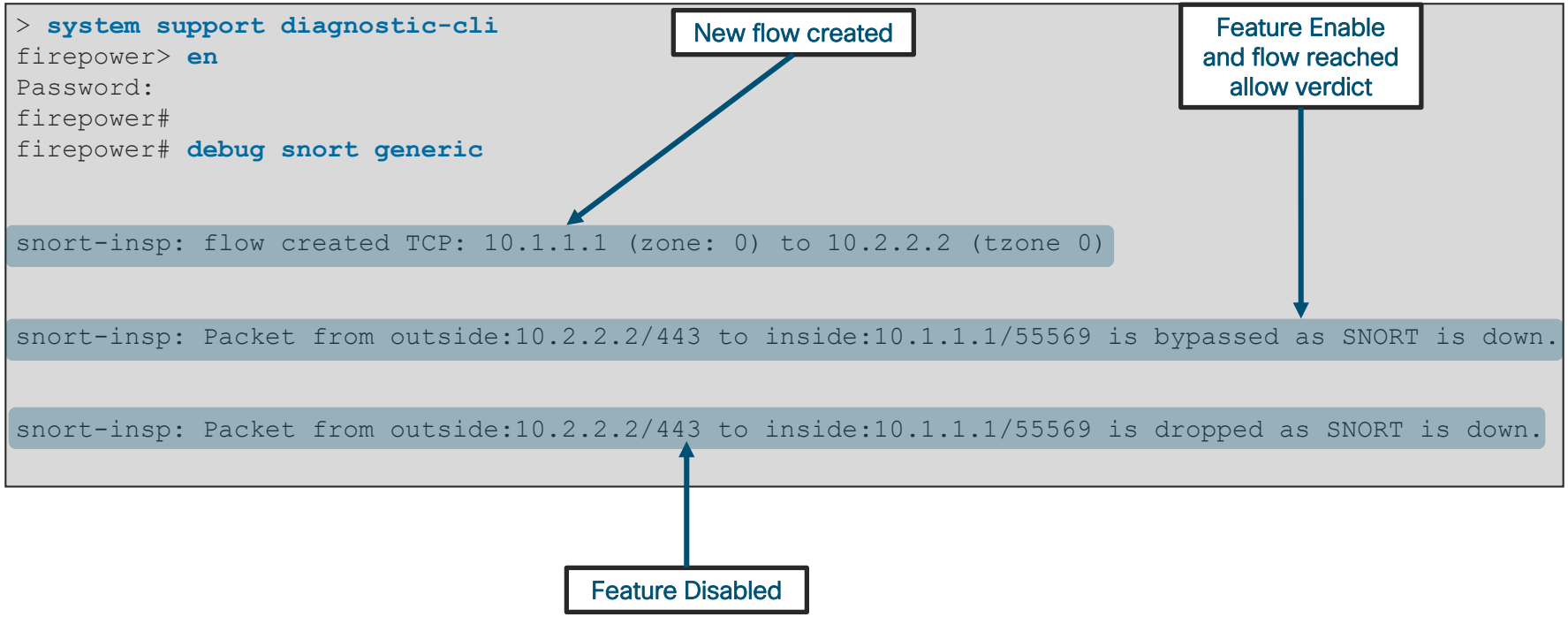
Snort Preserve-Connection: Troubleshooting

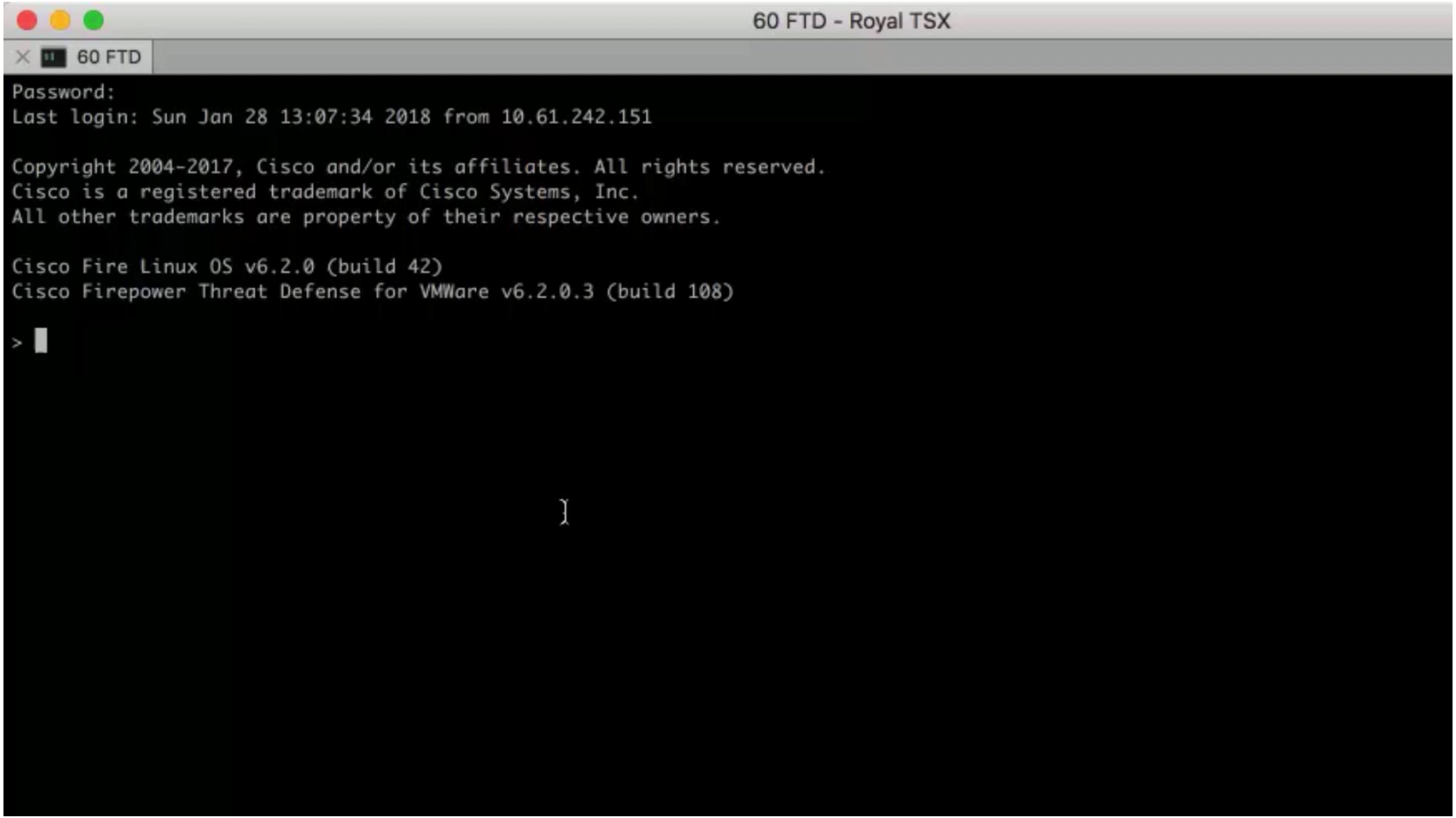
```
> show snort statistics

Packet Counters:
  Passed Packets                                  62501
  Blocked Packets                                  2339
  Injected Packets                                 5739
  Packets bypassed (Snort Down)                    5678
  Packets bypassed (Snort Busy)                       0


[lines removed]

> show conn

0 in use, 231 most used

Inspect Snort:

    preserve-connection: 14 enabled, 5 in effect, 215 most enabled, 40 most in effect

[lines removed]
```

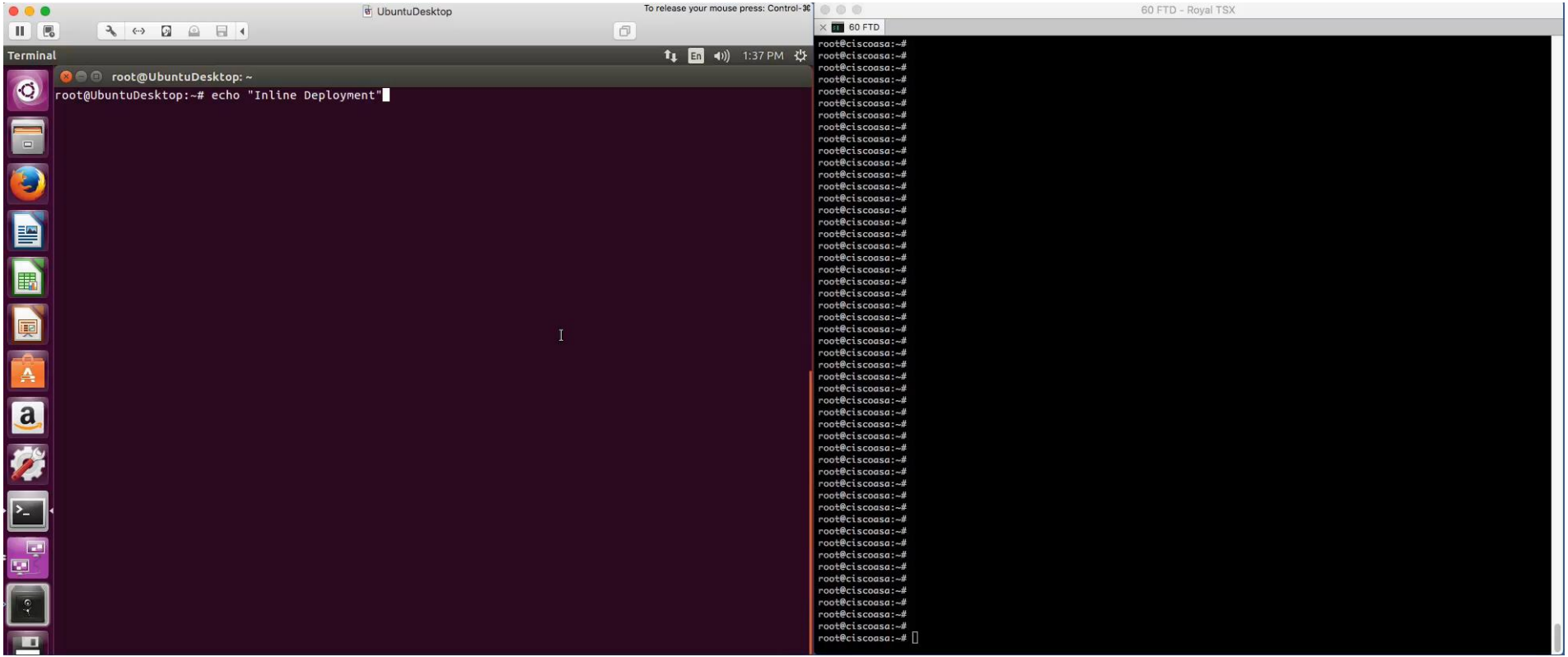


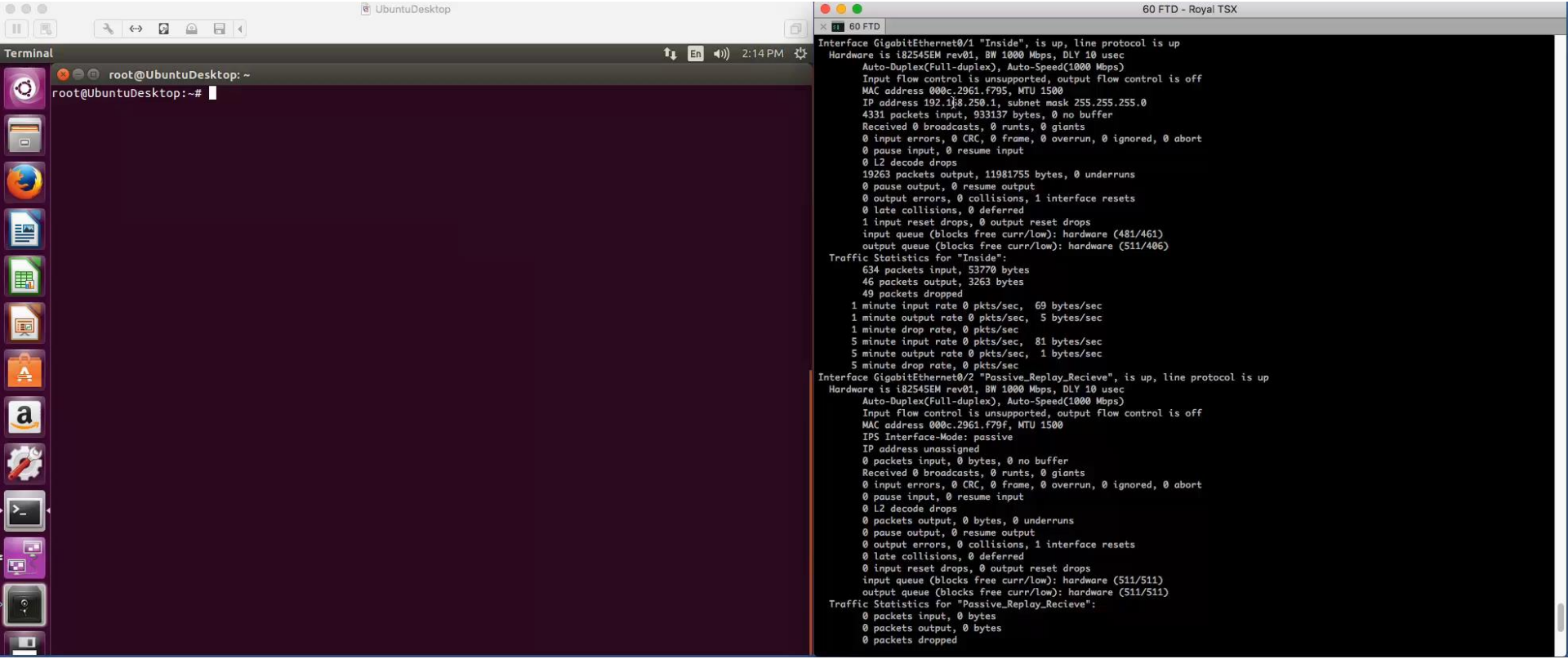


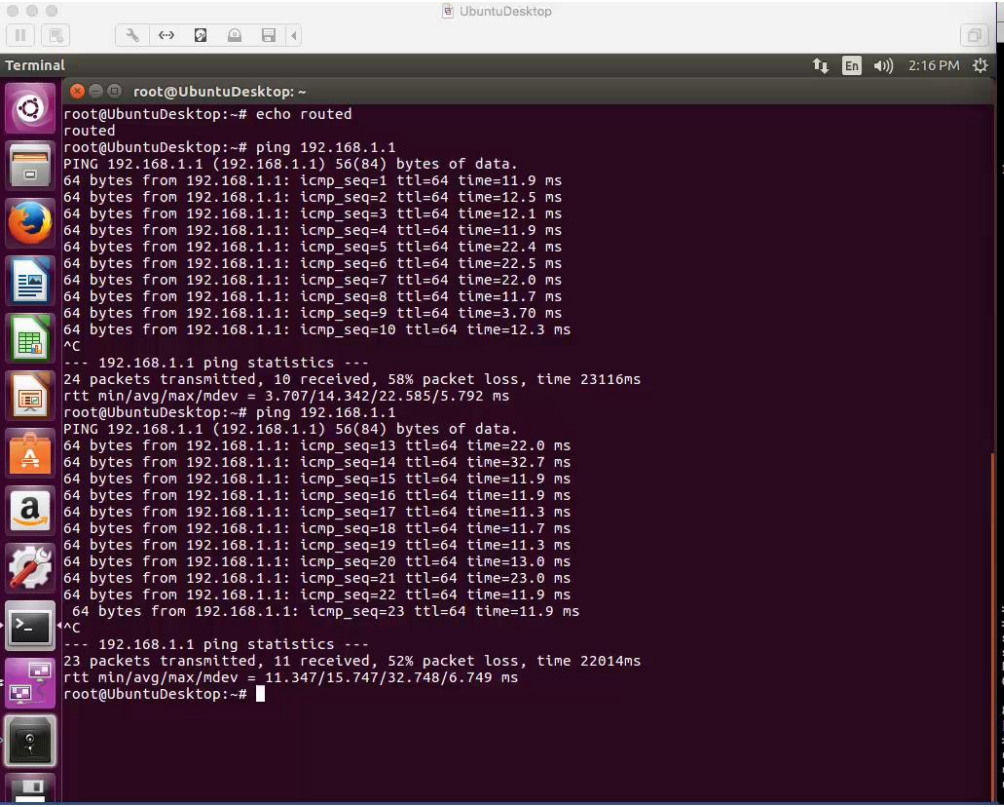
Snort Preserve-Connection: Troubleshooting

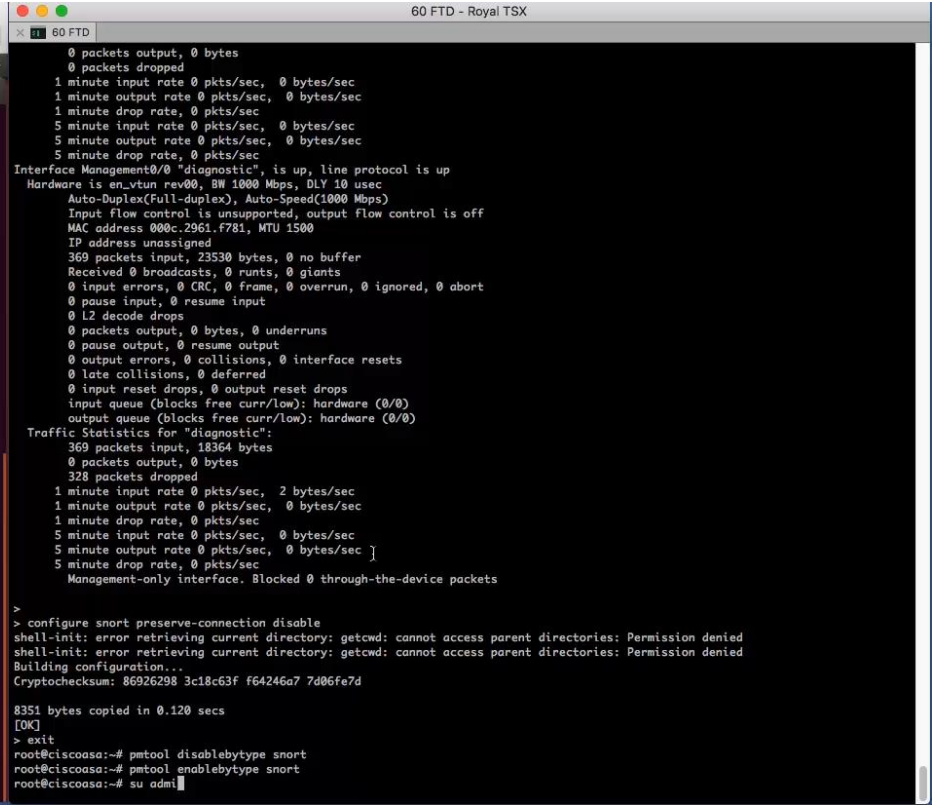


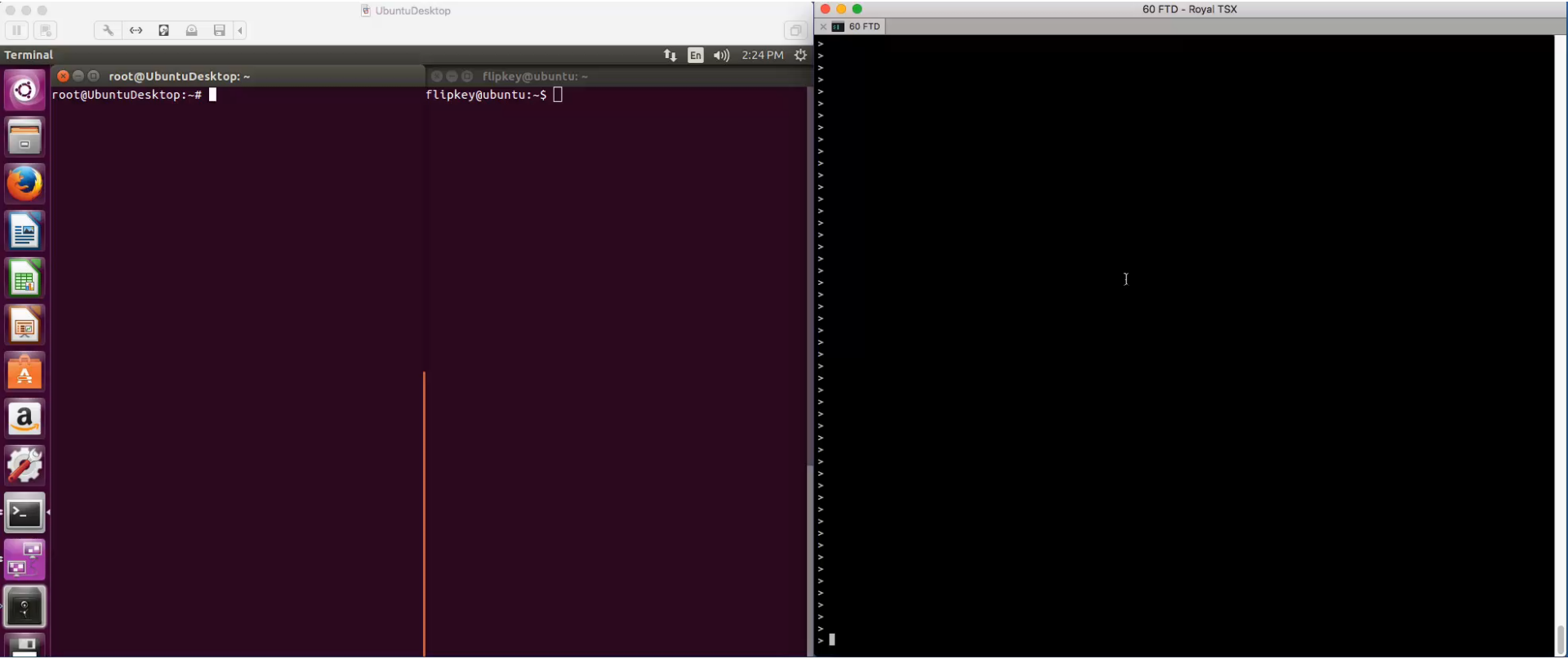








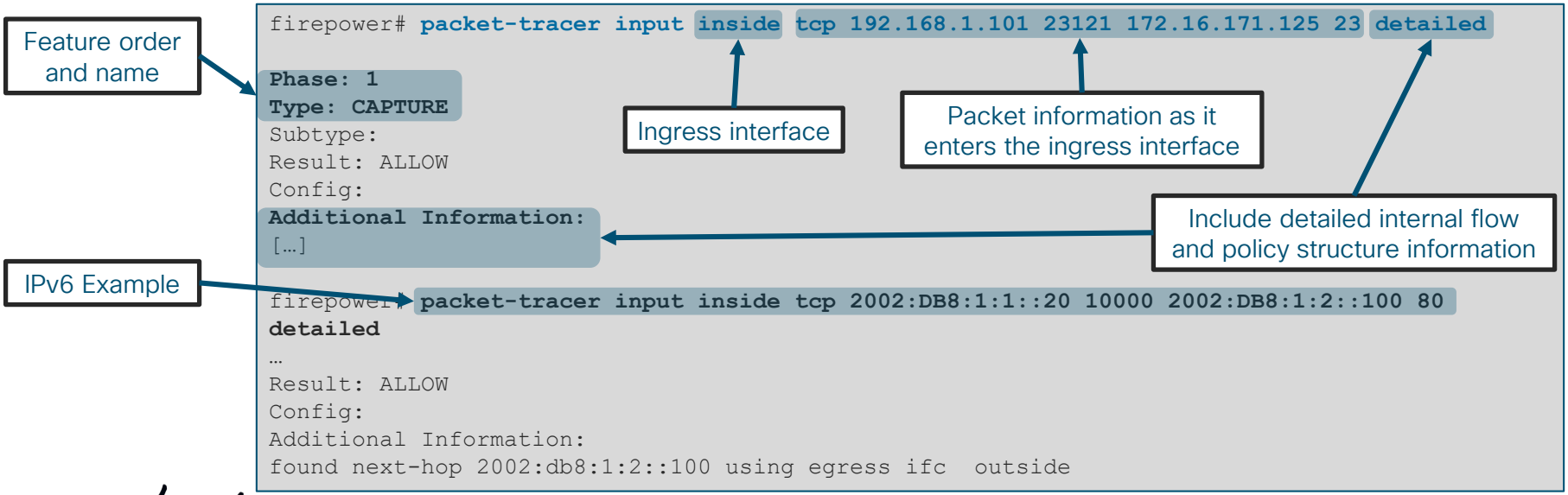




Packet Tracer

Packet Tracer

- Unique capability to record the path of a specially tagged packet through FTD
  - Best way to understand the packet path in the specific software version
- Inject a simulated packet to analyse the behaviour and validate configuration



Sample Packet Tracer Output

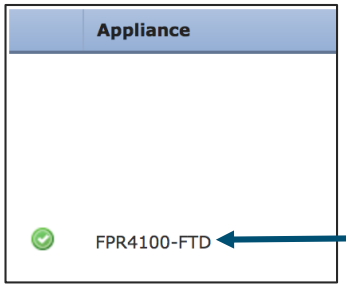
```
firepower# packet-tracer input outside tcp 172.18.124.66 1234 172.18.254.139 3389

Phase: 1
Type: CAPTURE
Subtype:
Result: ALLOW
Config:
Additional Information:
MAC Access list

Phase: 2
Type: ACCESS-LIST
Subtype:
Result: ALLOW
Config:
Implicit Rule
Additional Information:
MAC Access list

Phase: 3
Type: UN-NAT
Subtype: static
Result: ALLOW
Config:
nat (outside,dmz) source dynamic any interface destination static interface Win7-vm service rdp-outside rdp-outside
Additional Information:
NAT divert to egress interface dmz
Untranslate 172.18.254.139/3389 to 192.168.103.221/3389
```

Sample Packet Tracer Output (Cont'd)

```
Phase: 4
Type: ACCESS-LIST
Subtype: log
Result: ALLOW
Config:
access-group outside_in in interface outside
access-list outside_in extended permit tcp any any eq 3389
Additional Information:
……
Phase: 8
Type: NAT
Subtype:
Result: ALLOW
Config:
nat (outside,dmz) source dynamic any interface destination static interface Win7-vm service rdp-outside rdp-outside
Additional Information:
Dynamic translate 172.18.124.66/1234 to 192.168.103.221/1234
……
Phase: 12
Type: FLOW-CREATION
Subtype:
Result: ALLOW
Config:
Additional Information:
New flow created with id 16538274, packet dispatched to next module
```

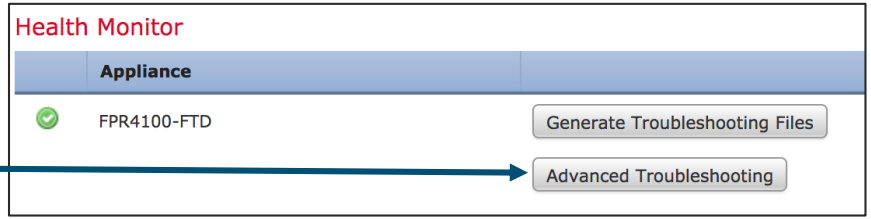
Packet Tracer in FMC

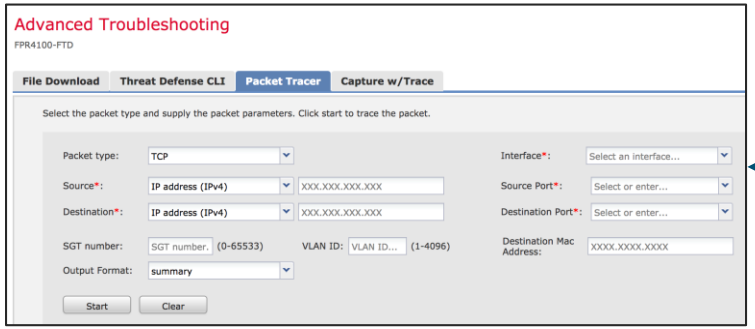
1 Launch from System Health > Monitor



Click on appliance hostname 2

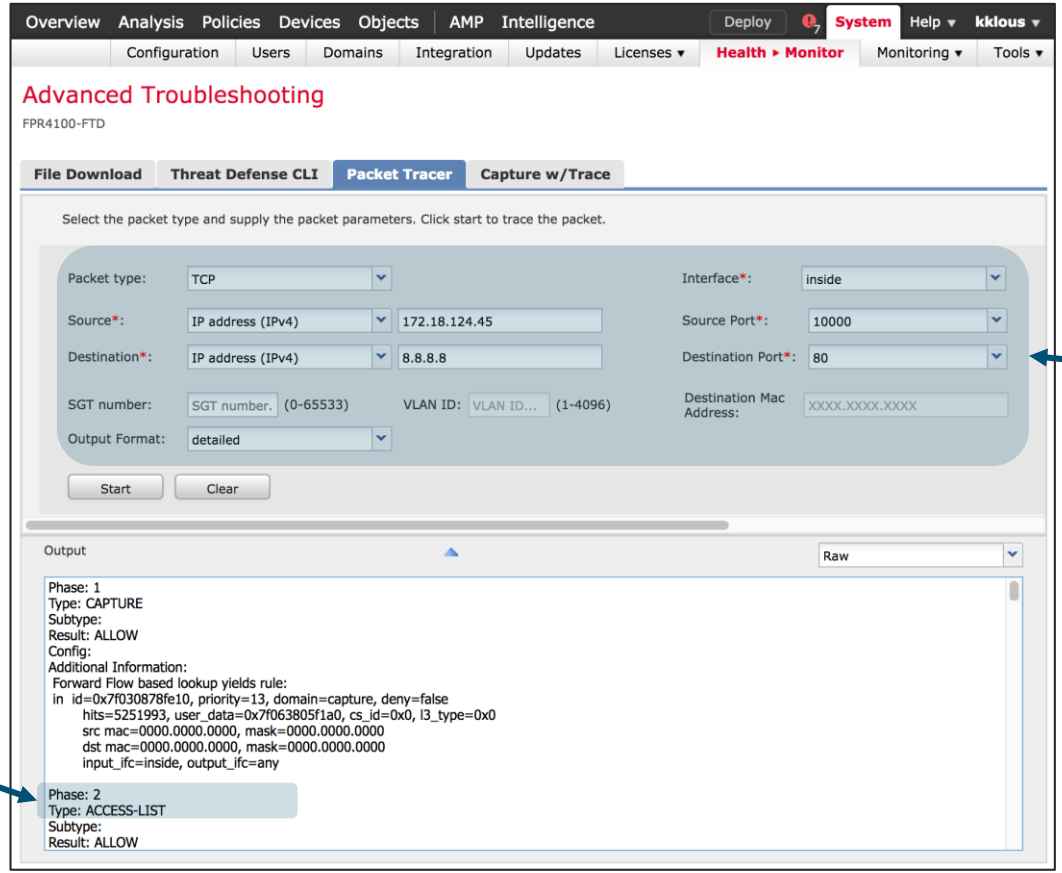
3 Advanced Troubleshooting





Input packet parameters 4

Packet Tracer in FMC – Example Output



Define simulated packet

Feature type and resulting action

Packet Capture w/ Trace

- Enable packet tracer within an internal packet capture

```
firepower# capture IN interface inside trace trace-count 200 match tcp any any eq
```

Trace inbound packets only

Traced packet count per capture (1-1000, 50 by default)

- Find the packet that you want to trace in the capture

```
firepower# show capture inside
  68 packets captured
  1: 15:22:47.581116 10.1.1.2.31746 > 198.133.219.25.80: S
  2: 15:22:47.583465 198.133.219.25.80 > 10.1.1.2.31746: S ack
  3: 15:22:47.585052 10.1.1.2.31746 > 198.133.219.25.80: . ack
  4: 15:22:49.223728 10.1.1.2.31746 > 198.133.219.25.80: P ack
  5: 15:22:49.223758 198.133.219.25.80 > 10.1.1.2.31746: . Ack
     ...
```

- Select that packet to show the tracer results

```
firepower# show capture inside trace packet-number 4
```

Packet capture with trace (continued)

- Likely the most used datapath troubleshooting tool in the TAC
- Troubleshooting capabilities continue to be developed – major improvements in FTD 6.2.3+:
  - You can now capture traffic post-decryption across a VPN tunnel w/ FTD as VPN endpoint:

```
firepower# capture OUT interface outside trace include-decrypted match tcp any any
```

New option captures packets that match the criteria after decryption

New packet-tracer option to allow egress of simulated packets

```
firepower#  packet-tracer input inside tcp 10.1.1.20 10000 10.1.2.100 80 transmit detailed
firepower#  sh cap capout
1 packet captured
   1: 12:08:30.837709       10.1.1.20.10000 > 10.1.2.100.80: S 1119191062:1119191062(0) win
```

Without this option, the packet is never transmitted onto the wire. This can be useful for troubleshooting.

Firewall Engine Debug / System Support Trace

Firewall Engine Debug (Snort)

- Shows Snort access control rule evaluation
- Indicates which rule a flow matches

```
> system support firewall-engine-debug

 Please specify an IP protocol: tcp
 Please specify a client IP address: 192.168.1.2
 Please specify a client port:
 Please specify a server IP address:
 Please specify a server port: 80

192.168.1.2-35948 > 172.16.2.10-80 6 AS 1 I 18 New session
[lines_removed]
192.168.1.2-35948 > 172.16.2.10-80 6 AS 1 I 18 match rule order 2, 'Block Port HTTP
Traffic', action Block
```

Common IP Header “Protocol” values:
1 or "icmp”
6 or "tcp”
17 or "udp"

Leave a field blank for “any”

- Debug is written to messages log file

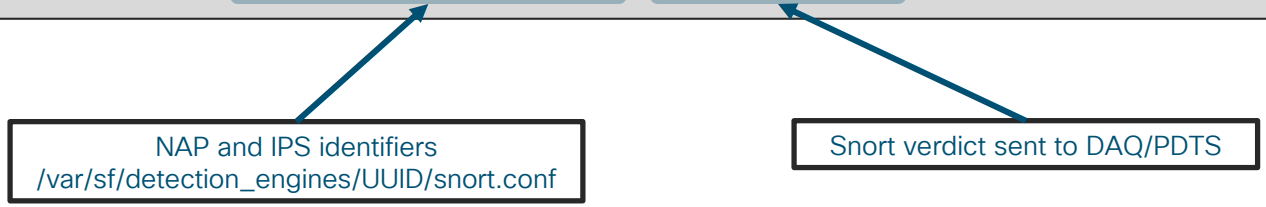
```
grep -i ngfwdbg /var/log/messages
```

System Support Trace (Snort)

> system support trace

- Debugs a flow in snort per packet (be careful!)
- Can optionally enable parallel firewall-engine-debug
- Shows preprocessor impact (Network Analysis Policy) not shown in other outputs

```
> system support trace

[lines removed]

10.2.2.2-443 - 10.1.1.1-5623 6 Packet: TCP, ACK, seq 1448114540, ack 4072763547
10.2.2.2-443 - 10.1.1.1-5623 6 Firewall: allow rule, 'Allow_Inside_to_Outside', allow
10.2.2.2-443 - 10.1.1.1-5623 6 AppID: service HTTPS (1122), application Microsoft
(1423)
10.1.1.1-5623 > 10.2.2.2-443 6 Firewall: allow rule, 'Allow_Inside_to_Outside', allow
10.1.1.1-5623 > 10.2.2.2-443 6 NAP id 2, IPS id 0, Verdict PASS
```



Troubleshooting Protocol Preprocessors

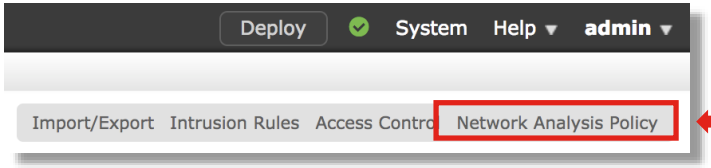
Use system support trace to find blocks by preprocessors

Trace

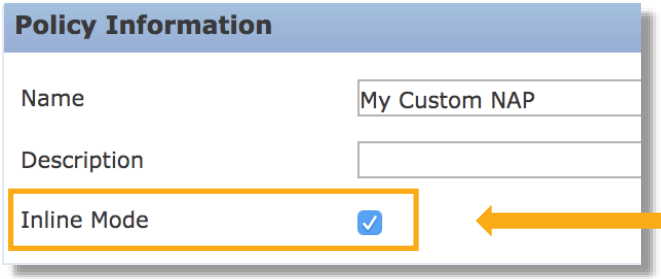
```
> system support trace

[omitted for brevity…]

172.16.111.226-51174 - 50.19.123.95-443 6 Packet: TCP, ACK, seq 3849839667, ack 1666843207
172.16.111.226-51174 - 50.19.123.95-443 6 Stream: TCP normalization error in timestamp, window, seq, ack,
fin, flags, or unexpected data, drop
172.16.111.226-51174 - 50.19.123.95-443 6 AppID: service unknown (0), application unknown (0)
172.16.111.226-51174 > 50.19.123.95-443 6 AS 4 I 0 Starting with minimum 3, 'block urls', and SrcZone first
with zones -1 -> -1, geo 0 -> 0, vlan 0, inline sgt tag: untagged, ISE sgt id: 0, svc 0, payload 0, client
0, misc 0, user 9999997, icmpType 0, icmpCode 0
172.16.111.226-51174 > 50.19.123.95-443 6 Firewall: starting rule matching, zone -1 -> -1, geo 0 -> 0, vlan
0, sgt 65535, user 9999997, icmpType 0, icmpCode 0
172.16.111.226-51174 > 50.19.123.95-443 6 AS 4 I 0 pending rule order 3, 'block urls', URL
172.16.111.226-51174 > 50.19.123.95-443 6 Firewall: pending rule-matching, 'block urls', pending URL
172.16.111.226-51174 > 50.19.123.95-443 6 Snort: processed decoder alerts or actions queue, drop
172.16.111.226-51174 > 50.19.123.95-443 6 IPS Event: gid 129, sid 14, drop
172.16.111.226-51174 > 50.19.123.95-443 6 NAP id 1, IPS id 0, Verdict BLOCK
172.16.111.226-51174 > 50.19.123.95-443 6 ===> Blocked by Stream
```

Policies > Access Control > Intrusion

Disable Inline Mode



Edit or create a Network Analysis Policy



Uncheck this box to disable Inline Mode

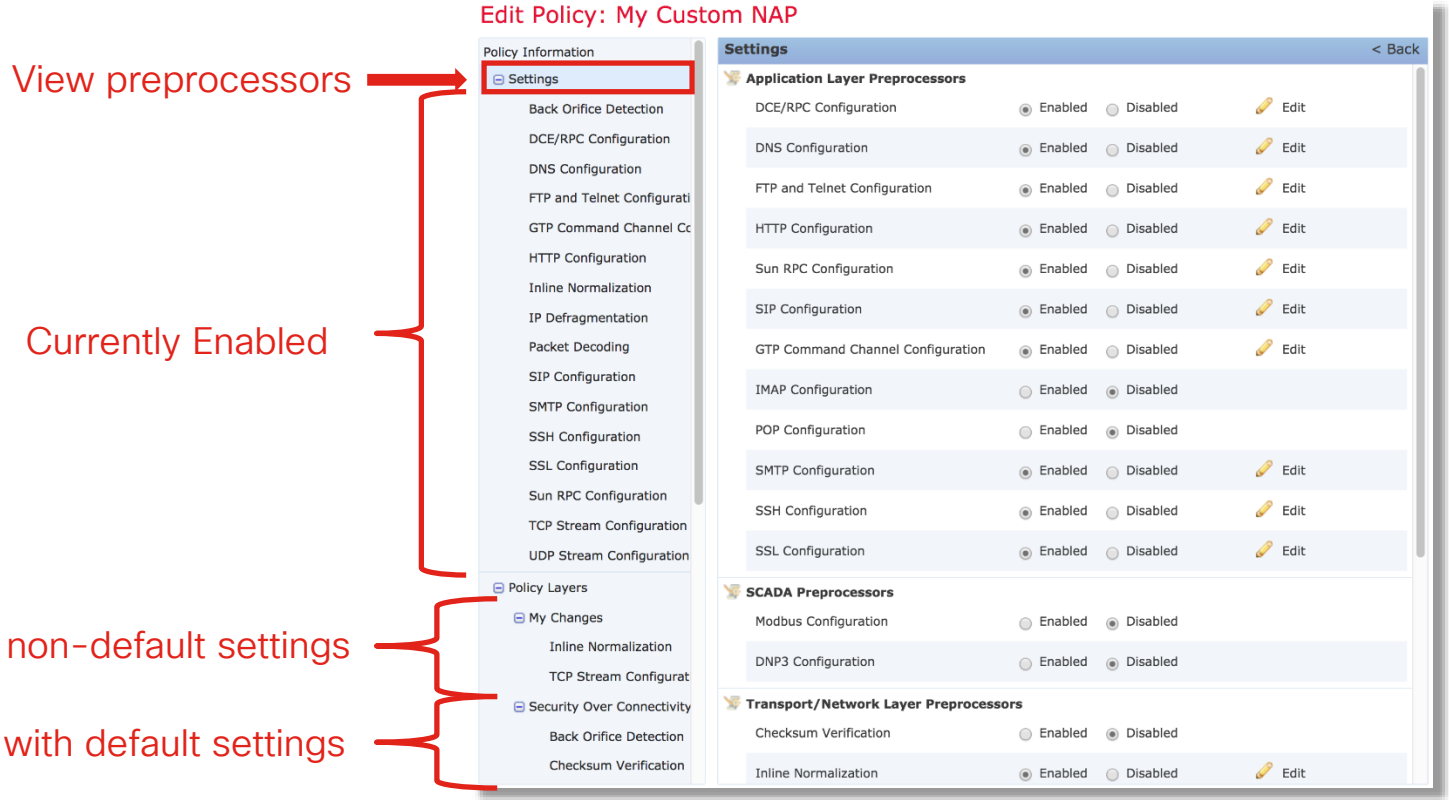
Inline Result	Source IP	Destination IP	Source Port / ICMP Type	Destination Port / ICMP Code	Message
	172.16.111.226	50.19.123.95	51177 / tcp	443 (https) / tcp	STREAM5_NO_TIMESTAMP (129:14:2)
↓	172.16.111.226	50.19.123.95	51174 / tcp	443 (https) / tcp	STREAM5_NO_TIMESTAMP (129:14:2)

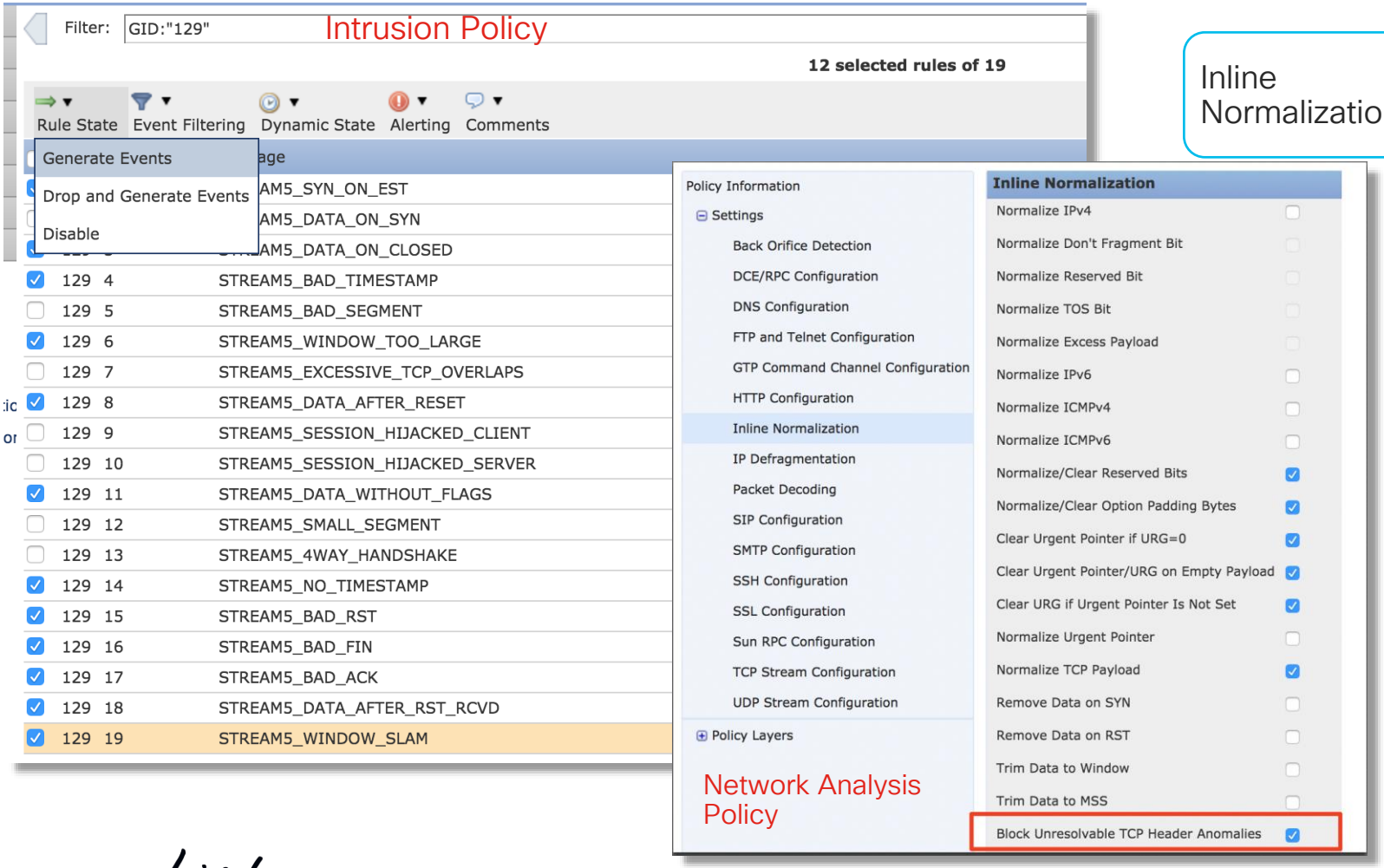
Inline Mode disabled = No Inline Result

Inline Mode enabled = “Dropped” Inline Result

Troubleshooting Protocol Preprocessors

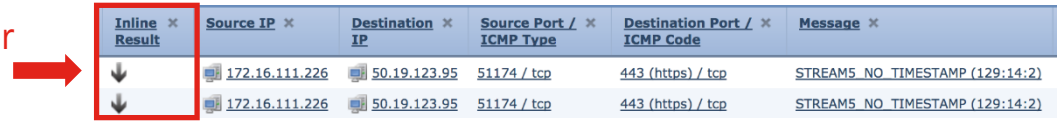
Network Analysis Configuration





Inline Normalization

Still drops after setting to generate



Inline Result	Source IP	Destination IP	Source Port / ICMP Type	Destination Port / ICMP Code	Message
↓	172.16.111.226	50.19.123.95	51174 / tcp	443 (https) / tcp	STREAM5_NO_TIMESTAMP (129:14:2)
↓	172.16.111.226	50.19.123.95	51174 / tcp	443 (https) / tcp	STREAM5_NO_TIMESTAMP (129:14:2)

Inline Normalization

Check configuration guide for relative protocols/preprocessors:

Block Unresolvable TCP Header Anomalies

When you enable this option, the system blocks anomalous TCP packets that, if normalized, would be invalid and likely would be blocked by the receiving host. For example, the system blocks any SYN packet transmitted subsequent to an established session.

The system also drops any packet that matches any of the following TCP stream preprocessor rules, regardless of whether the rules are enabled:

- 129:1
- 129:3
- 129:4
- 129:6
- 129:8
- 129:11
- 129:14 through 129:19

The Total Blocked Packets performance graph tracks the number of packets blocked in inline deployments and, in passive deployments and inline deployments in tap mode, the number that would have been blocked in an inline deployment.

Config guides: http://www.cisco.com/c/en/us/support/security/defense-center/products-installation-and-configuration-guides-list.html

TCP Ping

- Powerful troubleshooting tool in FTD
- Verify bi-directional TCP connectivity from FTD to a remote server using injected packet
- Provides FTD policy and upstream path verification without client host access
  - TCP RST and ICMP error responses are intercepted and displayed as well

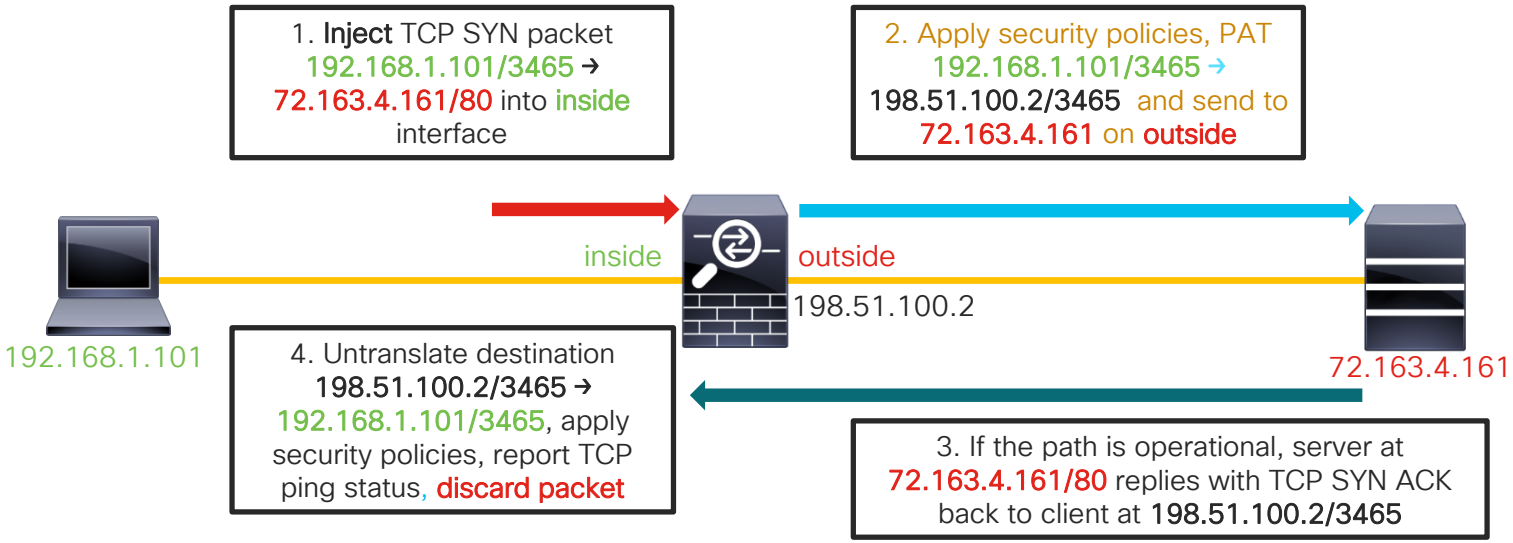
Example: TCP Ping

```
firepower# ping tcp
 Interface: inside
 Target IP address: 72.163.4.161
 Target IP port: 80
 Specify source? [n]: y
 Source IP address: 192.168.1.101
 Source IP port: [0]
 Repeat count: [5]
 Timeout in seconds: [2]
 Type escape sequence to abort.
 Sending 5 TCP SYN requests to 72.163.4.161 port 80
 from 192.168.1.101 starting port 3465, timeout is 5 seconds:
 !!!!!
 Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
```

Interface where the test host resides

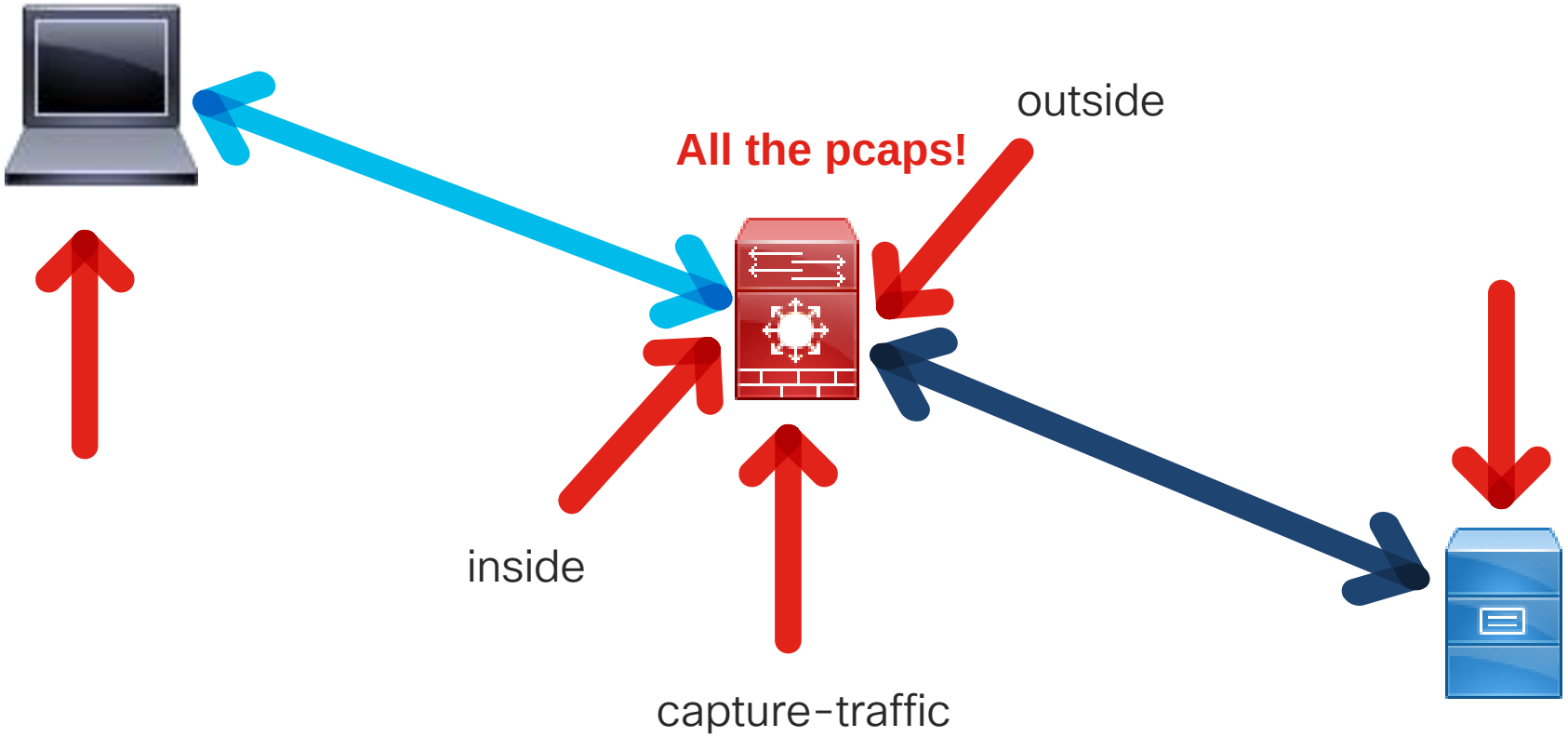
Real IP address of the test host; the host does not have to be online or even connected

Example: TCP Ping

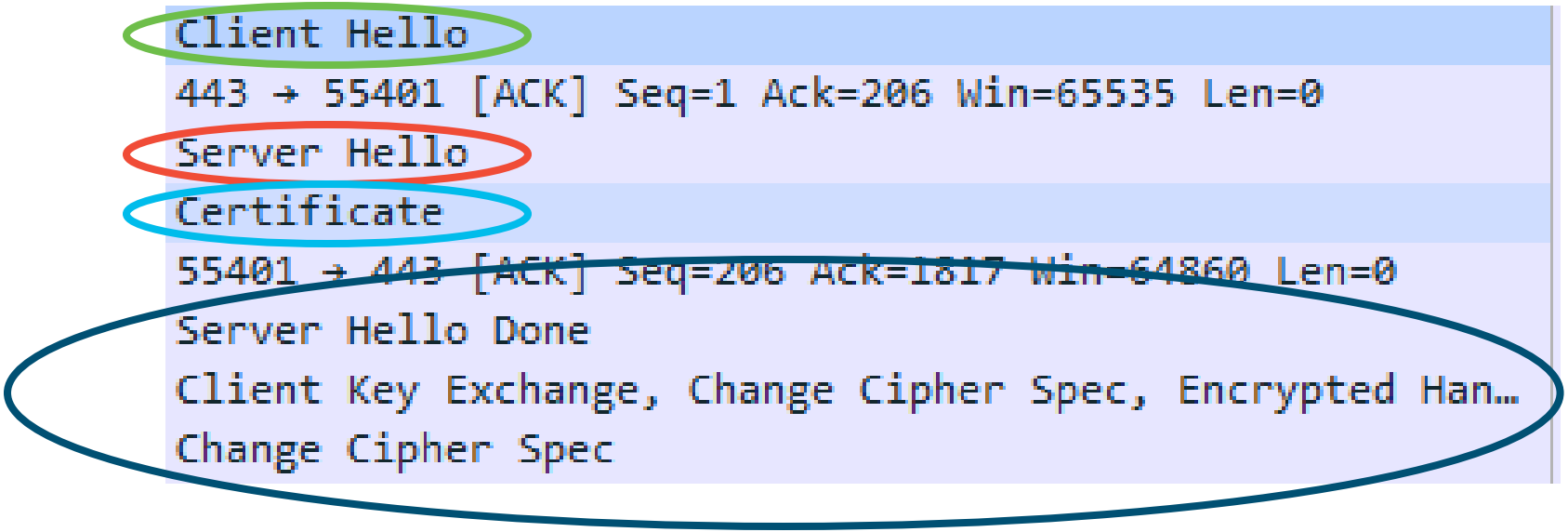


Packet Captures for SSL Decryption

Pcaps



Full handshake (Wireshark view)

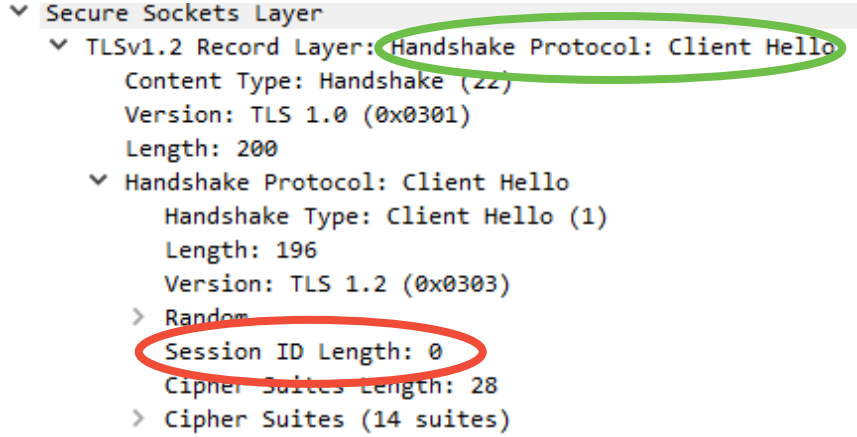


Other common handshakes

- Client Certificate request 
- Session Reuse 
  - Client Hello Modification required
- Certificate Pinning 
  - Phone applications

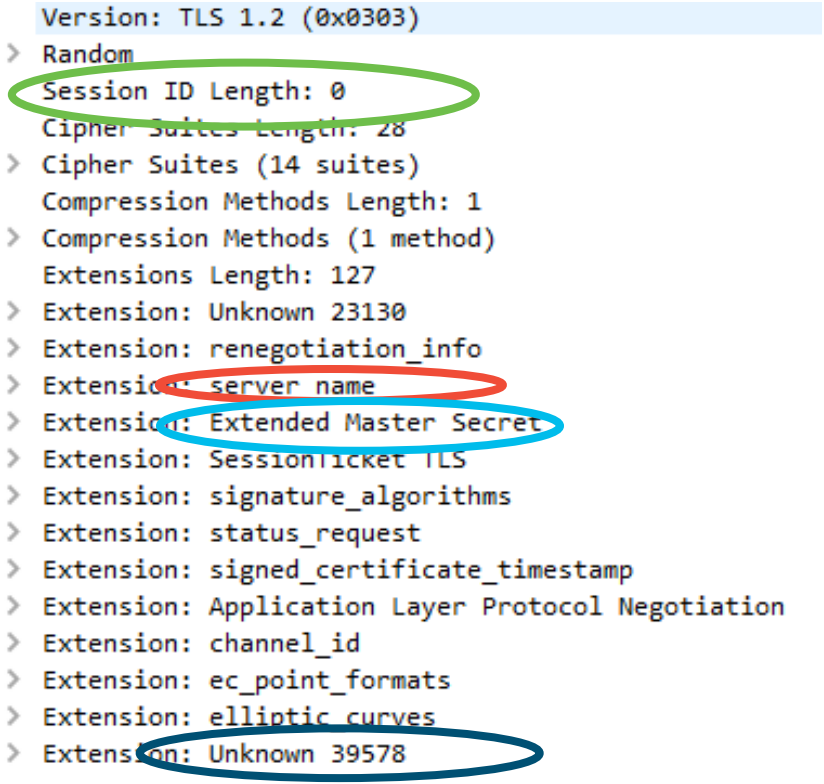
Pcap investigation Client Hello

- Identify Handshake
- Session ID



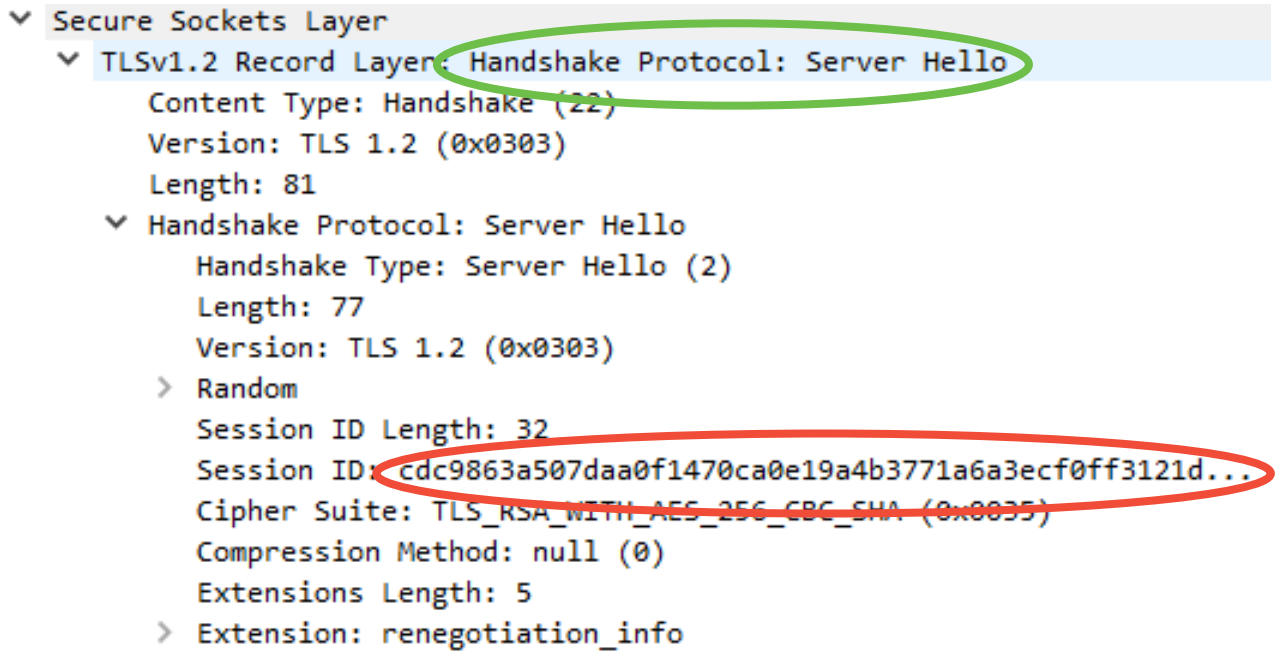
Pcap investigation Client Hello continued

- Session ID
- Server Name
- Known problems
- Potential problems



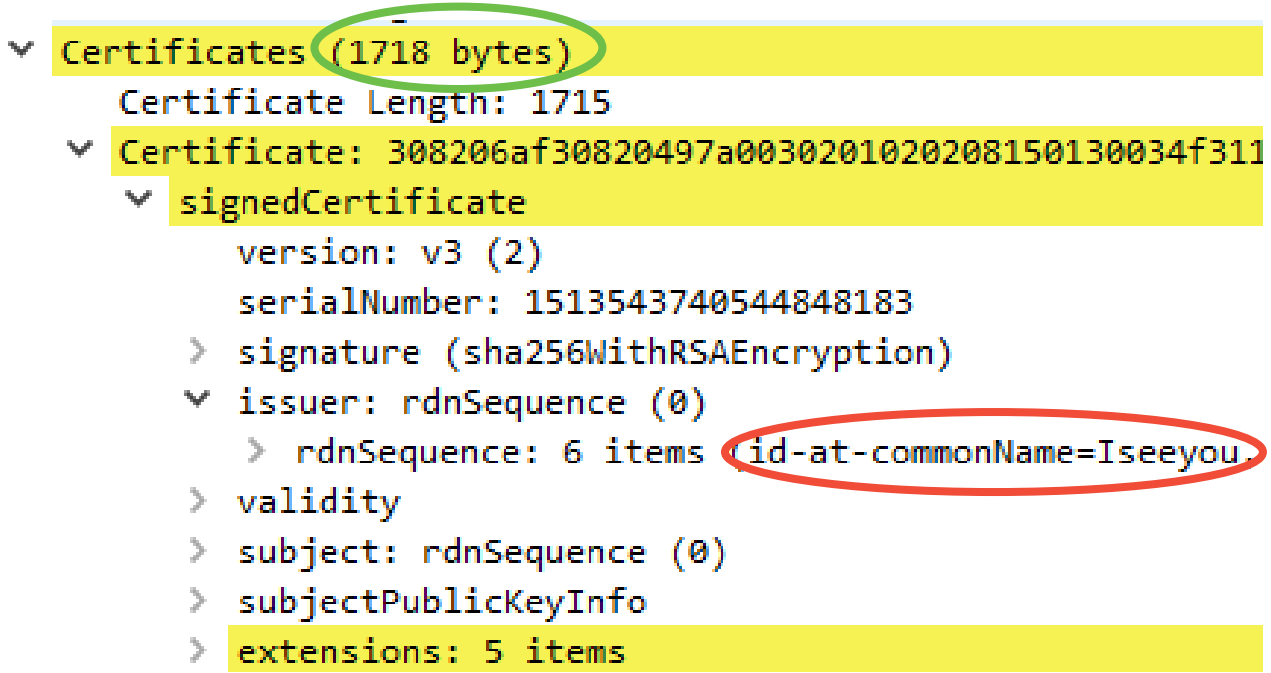
Pcap investigation: Server Hello

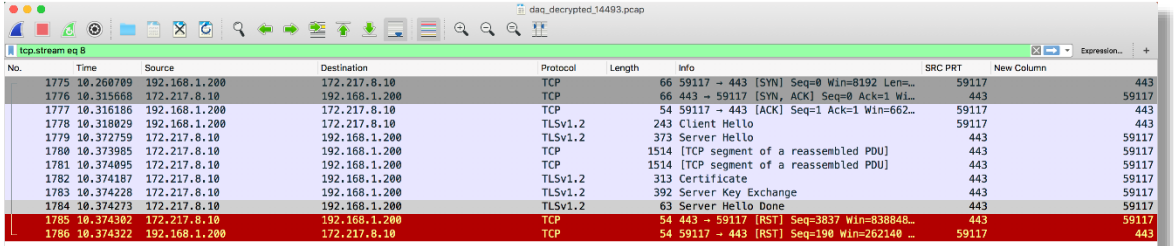
- Identify Handshake
- Session ID



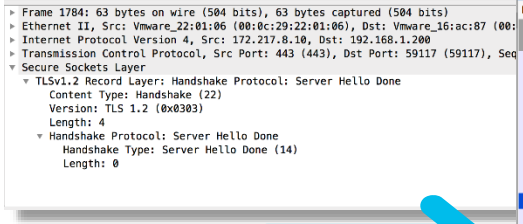
Pcap investigation: Certificate

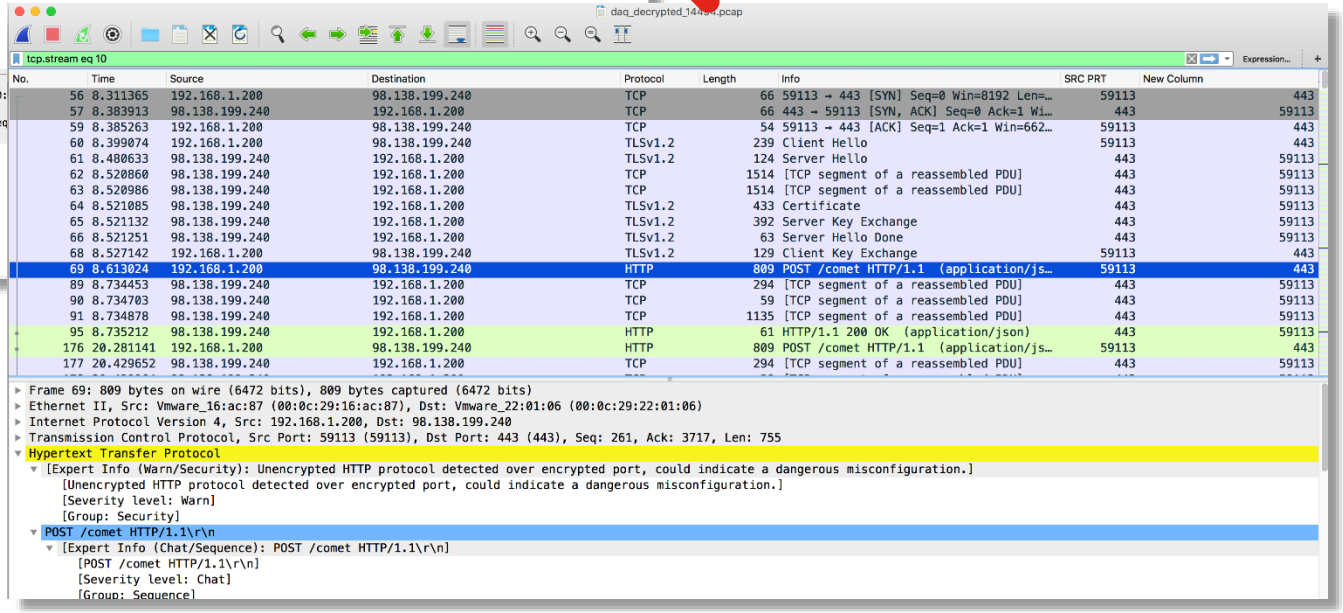
- Length
- Issuer





Decryption fails

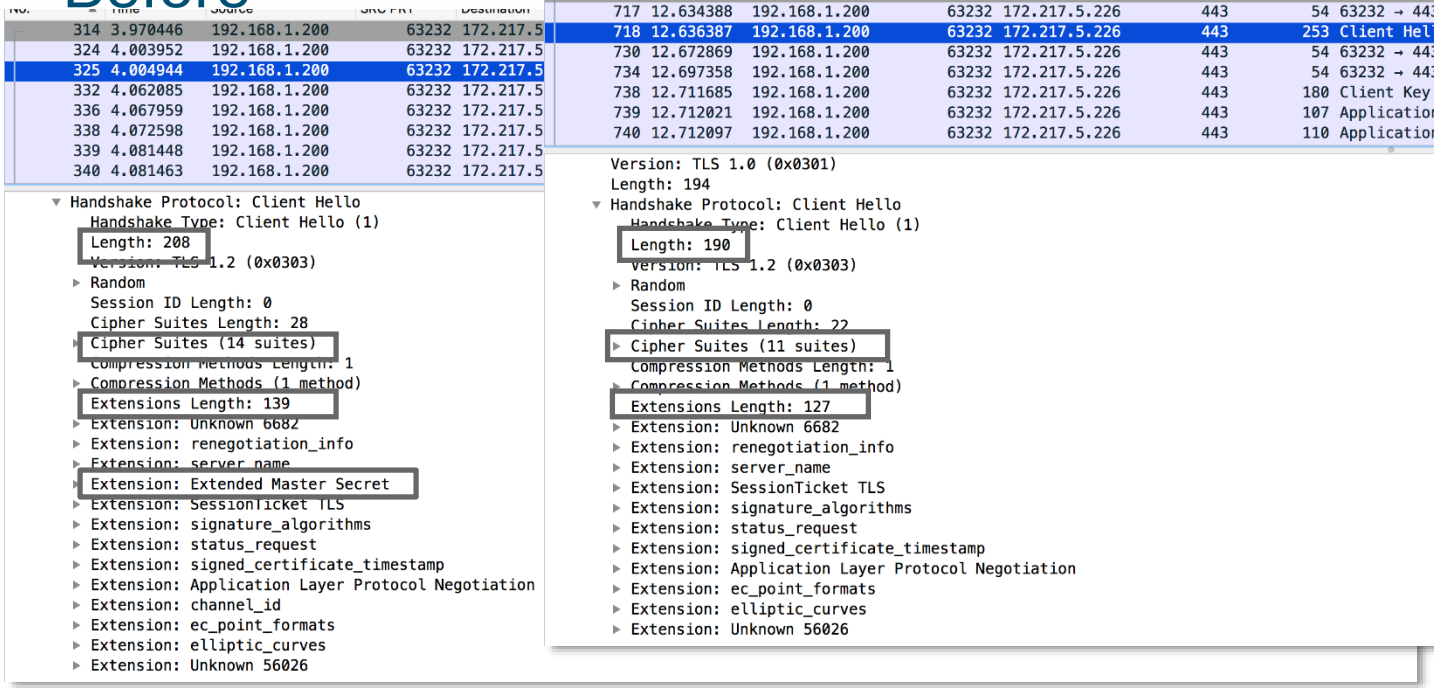




Decryption succeeds

Client Hello Modification (6.1.0+)

Before



After

Identity Troubleshooting Tools

Firewall engine debug

Firewall Engine Debug is the right tool to identify what is happening within the Access Control Policy

```
> system support firewall-engine-debug

Please specify an IP protocol: tcp
Please specify a client IP address: 172.16.1.2
Please specify a client port:
Please specify a server IP address: 192.168.0.10
Please specify a server port: 8081

Monitoring firewall engine debug messages

172.16.1.2-54255 > 192.168.0.10-8081 6 AS 1 I 0 New session
172.16.1.2-54255 > 192.168.0.10-8081 6 AS 1 I 0 Starting with minimum 4, 'Allow_Group2', and
IPProto first with zones 1 -> 2, geo 0 -> 0, vlan 0, inline sgt tag: untagged, ISE sgt id: 0,
svc 0, payload 0, client 0, misc 0, user 1, icmpType 0, icmpCode 0
172.16.1.2-54255 > 192.168.0.10-8081 6 AS 1 I 0 rule order 4, 'Allow_Group2', did not match
group 2
172.16.1.2-54255 > 192.168.0.10-8081 6 AS 1 I 0 no match rule order 4, 'Allow_Group2', user
1, realm 2
172.16.1.2-54255 > 192.168.0.10-8081 6 AS 1 I 0 match rule order 5, id 268434432 action Allow
172.16.1.2-54255 > 192.168.0.10-8081 6 AS 1 I 0 allow action
```

ID of currently mapped user:

1 - 999999X = Downloaded User
9999995 = Pending User
9999996 = Guest
9999997 = No Auth Required
9999998 = Failed Authentication
9999999 = Unknown

Identity-debug

The Identity-debug tool allows the user to troubleshoot the Identity Policy.

```
> system support identity-debug

Please specify an IP protocol: tcp
Please specify a client IP address: 172.16.1.2
Please specify a client port:
Please specify a server IP address: 192.168.0.10
Please specify a server port: 8081
Monitoring identity debug messages

172.16.1.2-43490 > 192.168.0.10-8081 6 AS 1 I 0 Starting authentication (sfAuthCheckRules params)
with zones 2 -> 3, port 43490 -> 8081, geo 16429296 -> 16429314
172.16.1.2-43490 > 192.168.0.10-8081 6 AS 1 I 0 Starting Auth SrcZone first with zones 2 -> 3, geo
2 -> 3, vlan 0
172.16.1.2-43490 > 192.168.0.10-8081 6 AS 1 I 0 Matched rule order 1, id 1, authRealmId 2, AD
Domain fire.int
172.16.1.2-43490 > 192.168.0.10-8081 6 AS 1 I 0 found captive portal session
172.16.1.2-43490 > 192.168.0.10-8081 6 AS 1 I 0 returning captive portal session
172.16.1.2-43490 > 192.168.0.10-8081 6 AS 1 I 0 found active binding for user_id 1
172.16.1.2-43490 > 192.168.0.10-8081 6 AS 1 I 0 matched auth rule id = 1 user_id = 1 realm_id = 2
```

The user_map_query script

```
root@FTD# user_map_query.pl -h

Options:
--dump-data <pre_str>        Dumps all troubleshooting data for user/group mapping. If provided,
                             the output files will be prepended with "<pre_str>_"
-d, --debug                  enable debug logging (off by default)
-g, --group                  Displays the users associated to the group(s) specified (can not be
                             passed with -i or -u)
-h, -?, --help               Print usage information
-i, --ip-addr                Displays the users associated to the IPv4 address(es) specified (can
                             not be passed with -g or -u)
--iu                         Include unified file data
--outfile                    Dumps the output to the specified file
-s, --snort                  Include data from snort's mapping
-u, --user                   Displays the IP addresses associated to the user(s) specified (can
                             not be passed with -g or -i)
--unified-all                Displays all of the unified data per record regardless of the type
                             of query
--unified-dir                The directory to look for unified files (default is
                             /var/sf/user_enforcement)
--use-id                     Treats the values passed as IDs (only relevant for user and group
                             queries)
```

Collect All Data

Troubleshoot Live

Finding who that User ID belongs to

```
root@FTD# user_map_query.pl --use-id -u 1

Current Time: 01/17/2019 15:54:38 UTC

Getting information on username(s)...

___
User #1: test1
___

  ID:         1
  Last Seen:  Unknown
  for_policy: 0
  Realm ID:   2

==============================
|          Database          |
==============================

##) IP Address [Realm ID]
 1) ::ffff:172.16.1.2 [2]

##) Group Name (ID)
 1) Test (3)
```

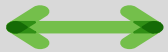
Username

Currently Mapped IP Address(s)

Groups user belongs to

Comparing Database and Snort output

```
root@FTD/home/admin# user_map_query.pl -s -u test1

Would you like to dump user data from snort now? (Current Time: 01/17/2019 16:08:03 UTC) [y,n]: y
Successfully commanded snort.
Current Time: 01/17/2019 16:08:05 UTC
Getting information on username(s)...

___
User #1: test1
---

  ID:         1
  Last Seen:  Unknown
  for_policy: 0
  Realm ID:   2

==============================                ==============================
|          Database          |                |            Snort           |
==============================                ==============================

##) IP Address [Realm ID]                     ##) IP Address [Realm ID] (instances)
 1) ::ffff:172.16.1.2 [2]                         1) ::ffff:172.16.1.2 [2] (instance 1)

##) Group Name (ID)                           ##) Group Name (ID) (instances)
 1) Test (3)                                  1) Test (3) (instance 1)
```



Collect data to give to TAC

```
root@FTD# user_map_query.pl --dump-data CiscoLive

Would you like to dump user data from snort now? (Current Time: 01/17/2019 17:44:27 UTC) [y,n]: y

Successfully commanded snort.

Current Time: 01/17/2019 17:44:30 UTC
Getting database dumps...
Dumping table user_group_map...Done
Dumping table realm_info...Done
Dumping table user_identities...Done
Dumping table user_group...Done
Dumping table estreamer_bookmark...Done
Dumping table current_user_ip_map...Done
Dumping table user_ip_map...Done
Dumping table user_identities...Done
Done getting database dumps.
Added /var/sf/user_enforcement/* files.
Added snort data dumps
Compressing data...Done!

File: /var/tmp/CiscoLive_utd.a76e92ea-aaab-11e7-be62-c7b57db57e79.1547747070.tar.gz
Cleaning up...Done!
```

Give this to TAC

Captive Portal packet captures

Lina Capture ➡ Tun1 Capture ➡ TEST ➡ Stop Tun1 Cap ➡ Stop Lina Cap ➡ Copy Lina Cap

```
> capture ins_captport interface inside buffer 1000000 match tcp host 172.16.1.2 any
> expert

root@FTD1:# tcpdump -i tun1 -s 1518 -w /ngfw/var/common/captive_portal.pcap
HS_PACKET_BUFFER_SIZE is set to 4.tcpdump:
listening on tun1, link-type RAW (Raw IP), capture size 1518 bytes

[TEST AUTHENTICATION]

^C
99 packets captured
99 packets received by filter
0 packets dropped by kernel

root@FTD1:# exit

> capture ins_captport stop
> copy /noconfirm /pcap capture:ins_captport ins_captport.pcap
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
398 packets copied in 0.80 secs
```

Lina Capture location: /mnt/disk0/ins_captport.pcap

Tun1 Capture location: /ngfw/var/common/captive_portal.pcap

The captures at an initial glance

ins_captport.pcap

No.	Destination	Source	Protocol	Lengt	Info
261	172.16.1.1	172.16.1.2	TCP	66	52441 → 885 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 W
262	172.16.1.2	172.16.1.1	TCP	66	885 → 52441 [SYN, ACK] Seq=0 Ack=1 Win=14600 Len=
263	172.16.1.1	172.16.1.2	TCP	54	52441 → 885 [ACK] Seq=1 Ack=1 Win=65536 Len=0
264	172.16.1.1	172.16.1.2	TCP	233	52441 → 885 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=
265	172.16.1.2	172.16.1.1	TCP	54	885 → 52441 [ACK] Seq=1 Ack=180 Win=15744 Len=0
266	172.16.1.2	172.16.1.1	TCP	723	885 → 52441 [PSH, ACK] Seq=1 Ack=180 Win=15744 Le
267	172.16.1.1	172.16.1.2	TCP	268	52441 → 885 [PSH, ACK] Seq=180 Ack=670 Win=65024
268	172.16.1.2	172.16.1.1	TCP	336	885 → 52441 [PSH, ACK] Seq=670 Ack=394 Win=16768
269	172.16.1.1	172.16.1.2	TCP	571	52441 → 885 [PSH, ACK] Seq=394 Ack=952 Win=64512
270	172.16.1.2	172.16.1.1	TCP	54	885 → 52441 [ACK] Seq=952 Ack=911 Win=17920 Len=0
273	172.16.1.2	172.16.1.1	TCP	816	885 → 52441 [PSH, ACK] Seq=952 Ack=911 Win=17920

Before bltd NAT

After bltd NAT

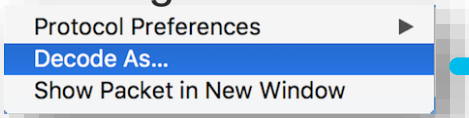
Same ports

No.	Destination	Source	Protocol	Lengt	Info
63	169.254.0.1	169.254.3.88	TCP	52	52441 → 885 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 W
64	169.254.3.88	169.254.0.1	TCP	52	885 → 52441 [SYN, ACK] Seq=0 Ack=1 Win=14600 Len=
65	169.254.0.1	169.254.3.88	TCP	40	52441 → 885 [ACK] Seq=1 Ack=1 Win=65536 Len=0
66	169.254.0.1	169.254.3.88	TCP	219	52441 → 885 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=
67	169.254.3.88	169.254.0.1	TCP	40	885 → 52441 [ACK] Seq=1 Ack=180 Win=15744 Len=0
68	169.254.3.88	169.254.0.1	TCP	709	885 → 52441 [PSH, ACK] Seq=1 Ack=180 Win=15744 L
69	169.254.0.1	169.254.3.88	TCP	254	52441 → 885 [PSH, ACK] Seq=180 Ack=670 Win=65024
70	169.254.3.88	169.254.0.1	TCP	322	885 → 52441 [PSH, ACK] Seq=670 Ack=394 Win=16768
71	169.254.0.1	169.254.3.88	TCP	557	52441 → 885 [PSH, ACK] Seq=394 Ack=952 Win=64512
72	169.254.3.88	169.254.0.1	TCP	40	885 → 52441 [ACK] Seq=952 Ack=911 Win=17920 Len=0
73	169.254.3.88	169.254.0.1	TCP	802	885 → 52441 [PSH, ACK] Seq=952 Ack=911 Win=17920

captive_portal.pcap

The captures may need to be decoded

Right click..



Choose SSL for each port

Field	Value	Type	Default	Current
TCP port	52441	Integer, base 10	(none)	SSL
TCP port	885	Integer, base 10	(none)	SSL

Raw

Protocol	Length	Info
TCP	52	52441 → 885 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SA
TCP	52	885 → 52441 [SYN, ACK] Seq=0 Ack=1 Win=14600 Len=0 MSS=14
TCP	40	52441 → 885 [ACK] Seq=1 Ack=1 Win=65536 Len=0
TCP	219	52441 → 885 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=179
TCP	40	885 → 52441 [ACK] Seq=1 Ack=180 Win=15744 Len=0
TCP	709	885 → 52441 [PSH, ACK] Seq=1 Ack=180 Win=15744 Len=669
TCP	254	52441 → 885 [PSH, ACK] Seq=180 Ack=670 Win=65024 Len=214
TCP	322	885 → 52441 [PSH, ACK] Seq=670 Ack=394 Win=16768 Len=282
TCP	557	52441 → 885 [PSH, ACK] Seq=394 Ack=952 Win=64512 Len=517
TCP	40	885 → 52441 [ACK] Seq=952 Ack=911 Win=17920 Len=0
TCP	802	885 → 52441 [PSH, ACK] Seq=952 Ack=911 Win=17920 Len=762

Decoded

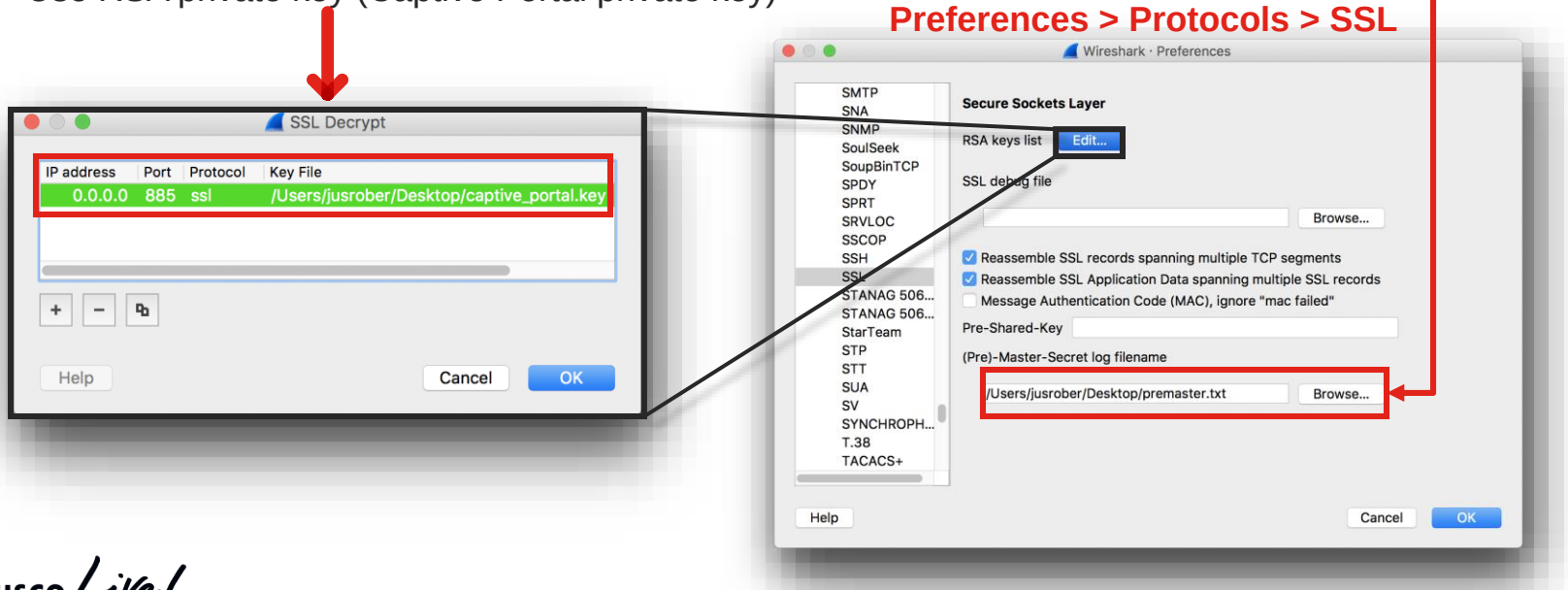
Protocol	Length	Info
TCP	52	52441 → 885 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS
TCP	52	885 → 52441 [SYN, ACK] Seq=0 Ack=1 Win=14600 Len=0
TCP	40	52441 → 885 [ACK] Seq=1 Ack=1 Win=65536 Len=0
TLSv1.2	219	Client Hello
TCP	40	885 → 52441 [ACK] Seq=1 Ack=180 Win=15744 Len=0
TLSv1.2	709	Server Hello, Certificate, Server Hello Done
TLSv1.2	254	Client Key Exchange, Change Cipher Spec, Finished
TLSv1.2	322	New Session Ticket, Change Cipher Spec, Finished
TLSv1.2	557	Application Data
TCP	40	885 → 52441 [ACK] Seq=952 Ack=911 Win=17920 Len=0
TLSv1.2	802	Application Data, Application Data

Decrypting the captures provides even more insight

1. While testing captive portal, have sessions write out key information (Windows):
   - Set environment variable to create a premaster secret file:

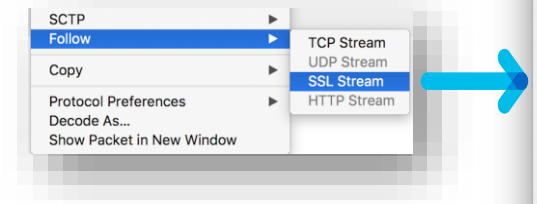
     setx SSLKEYLOGFILE "%HOMEPATH%\Desktop\premaster.txt”
   - Open a private / incognito window and test
2. Use RSA private key (Captive Portal private key)

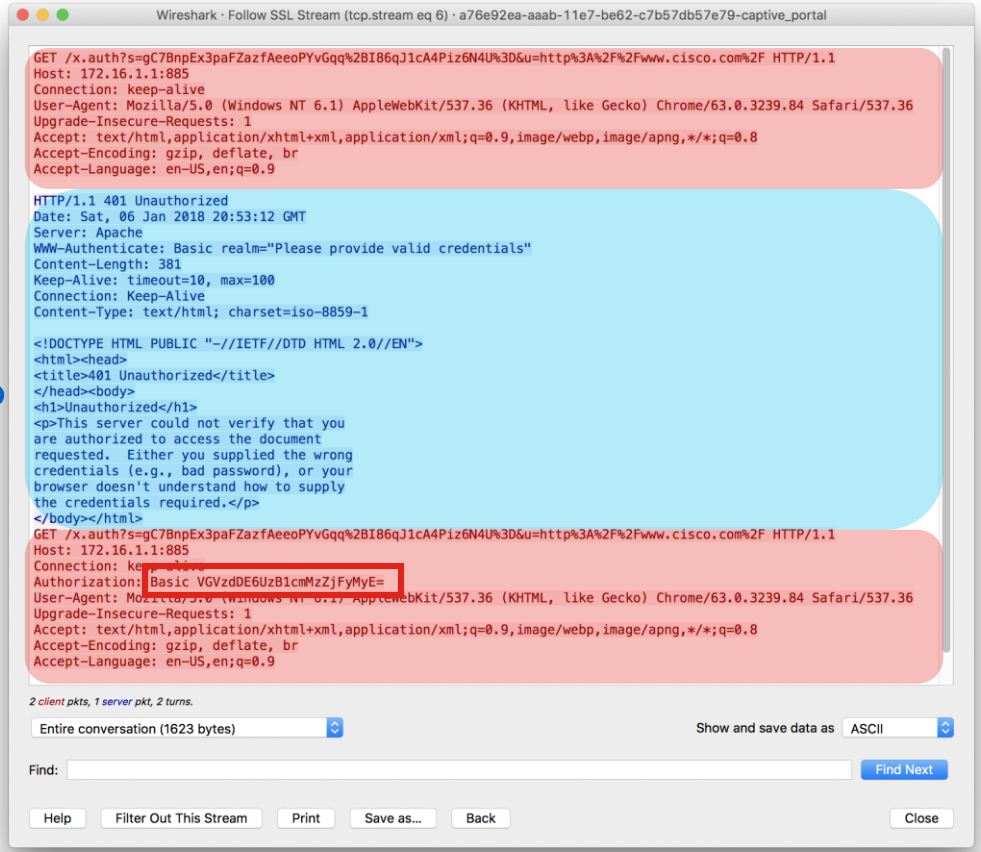
Preferences > Protocols > SSL



You can now follow the SSL Stream

(Right click any SSL Packet)





GET request
after initial redirect

401 Unauthorized
Challenge Response

Captured
Credentials

Redirect back to original destination

```
GET /x.auth?s=gC7BnpEx3paFZazfAeeoPYvGqq%2BI86qJ1cA4Piz6N4U%3D&u=http%3A%2F%2Fwww.cisco.com%2F HTTP/1.1
Host: 172.16.1.1:885
Connection: keep-alive
Authorization: Basic VGVzdDE6UzB1cmMzZjFyMyE=
User-Agent: Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/63.0.3239.84 Safari/537.36
Upgrade-Insecure-Requests: 1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9

HTTP/1.1 307 Temporary Redirect
Date: Sat, 06 Jan 2018 20:53:22 GMT
Server: Apache
Location: http://www.cisco.com/
Content-Length: 231
Keep-Alive: timeout=10, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>307 Temporary Redirect</title>
</head><body>
<h1>Temporary Redirect</h1>
<p>The document has moved <a href="http://www.cisco.com/">here</a>.</p>
</body></html>
```

Original Destination

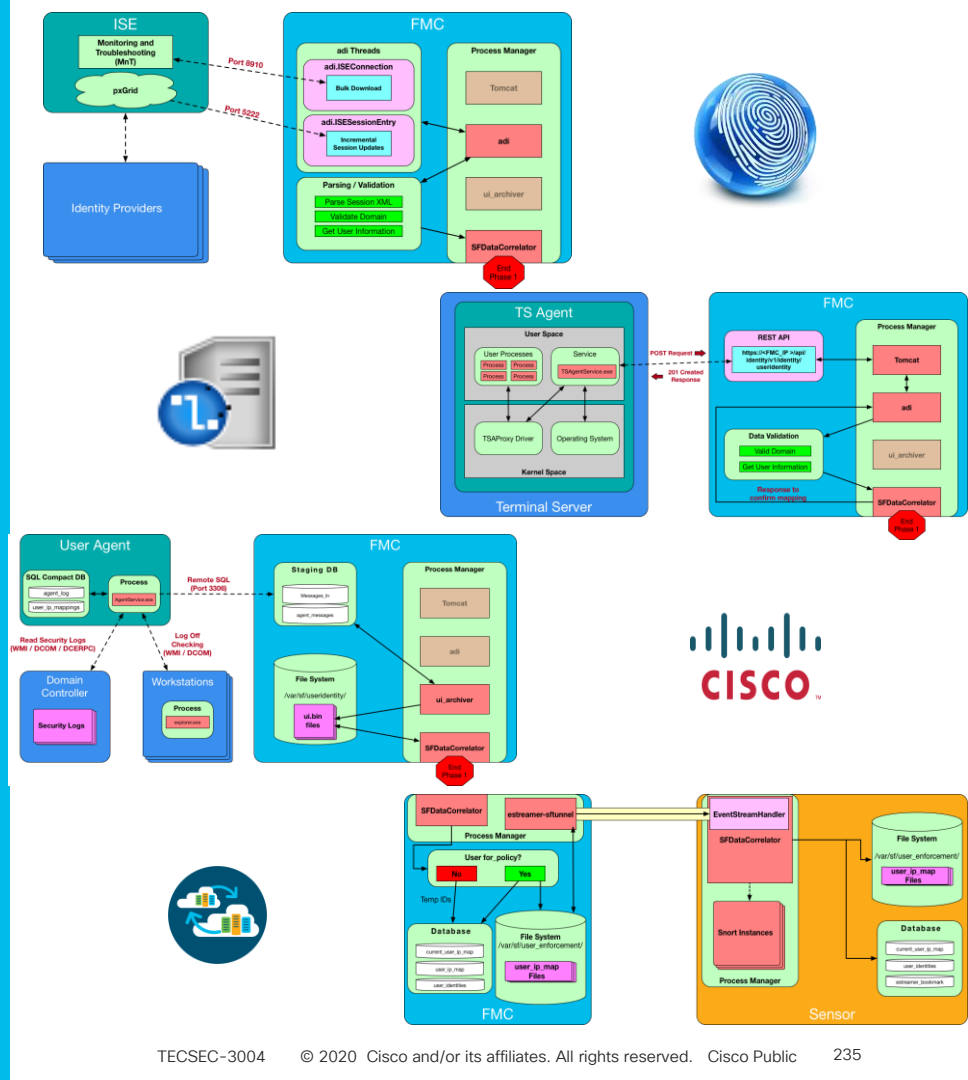
Want more on Identity?

Check the Cisco Live

On-Demand Catalog for:

BRKSEC-3227

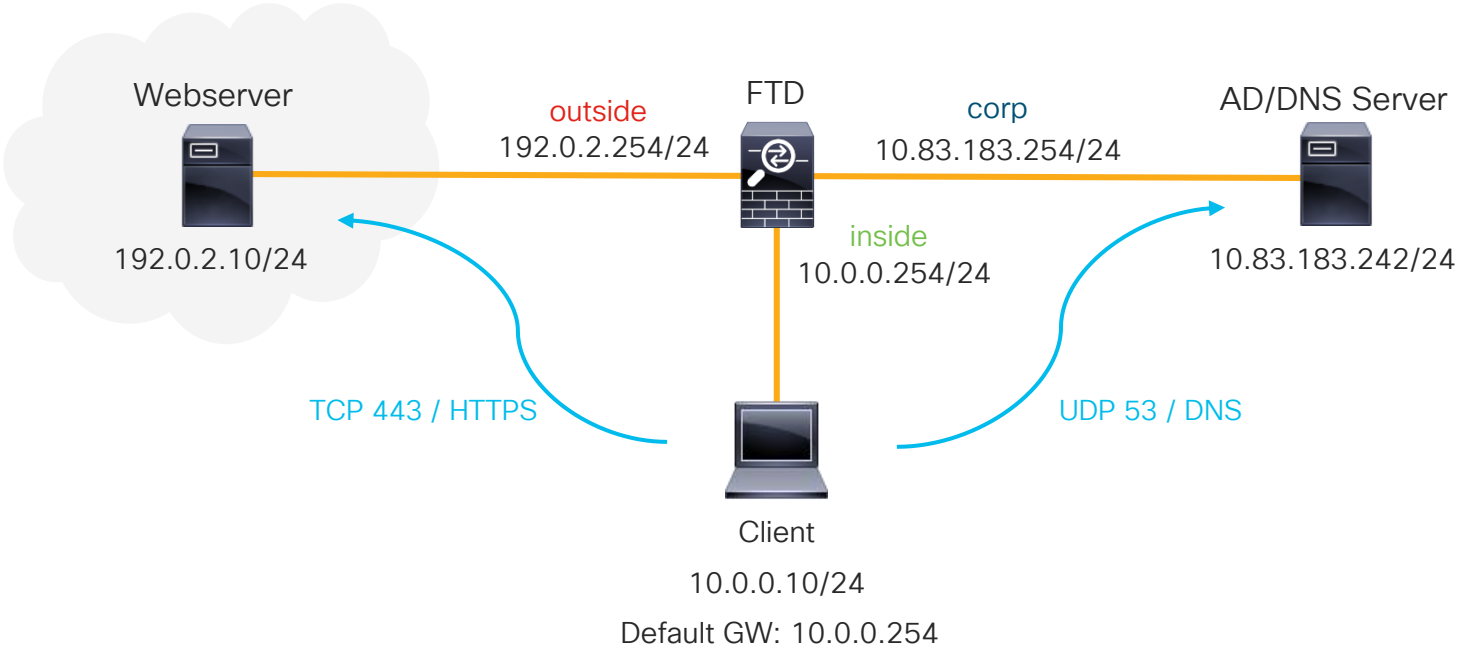
Integrating & Troubleshooting Identity Features on the Firepower System



Interactive Troubleshooting

Scenario Topology

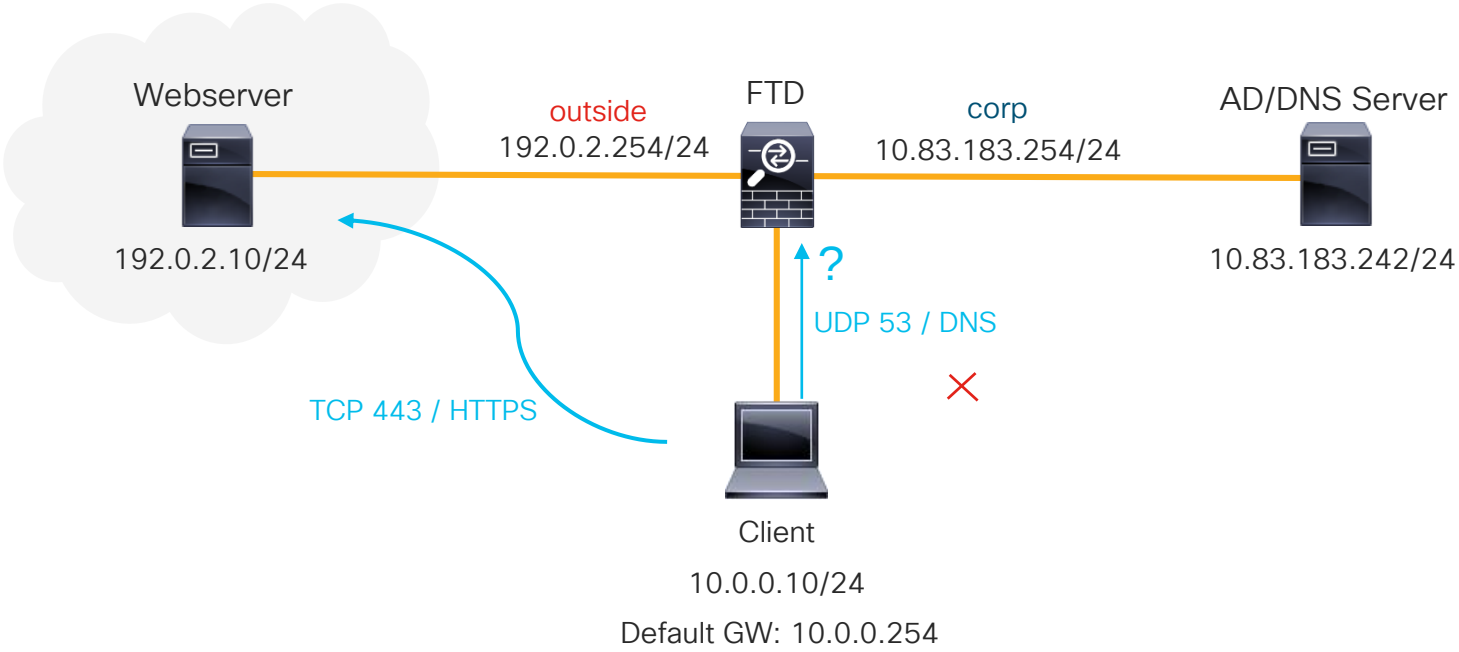
- Goal: Client to retrieve a file from an external webserver via HTTPS through FTD



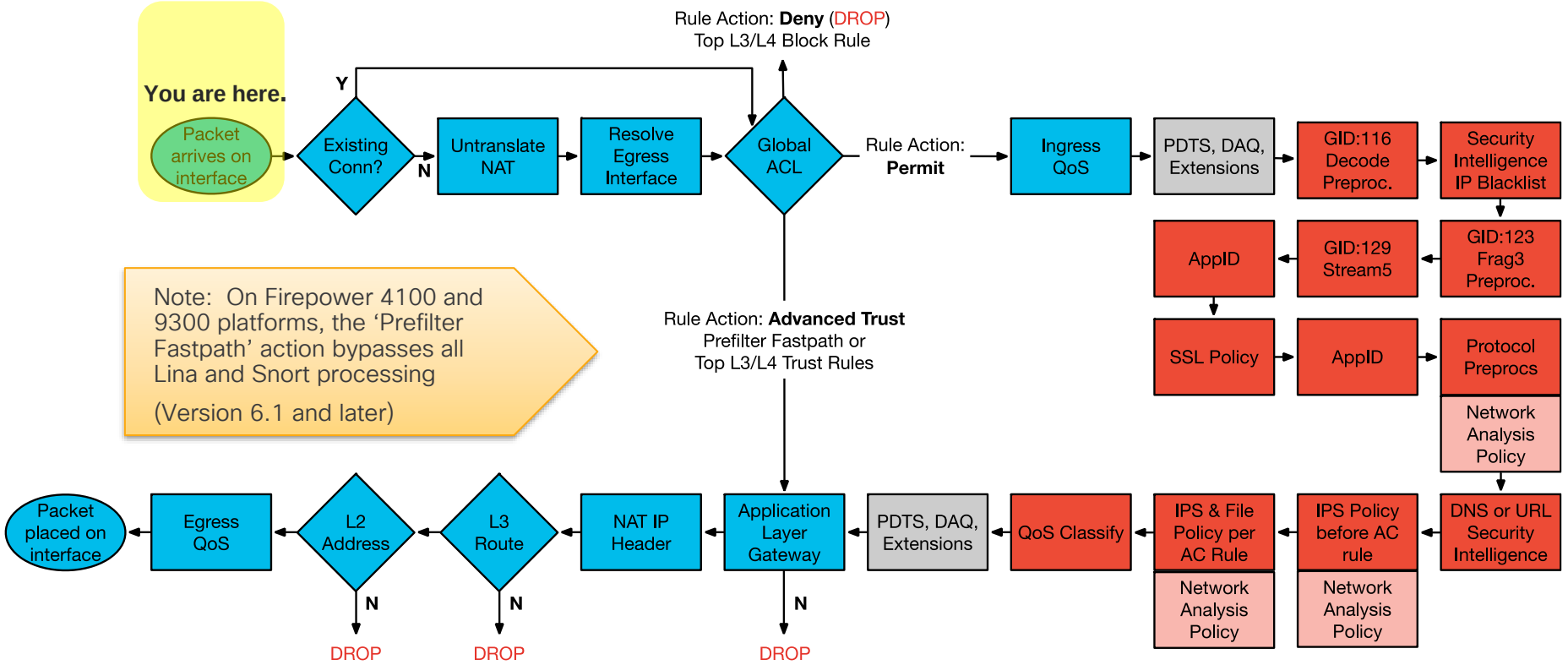
Stage 1

Stage 1: DNS resolution failure (client ARP)

- Client cannot reach the DNS server because of a bad static ARP entry for its default gateway



Reference Slide: Routed FTD Path of Packet



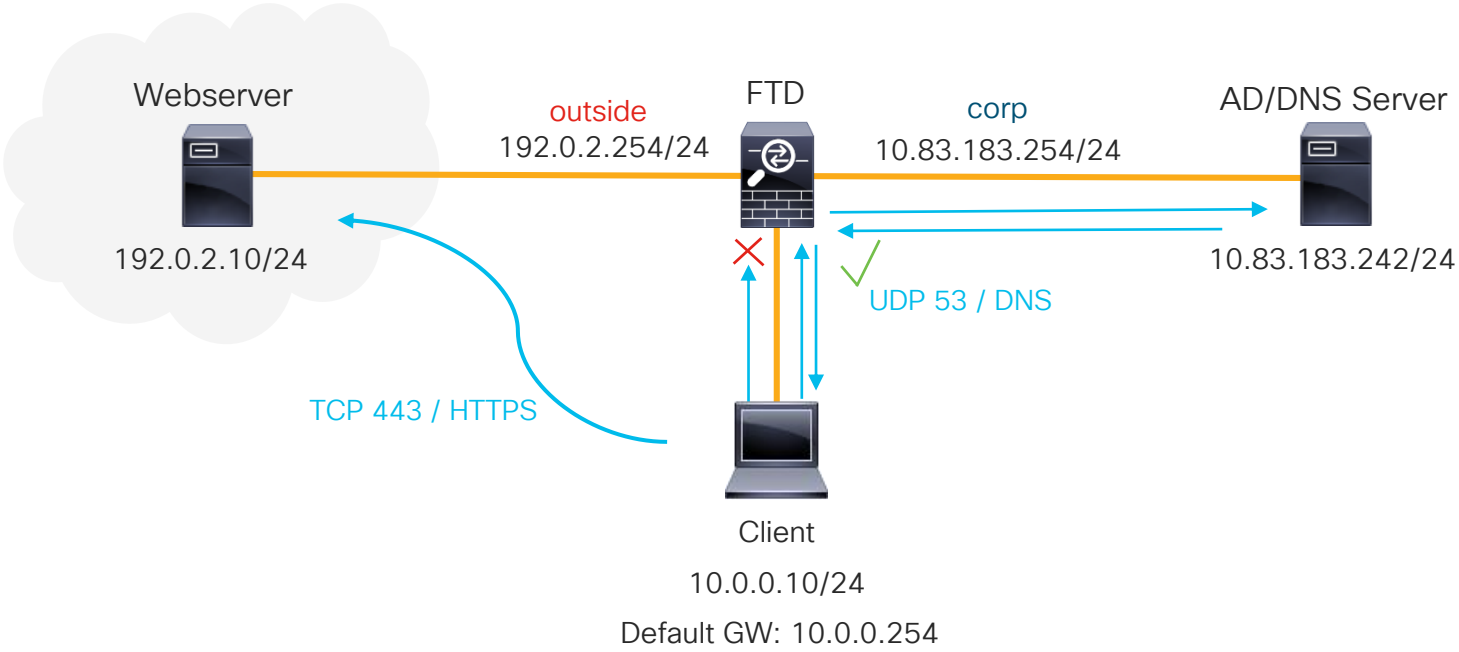
LINA ASA Engine = BLUE

Snort Engine = RED

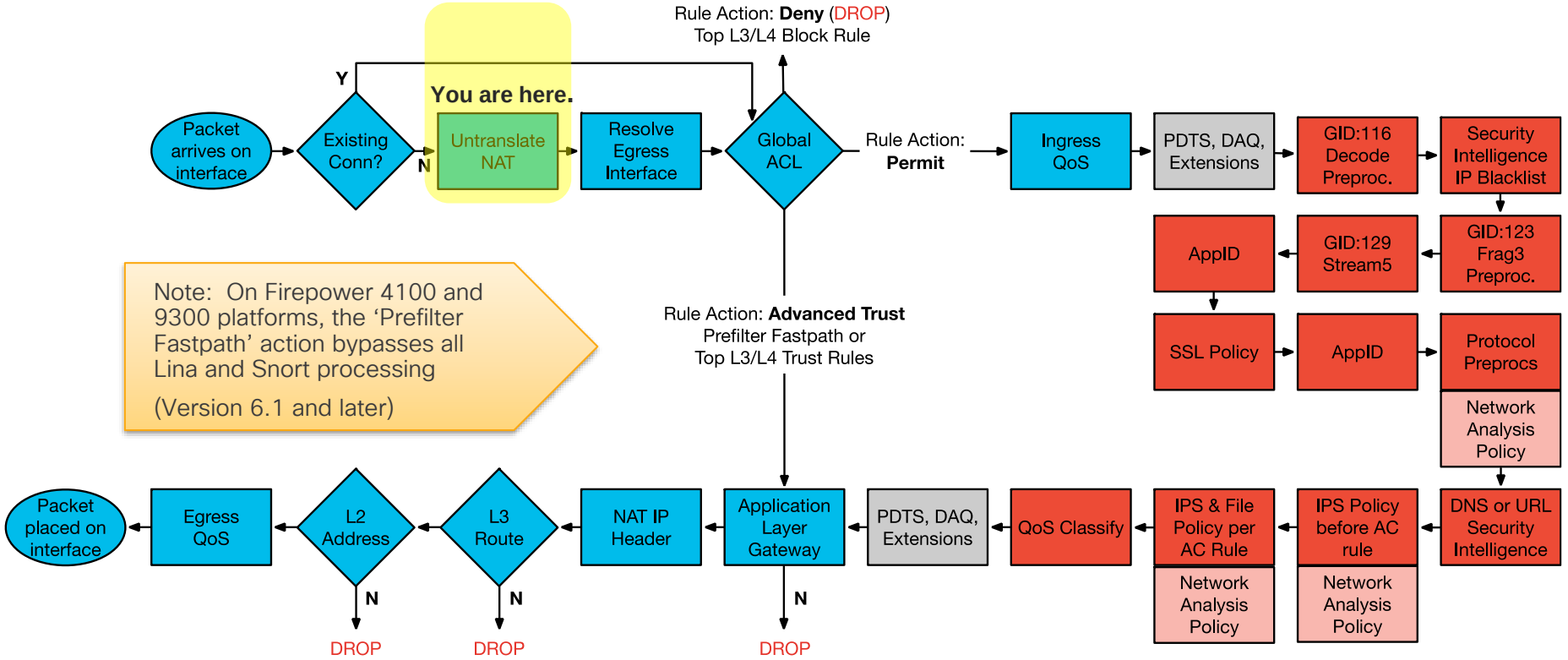
Stage 2

Stage 2: Connection Fails (Bad Static NAT)

- A static NAT rule was configured to send traffic out of the wrong interface (corp)



Reference Slide: Routed FTD Path of Packet



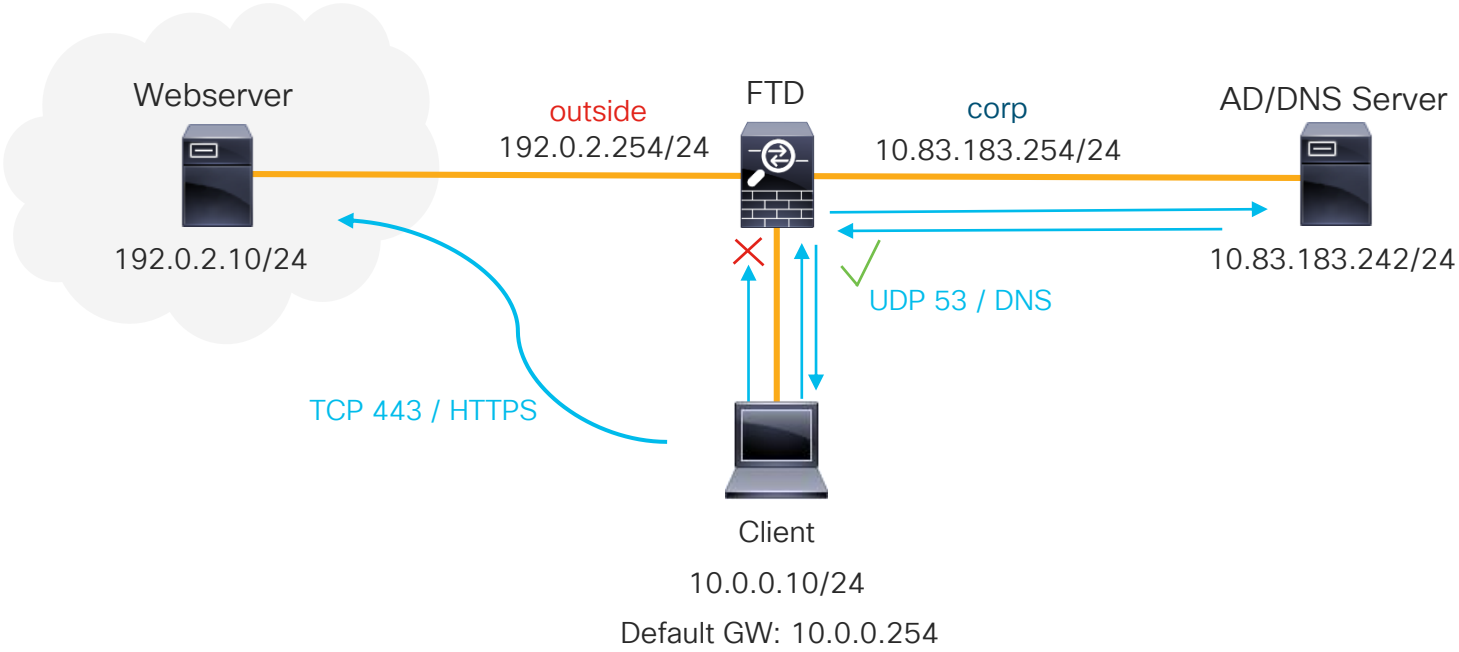
LINA ASA Engine = BLUE

Snort Engine = RED

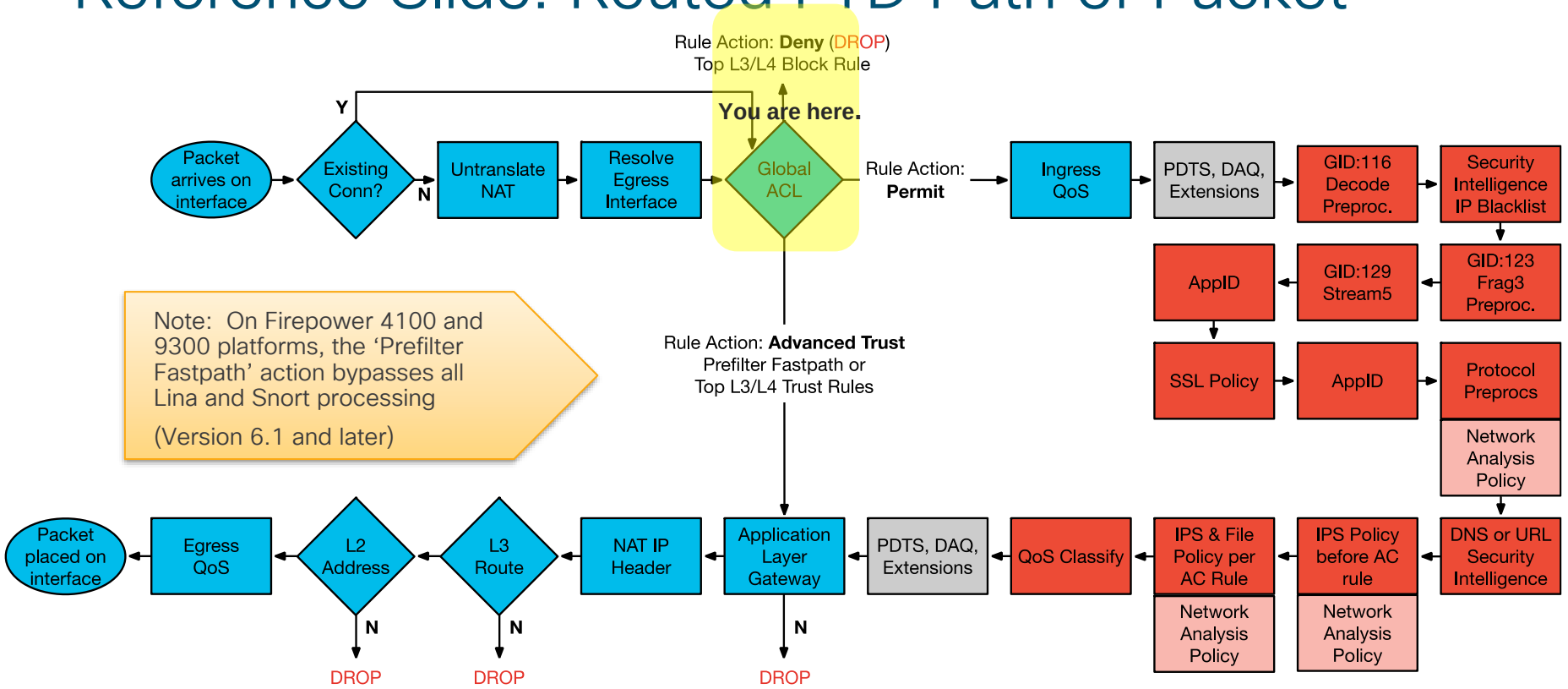
Stage 3

Stage 3: Packet dropped by FTD (ACL Block)

- A pre-filter rule was configured to block all traffic from the Client to the Webserver



Reference Slide: Routed FTD Path of Packet



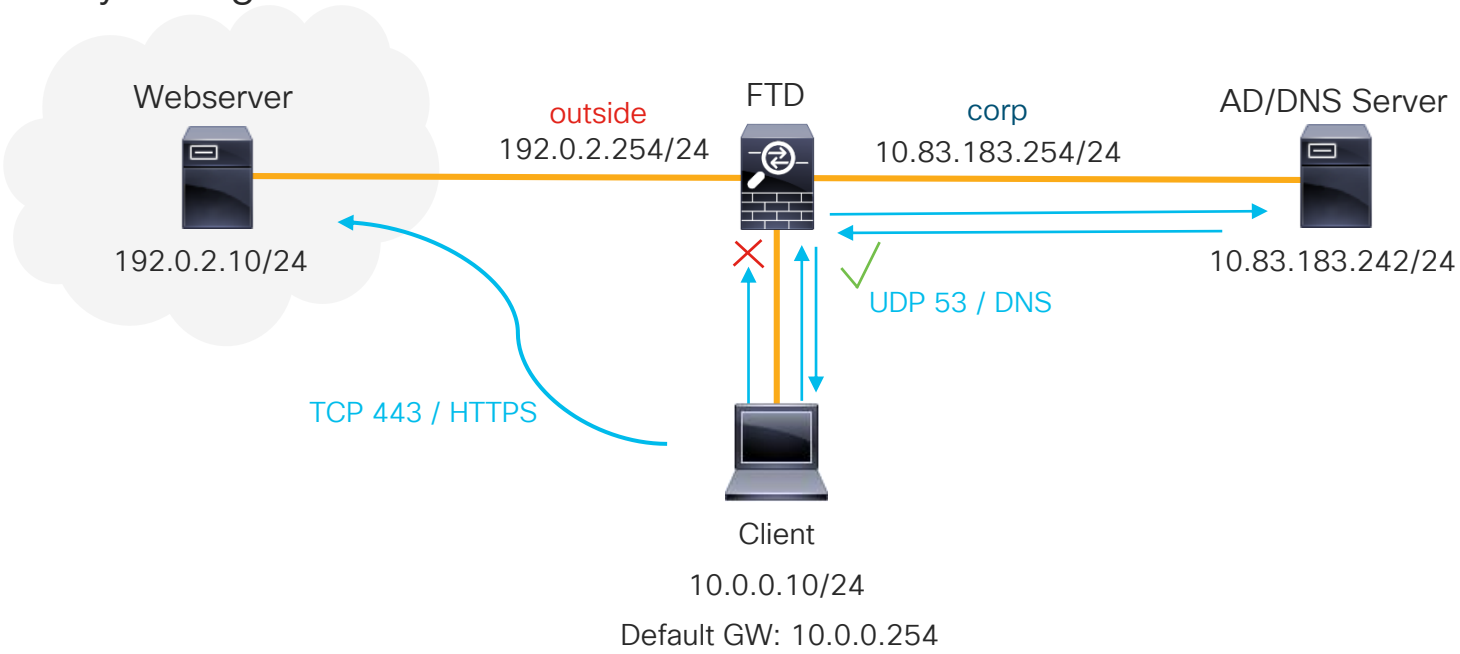
LINA ASA Engine = BLUE

Snort Engine = RED

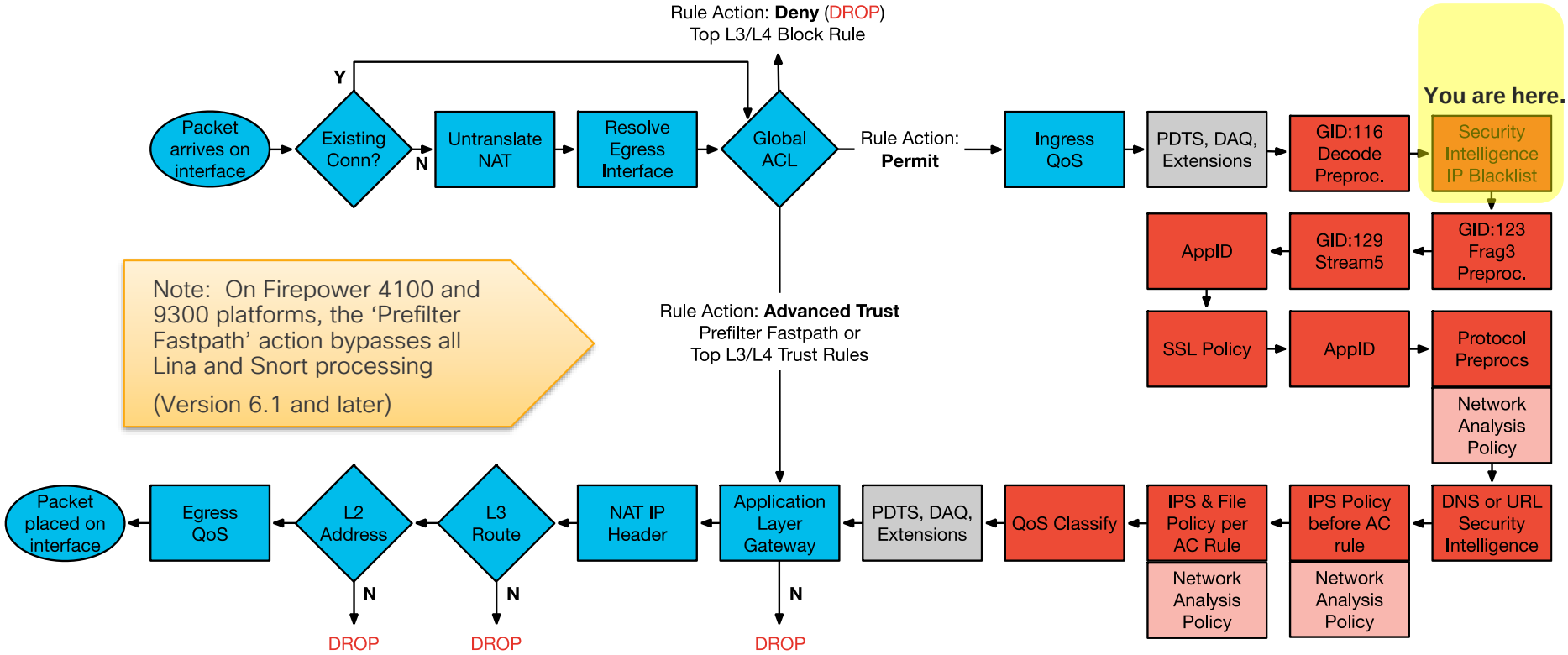
Stage 4

Stage 4: Packet dropped by Snort (SI Block)

- The Webservers IP address (192.0.2.10) was included in the custom blacklist for security intelligence



Reference Slide: Routed FTD Path of Packet



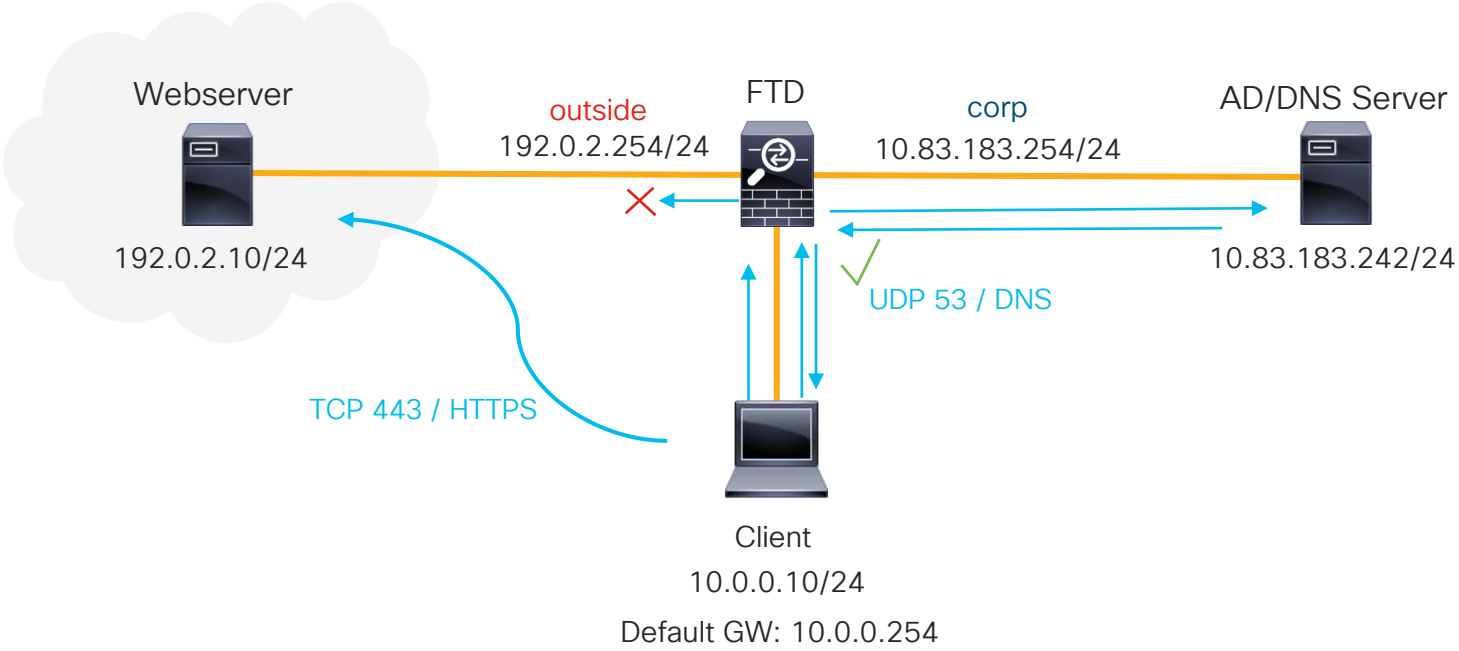
LINA ASA Engine = BLUE

Snort Engine = RED

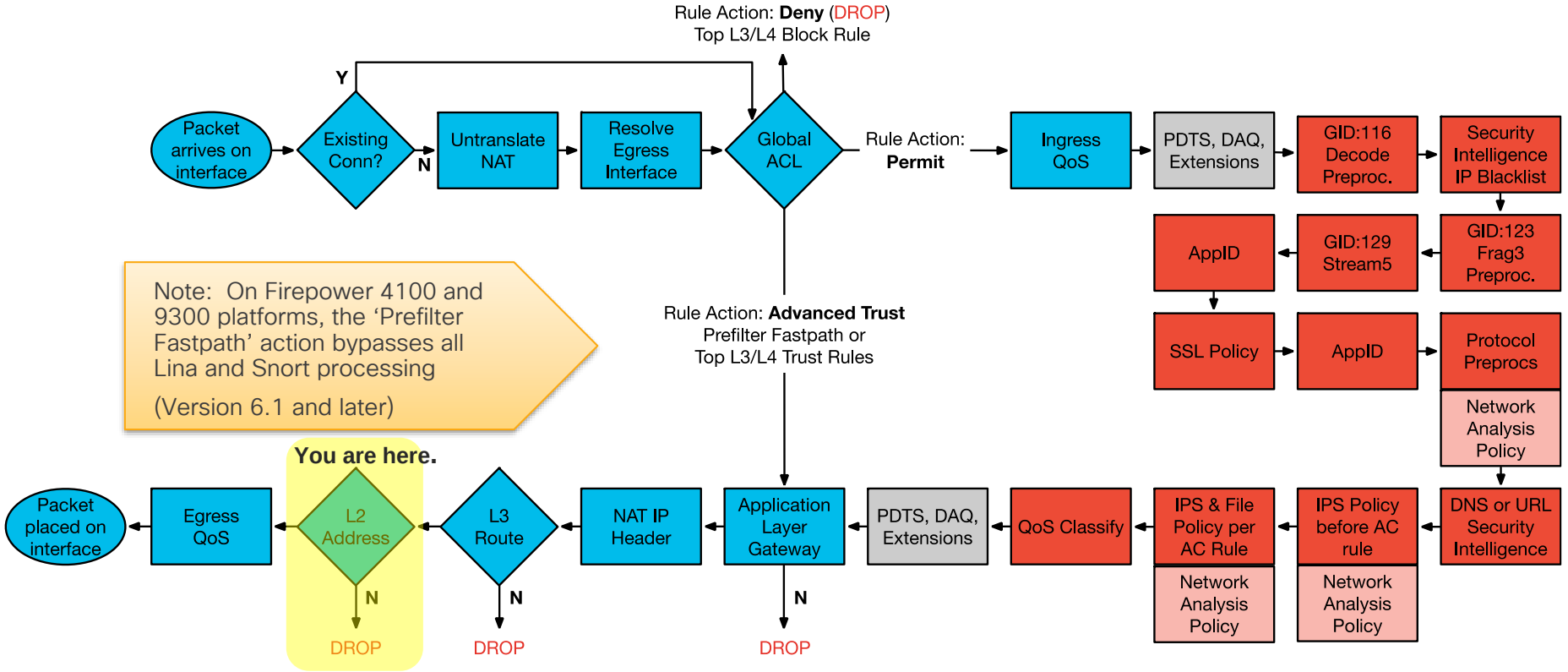
Stage 5

Stage 5: Packet doesn't egress (No next hop ARP)

- FTD has a static route for 192.0.2.10 with a next hop that does not exist. This results in an L2 Adjacency failure and the packet does not egress on the outside



Reference Slide: Routed FTD Path of Packet



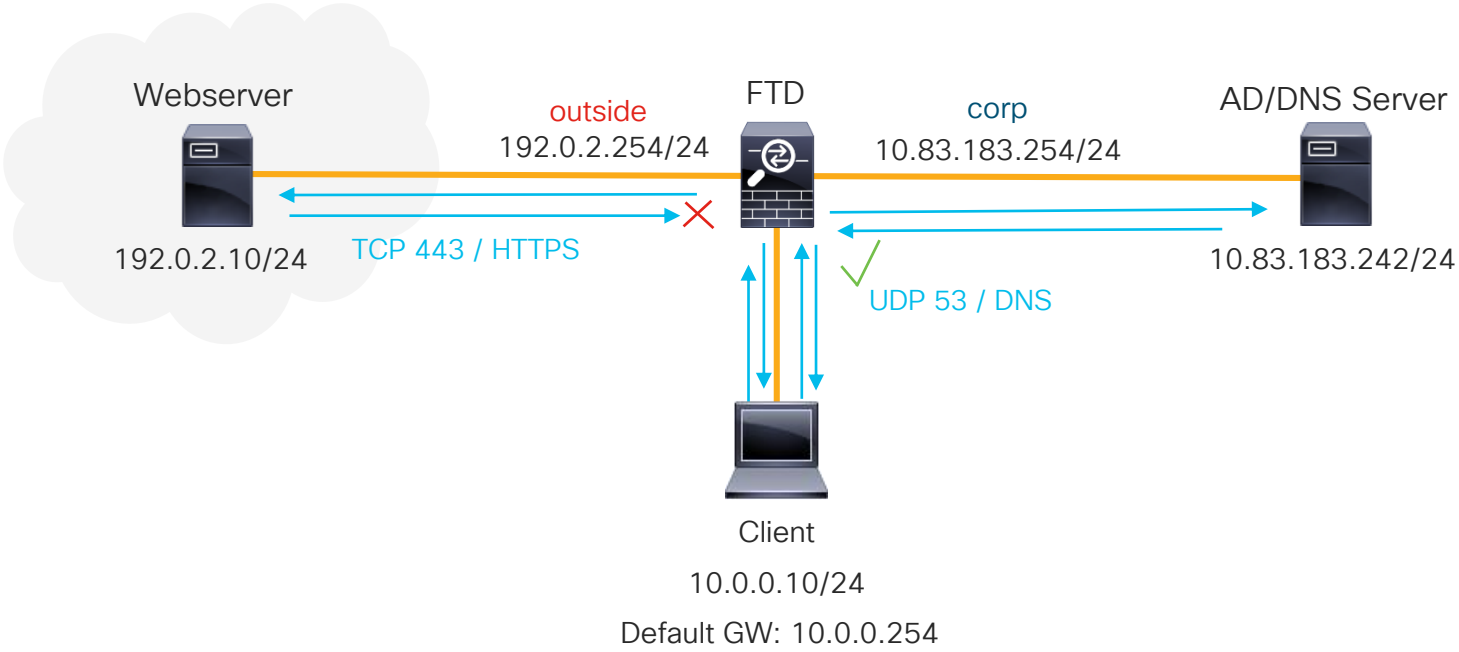
LINA ASA Engine = BLUE

Snort Engine = RED

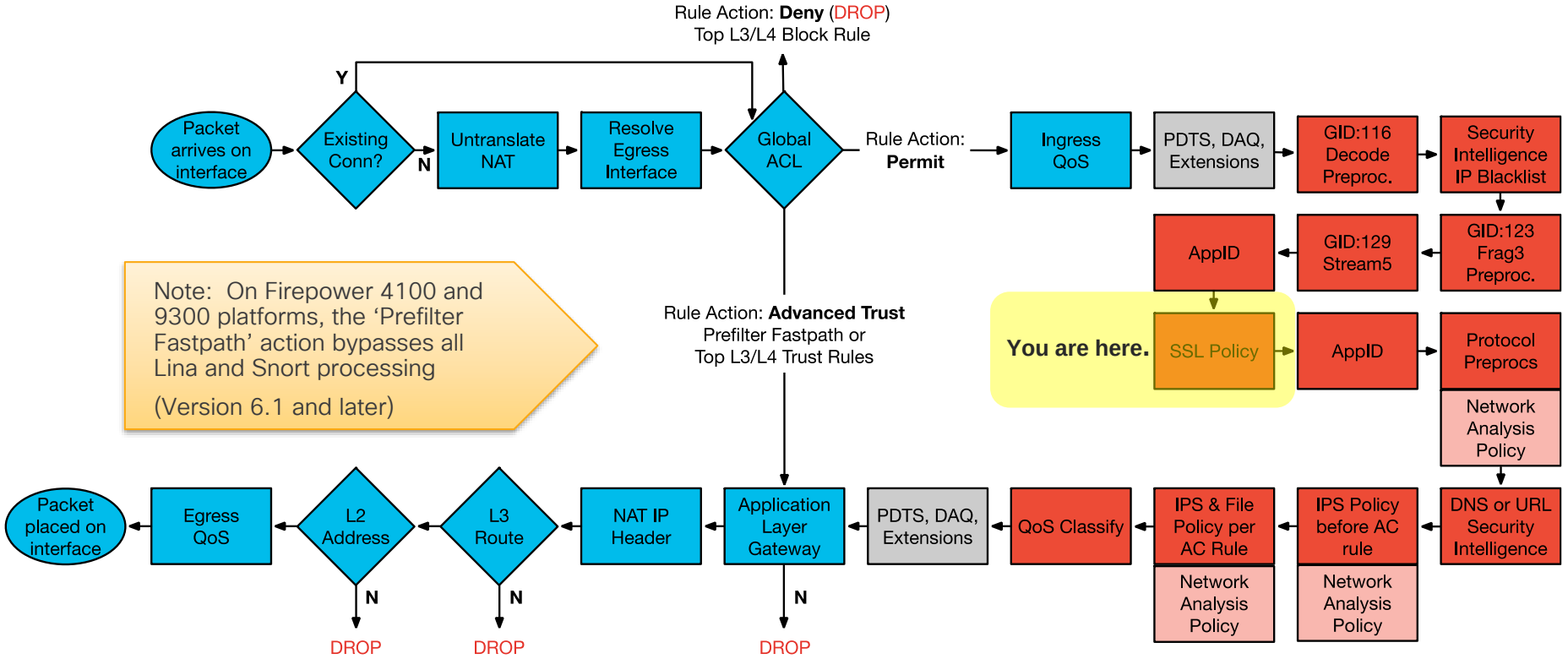
Stage 6

Stage 6: TLS connection reset (SSL Block)

- TLS connection to Webserver fails because of a “Block w/ reset” rule in the SSL Policy set to match on the CN of the servers certificate



Reference Slide: Routed FTD Path of Packet



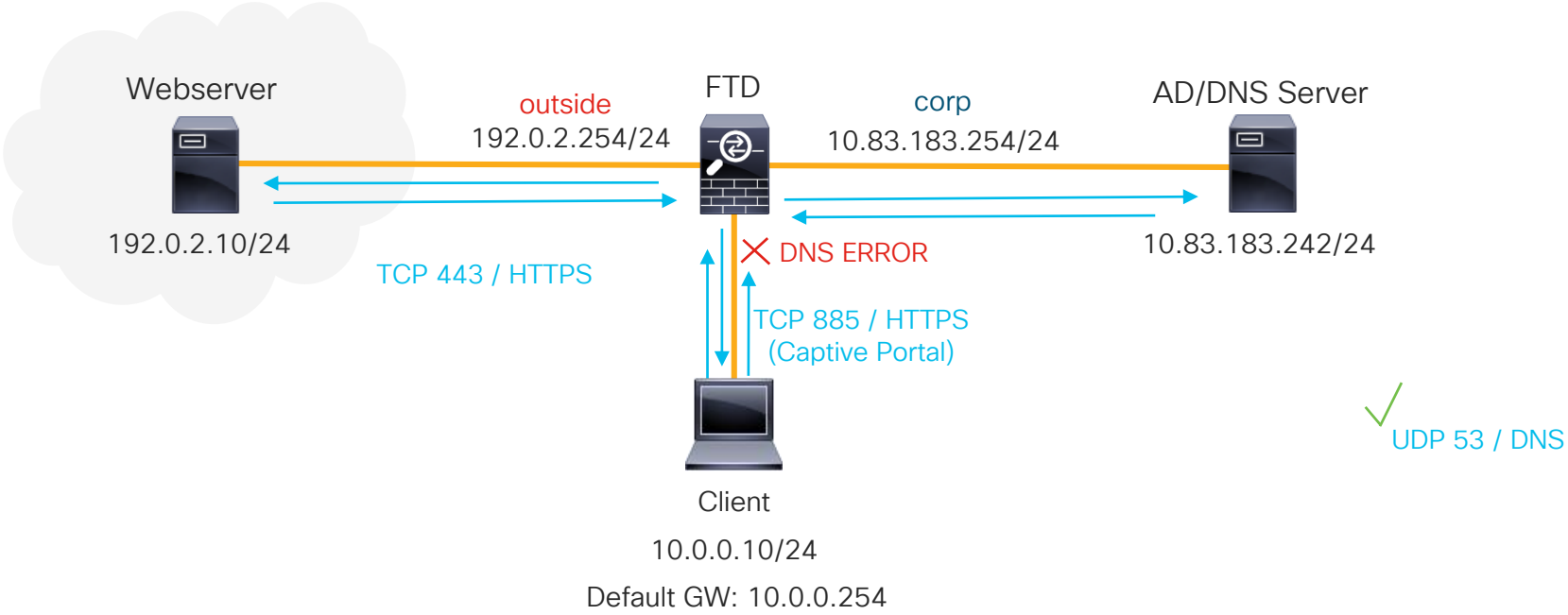
LINA ASA Engine = BLUE

Snort Engine = RED

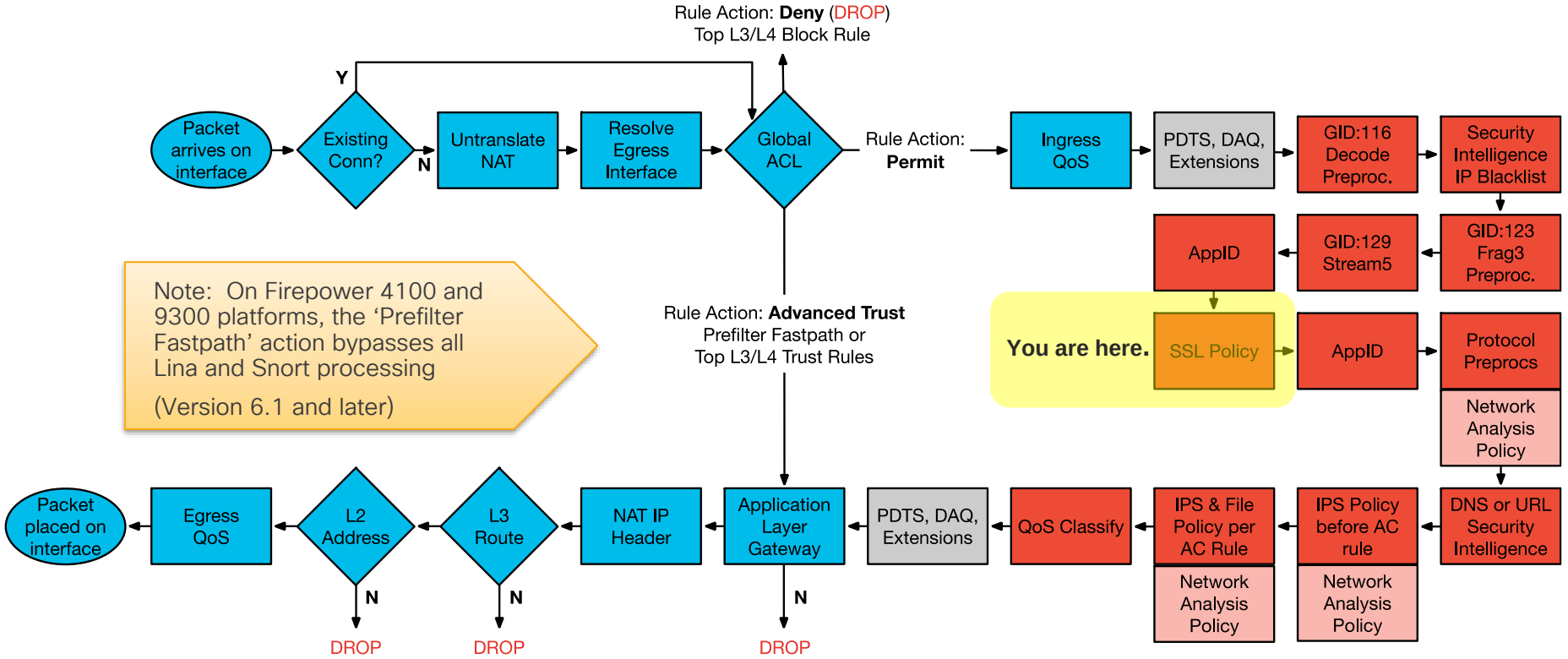
Stage 7

Stage 7: Captive Portal redirect (DNS failure)

- Captive portal intercepts the connection and redirects the user to its hostname. This redirect fails on name resolution because there is no A-record in the DNS server for this host



Reference Slide: Routed FTD Path of Packet



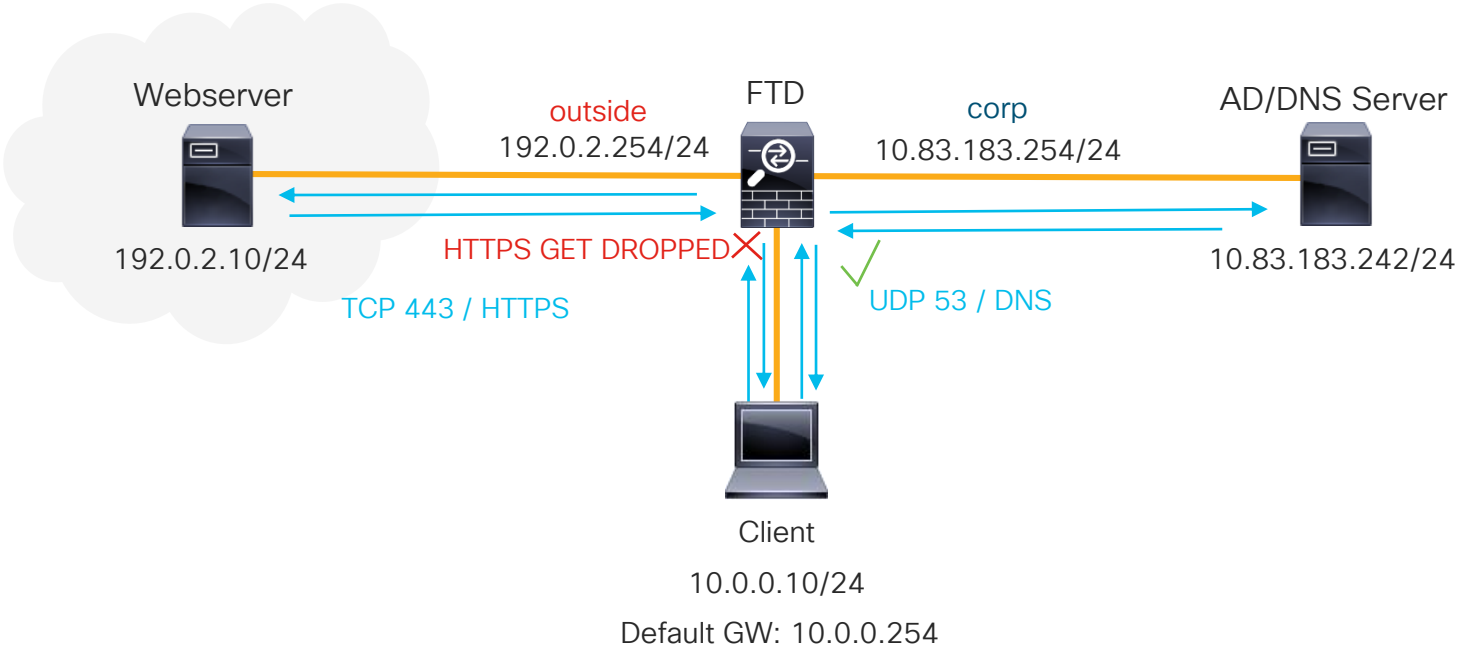
LINA ASA Engine = BLUE

Snort Engine = RED

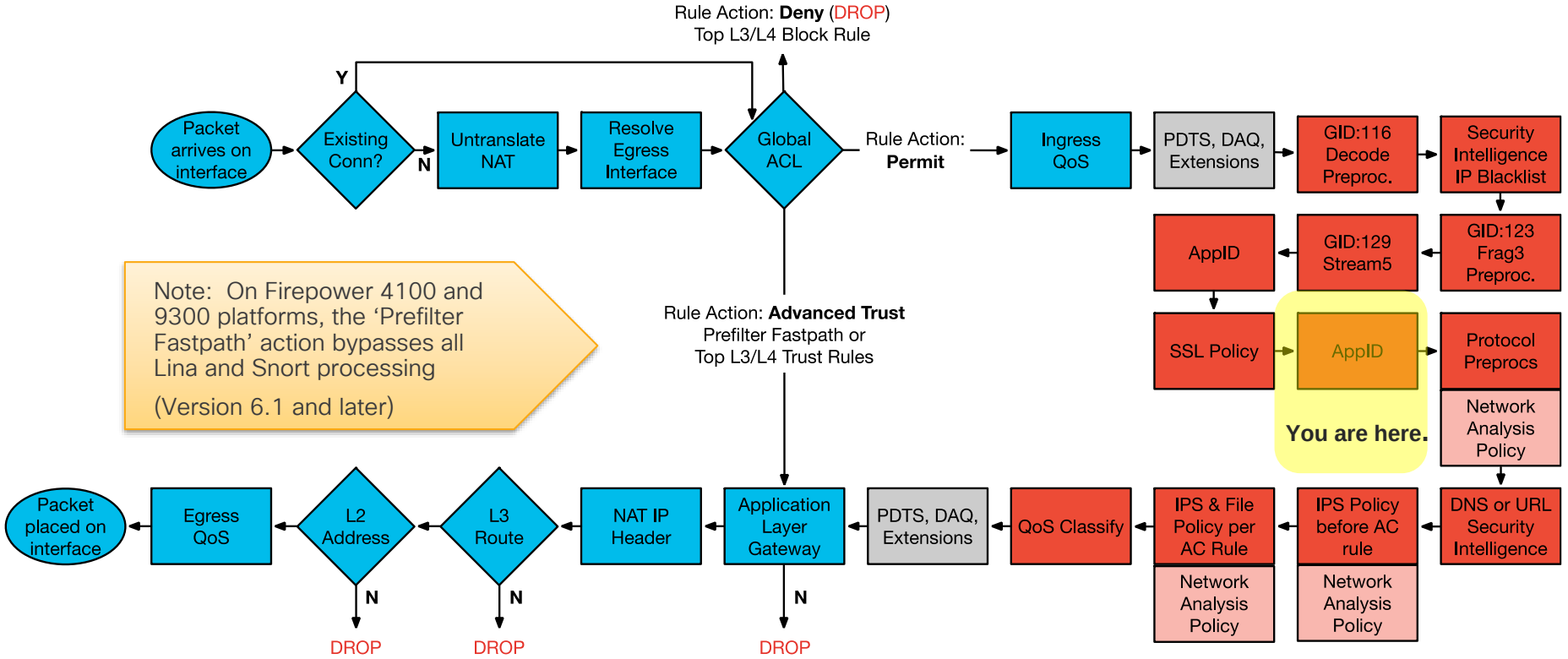
Stage 8

Stage 8: HTTP GET is dropped (AppID Block Rule)

- Use 'system support trace' w/ firewall-engine-debug enabled



Reference Slide: Routed FTD Path of Packet



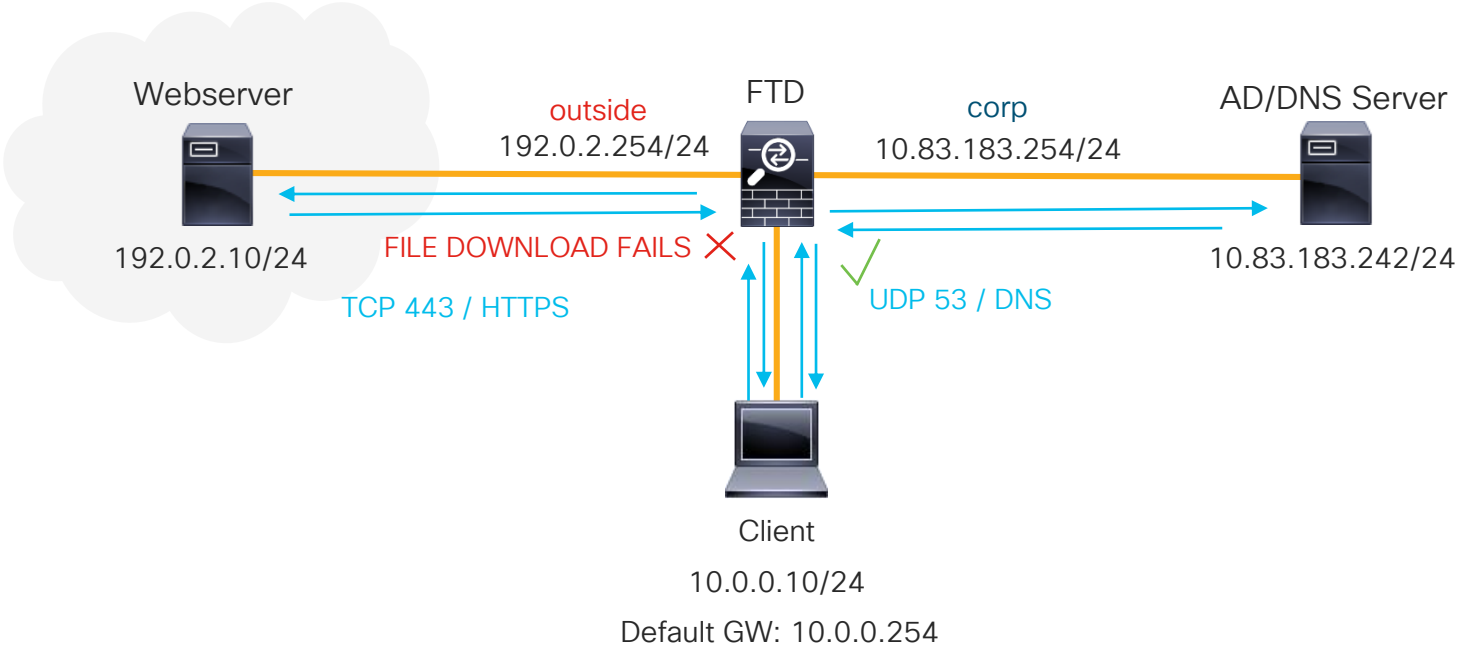
LINA ASA Engine = BLUE

Snort Engine = RED

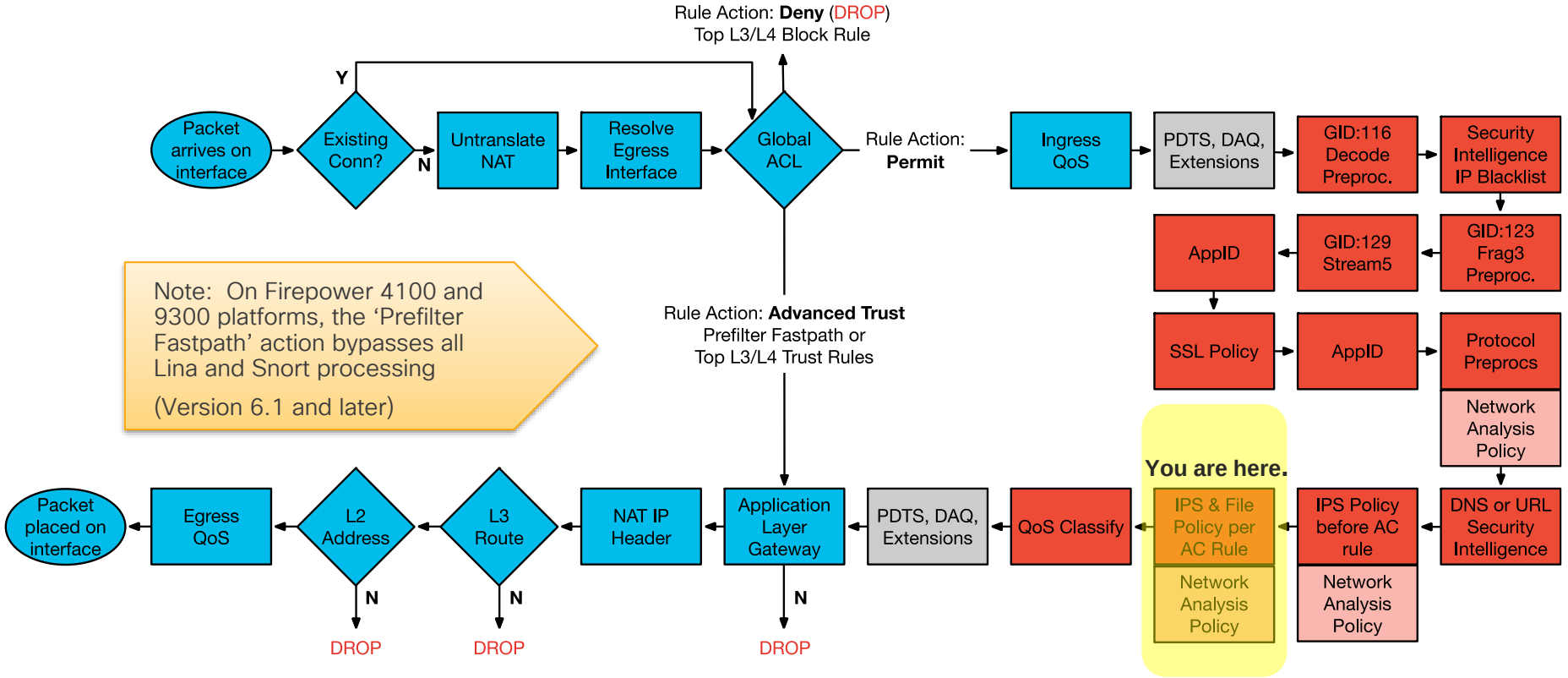
Stage 9

Stage 9: File Download blocked (Snort rule)

- A local rule was enabled in the Intrusion Policy to “Drop and Generate events” that matched the URI of the download request for “my_important_doc.pdf”



Reference Slide: Routed FTD Path of Packet



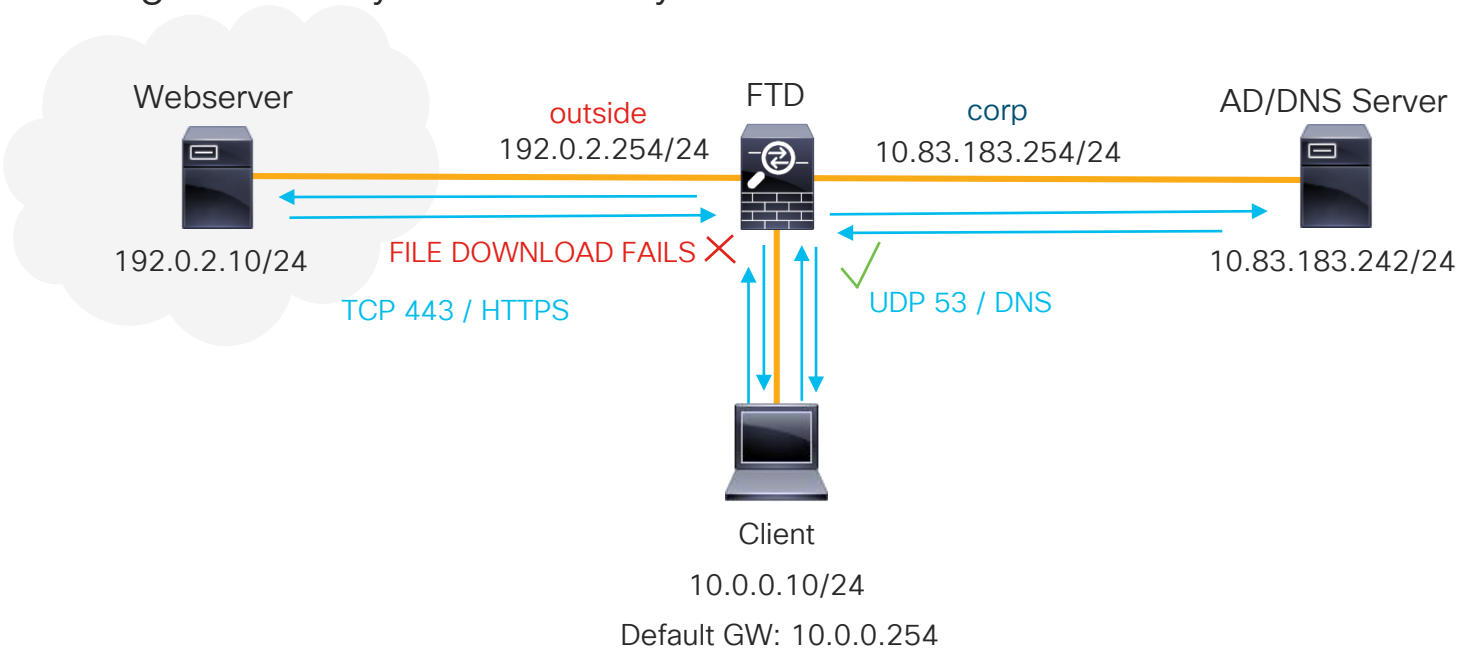
LINA ASA Engine = BLUE

Snort Engine = RED

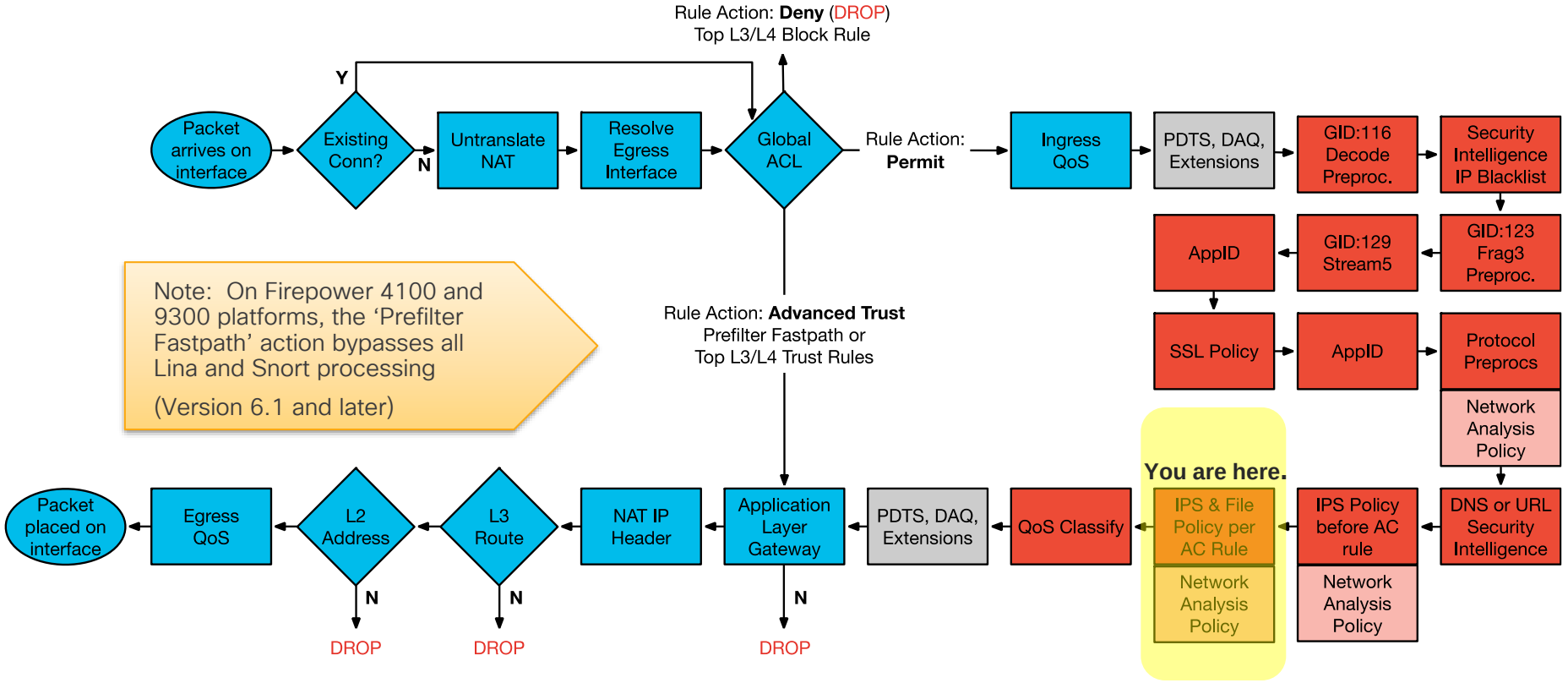
Stage 10

Stage 10: File Download blocked (File Policy)

- The hash of ”my_important_doc.pdf” was present in the custom detection file list and was being blocked by the File Policy



Reference Slide: Routed FTD Path of Packet



Questions & Answers

TAC Security Podcast

Knowledge from TAC… On the go!

- Periodic podcast episodes with troubleshooting tips from TAC
- Focus on new Cisco Security technology trends like Firepower Threat Defense, ISE 2.1, Anyconnect, voice security, etc…and CCIE study tips!

50+ Episodes already available!

Cisco Technical Support App

FOLLOW US ON twitter

@CiscoTACPodcast

cisco Live!

Latest Podcast Episodes

Ep. #	Topic
55	Firepower 6.4 and Other Ramblings
54	A Discussion on Cisco Encrypted Traffic Analytics (ETA) with the Experts
53	Thoughts on Security at Cisco Live US 2018 in Orlando
52	ASA/FTD Troubleshooting Enhancements and Cisco Live US 2018
51	EasyConnect in ISE 2.1
50	Finding Your Firepower – A discussion on Firepower Technologies
49	Cryptic Thoughts – A discussion on changing crypto standards
48	Cisco Live! 2015 – San Diego
47	The Cisco Security Ninja Program
46	New features in ASA version 9.3(2)
45	Introduction to Cisco Wireless Security
44	Cisco Live! 2014 in San Francisco
43	ASA Version 9.2 and Interesting TAC Cases
42	The Cisco Secure Development Lifecycle

Security Beta Programs

Beta Software Access

Product Training

Access to Dev Teams

Test Hardware and Licenses

Bugs Fixed for Release

Influence Product Roadmap

Presented By Cisco Security Customer Insights

ASA | NGFW | NGIPS | Firepower Platforms | AMP | CTA | ESA | WSA | ISE | Umbrella

Enroll today!

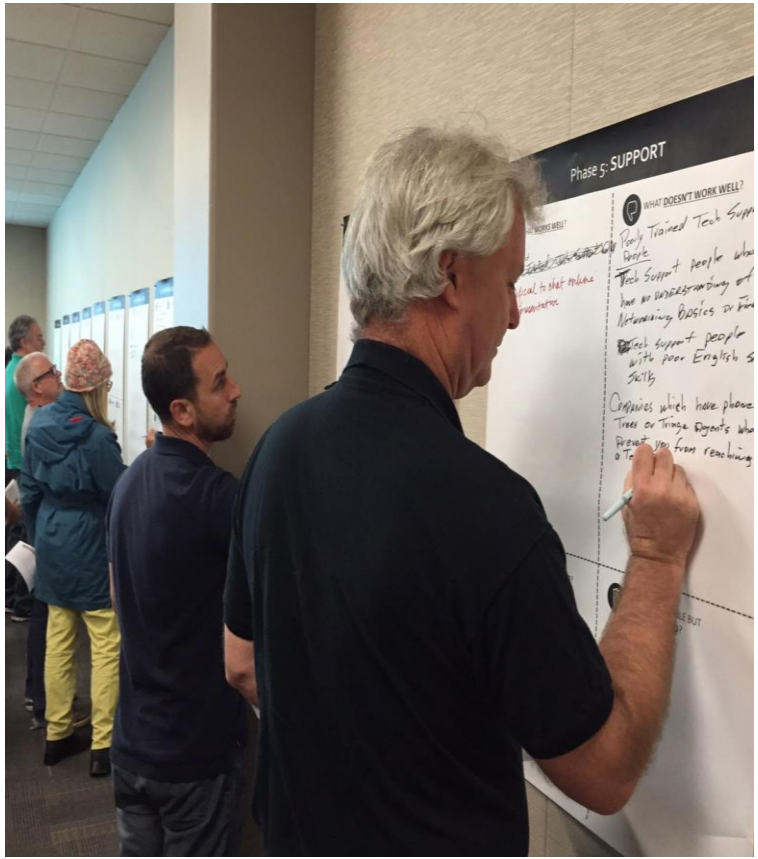
http://cs.co/security-beta-nomination

ask-sbg-beta@cisco.com

"I've been involved in many beta programs … I must say that this one has been the best organized. This beta has taken a very active, hands-on approach." - **Liberal Arts College Customer**

SBG's **User Experience (UX) team** is running collaborative Design Thinking Sessions at Cisco Live!

Your ideas →
Sharpies + Inner Picasso
→
Product Improvements!



Do you:

- use our **NextGen FireWall** product(s)?
- wonder who you can bring your experience **pain points** to?
- have **ideas** that keep you up at night?
- want to improve product experience for yourself?

Come talk to **Security User Experience (UX)** Team!!

cisco Live!

1

Come join our Design Thinking session on Tuesday or Thursday! Signup using QR code 1 (above).

2

Don't have time at Cisco Live? Join our UX participant database and we'll be in touch to showcase upcoming features and get your feedback! Signup using QR code 2.

Wrapping it up

- Apply new skills to your daily FTD troubleshooting.
- Check out the additional resources and slides for future reference purposes.
- Although FTD is complex, you should now have a better understanding of the product architecture, traffic flow, and troubleshooting tools that are available to help you quickly resolve issues.
- If you leverage those newfound skills and resources, before you know it you'll be troubleshooting FTD like a TAC engineer!

Complete your online session survey

- Please complete your session survey after each session. Your feedback is very important.
- Complete a minimum of 4 session surveys and the Overall Conference survey (starting on Thursday) to receive your Cisco Live t-shirt.
- All surveys can be taken in the Cisco Events Mobile App or by logging in to the Content Catalog on ciscolive.com/emea.

Cisco Live sessions will be available for viewing on demand after the event at ciscolive.com.

cisco Live!

Continue your education

Demos in the Cisco Showcase

Walk-In Labs

Meet the Engineer 1:1 meetings

Related sessions

Thank you

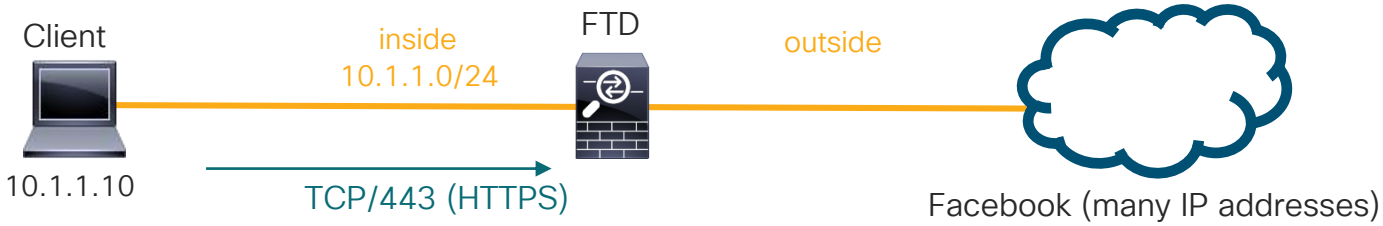
cisco Live!

You make **possible**

Appendix

Troubleshooting Walkthroughs

Scenario 1: Facebook is not blocked as expected and CNN is unexpectedly being blocked



The customer on 10.1.1.10 is able to access Facebook.com, whereas this client should be blocked from all Social Networking sites.

The customer's Access Control Policy is many pages long!

Let's troubleshoot this using a systematic approach to FTD troubleshooting

Using our FTD troubleshooting tools

Remember: Always check events and syslogs! FMC: Analysis → Connections → Events

No connection events for 10.1.1.10 navigating to Facebook. We must not be logging the rule which allows it.

							SMTP						
7	Block Auction URL	Any	Any	Any	Any	Any	Any	Any	Any	Any	Auctions (Any Re	Any	Block
8	Block Games URL	Any	Any	Any	Any	Any	Any	Any	Any	Any	Games (Any Repu	Any	Block
9	Block Hacking URL	Any	Any	Any	Any	Any	Any	Any	Any	Any	Hacking (Any Rep	Any	Block
10	Block Job Search URL	Any	Any	Any	Any	Any	Any	Any	Any	Any	Job Search (Any	Any	Block
11	Block Malware URL	Any	Any	Any	Any	Any	Any	Any	Any	Any	Malware Sites (A	Any	Block
12	Block Parked Domains URL	Any	Any	Any	Any	Any	Any	Any	Any	Any	Parked Domains	Any	Block
13	Block Social Networking URL	Any	Any	Any	Any	Any	Any	Any	Any	Any	Social Network (A	Any	Block

The rule we expect traffic to hit

Firewall engine debug

At this point, we suspect there is a problem with rule evaluation.

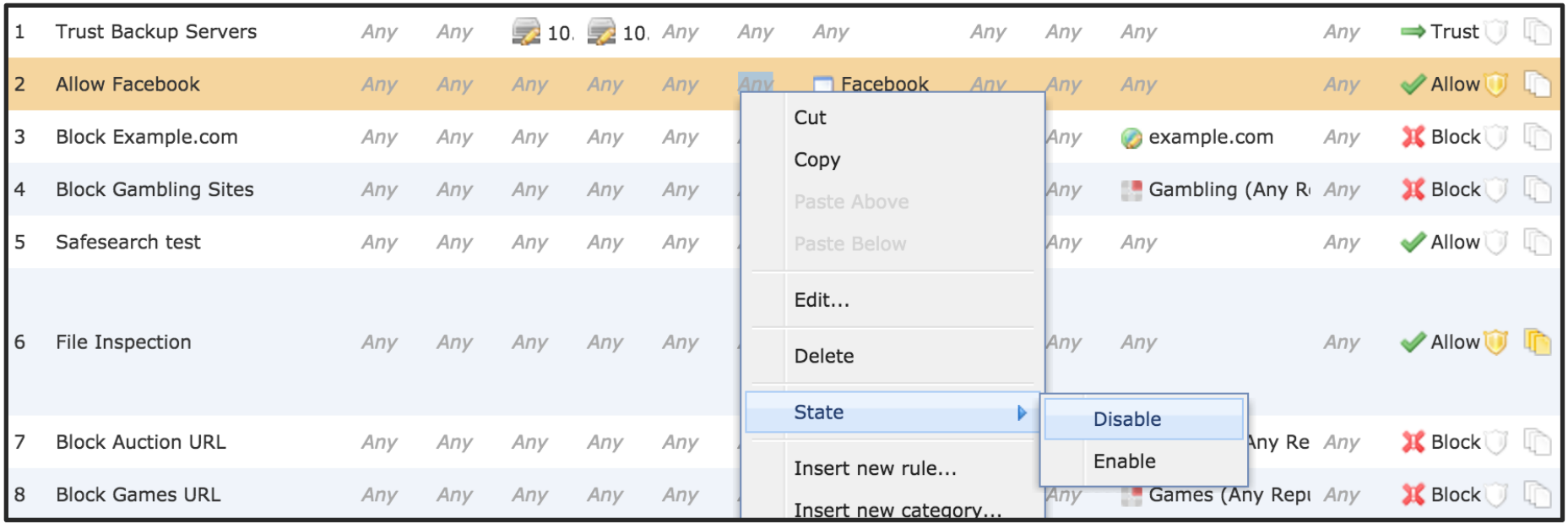
Firewall Engine Debug is the right tool to identify what is happening within the Access Control Policy

```
> system support firewall-engine-debug

Please specify an IP protocol: tcp
Please specify a client IP address: 192.168.1.10
Please specify a client port:
Please specify a server IP address:
Please specify a server port: 443
Monitoring firewall engine debug messages

192.168.1.10-49986 > 31.13.69.228-443 6 AS 1 I 1 New session

192.168.1.10-49986 > 31.13.69.228-443 6 AS 1 I 1 Starting with minimum 2, 'Allow Facebook', and SrcZone
first with zones 4 -> 3, geo 0(0) -> 0, vlan 0, inline sgt tag: untagged, ISE sgt id: 0, svc 1122,
payload 629, client 1296, misc 0, user 9999997, url facebook.com, xff
192.168.1.10-49986 > 31.13.69.228-443 6 AS 1 I 1 match rule order 2, 'Allow Facebook', action Allow
192.168.1.10-49986 > 31.13.69.228-443 6 AS 1 I 1 allow action
```

Whoops… we must have forgotten about an earlier rule.

Revisiting the Access Control Policy

Rule 2 (Allow application Facebook) is not logging, so connection events are not generated

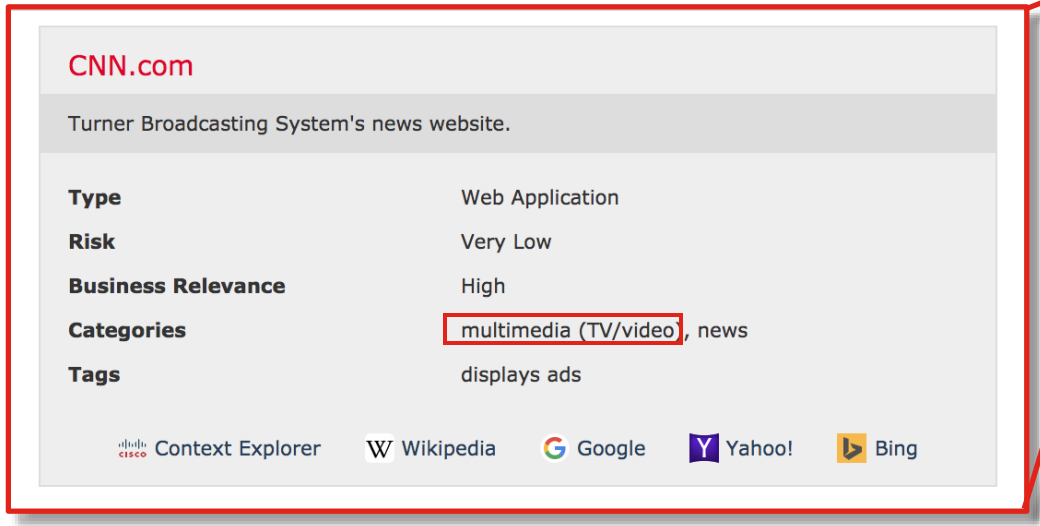


Key Takeaway: Firewall Engine Debug shows rule evaluation, even if logging is not enabled

Check Application Categories and Tags

Connection Events

First Packet	Last Packet	Action	Initiator IP	Responder IP	Source Port / ICMP Type	Destination Port / ICMP Code	Application Protocol	Web Application	Application Risk	Business Relevance	URL
2017-05-19 16:02:29		Block	192.168.62.63	151.101.65.67	54308 / tcp	80 (http) / tcp	HTTP	CNN.com	Medium	Medium	http://cnn.com/



Check Application Categories and Tags

firewall-engine-debug

```
192.168.62.63-54308 > 151.101.65.67-80 6 AS 1 I 0 New session
192.168.62.63-54308 > 151.101.65.67-80 6 AS 1 I 0 Starting with minimum 4, 'block by category', and SrcZone
first with zones 1 -> 2, geo 0 -> 0, vlan 0, inline sgt tag: untagged, ISE sgt id: 0, svc 0, payload 0,
client 0, misc 0, user 9999997, icmpType 0, icmpCode 0
192.168.62.63-54308 > 151.101.65.67-80 6 AS 1 I 0 pending rule order 4, 'block by category', AppID
192.168.62.63-54308 > 151.101.65.67-80 6 AS 1 I 0 Starting with minimum 4, 'block by category', and SrcZone
first with zones 1 -> 2, geo 0 -> 0, vlan 0, inline sgt tag: untagged, ISE sgt id: 0, svc 0, payload 0,
client 0, misc 0, user 9999997, icmpType 0, icmpCode 0
192.168.62.63-54308 > 151.101.65.67-80 6 AS 1 I 0 pending rule order 4, 'block by category', AppID
192.168.62.63-54308 > 151.101.65.67-80 6 AS 1 I 0 Starting with minimum 4, 'block by category', and SrcZone
first with zones 1 -> 2, geo 0 -> 0, vlan 0, inline sgt tag: untagged, ISE sgt id: 0, svc 0, payload 0,
client 0, misc 0, user 9999997, icmpType 0, icmpCode 0
192.168.62.63-54308 > 151.101.65.67-80 6 AS 1 I 0 pending rule order 4, 'block by category', AppID
192.168.62.63-54308 > 151.101.65.67-80 6 AS 1 I 0 URL SI: ShmDBLookupURL("http://cnn.com/") returned 0
192.168.62.63-54308 > 151.101.65.67-80 6 AS 1 I 0 Starting with minimum 4, 'block by category', and SrcZone
first with zones 1 -> 2, geo 0(0) -> 0, vlan 0, inline sgt tag: untagged, ISE sgt id: 0, svc 676, payload
1190, client 638, misc 0, user 9999997, url http://cnn.com/, xff
192.168.62.63-54308 > 151.101.65.67-80 6 AS 1 I 0 match rule order 4, 'block by category', action Block
192.168.62.63-54308 > 151.101.65.67-80 6 AS 1 I 0 sending block response of 605 bytes
192.168.62.63-54308 > 151.101.65.67-80 6 AS 1 I 0 Deleting session
```

Scenario 2: Network traffic failure through FTD

The customer states that FTD is causing network performance problems after a weekend migration from another vendor firewall

What we know:

1. The problem began right around the time users started arriving to the office.
2. Users are unable to open web sites.
3. The engineer is unable to join a WebEx.
4. The engineer states that Snort is “stuck at 100% utilization”

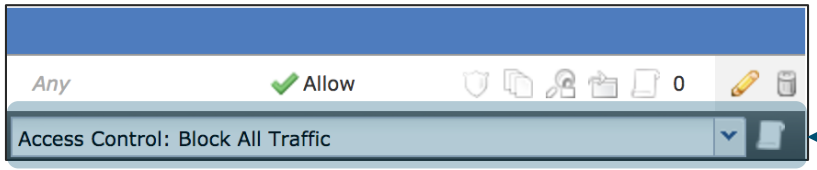
So, what does a “systemic approach to FTD troubleshooting” look like in this scenario?

Network traffic failure through FTD

Step 1: Given the impact and since we have no access to troubleshoot directly, we enable a Prefilter policy for all traffic to temporarily stop sending traffic to Snort.

This alleviates the problem and the engineer is able to join a WebEx.
Since a Prefilter policy improved the situation, we suspect a Snort oversubscription or policy issue.

Step 2: Visually review policy to determine what rule traffic would match



Customer had a “Block All” rule that was unintentional

What troubleshooting tool would have shown this without a visual inspection?

Network traffic failure through FTD

Minutes later, intermittent connectivity issues continue. Engineer's PC loses connectivity to Exchange.

```
capture capin type raw-data buffer 33000000 trace interface Inside
[Capturing - 25500768 bytes]  match tcp host 10.0.10.1 any eq https
```

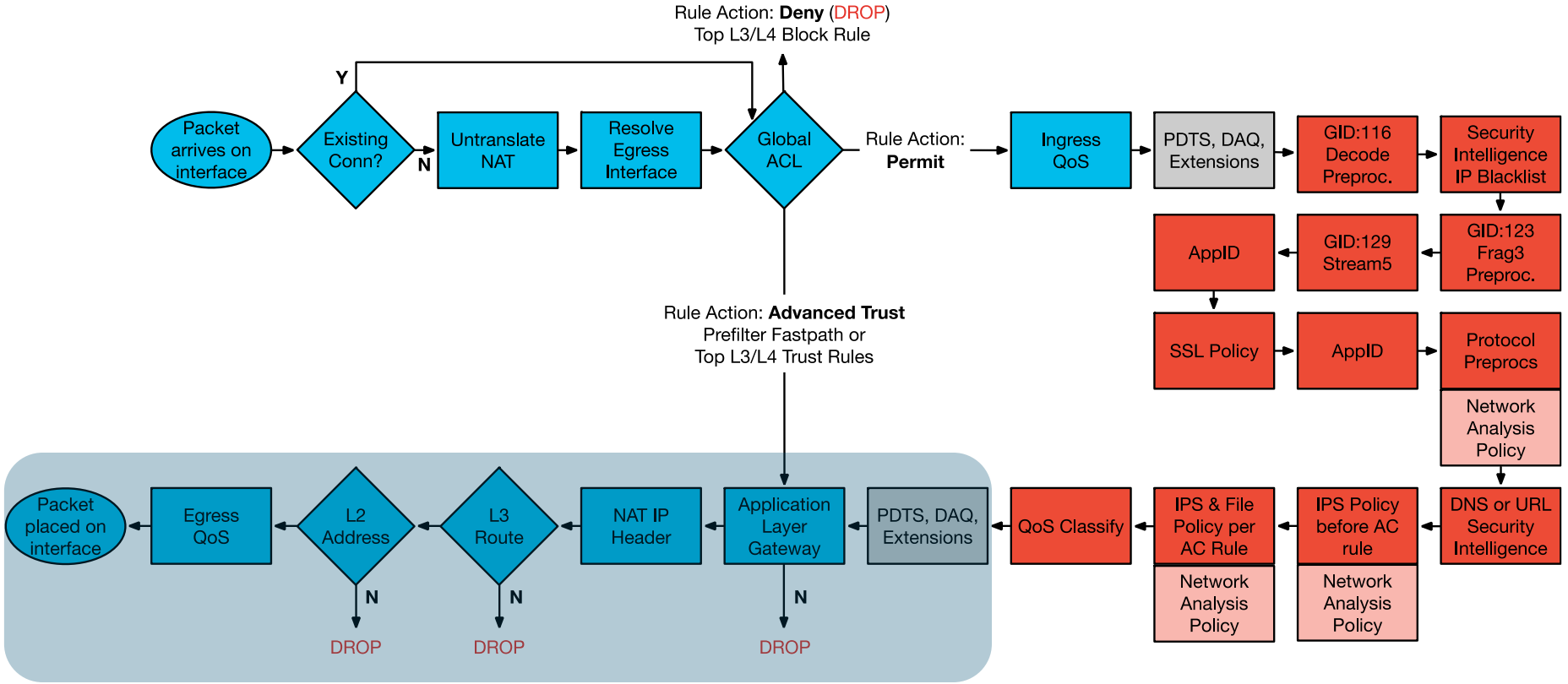
Enable capture for failing flow

```
firepower# sh cap capin | i S
   3: 13:23:11.905669       10.0.10.1.5377 > 192.0.2.194.443: S 2773524504:2773524504(0) win 8192
  19: 13:23:12.514499       10.0.10.1.5386 > 192.0.2.18.443: S 1117279318:1117279318(0) win 8192
  30: 13:23:12.797398       10.0.10.1.5379 > 192.0.2.98.443: S 3103152246:3103152246(0) win 8192
  32: 13:23:13.123650       10.0.10.1.5389 > 192.0.2.194.443: S 3496291677:3496291677(0) win 8192
  34: 13:23:13.163733       10.0.10.1.5387 > 192.0.2.194.443: S 3669311460:3669311460(0) win 8192
  43: 13:23:13.306411       10.0.10.1.5381 > 192.0.2.194.443: S 1115384746:1115384746(0) win 8192
  44: 13:23:13.446372       10.0.10.1.5390 > 192.0.2.194.443: S 3466698234:3466698234(0) win 8192
```

Identify instance of TCP connection attempt (SYN)

Based on what we learned today, what should we check next?

Reference Slide: Routed FTD Path of Packet



LINA ASA Engine = BLUE

Snort Engine = RED

Network traffic failure through FTD

Packet tracer output for affected traffic:

```
firepower# show capture capin trace pack 19
56752 packets captured

  19: 13:23:12.514499       10.0.10.1.5386 > 192.0.2.18.443: S 1117279318:1117279318(0) win
8192 Phase: 1
Type: CAPTURE
Subtype:
Result: ALLOW
Config:
Additional Information:
MAC Access list
...
Result:
input-interface: Inside
input-status: up
input-line-status: up
output-interface: Outside
output-status: up
output-line-status: up
Action: drop
Drop-reason: (nat-xlate-failed) NAT failed
```

Here we see that we have a NAT problem that is unrelated to Snort policy.

Network traffic failure through FTD

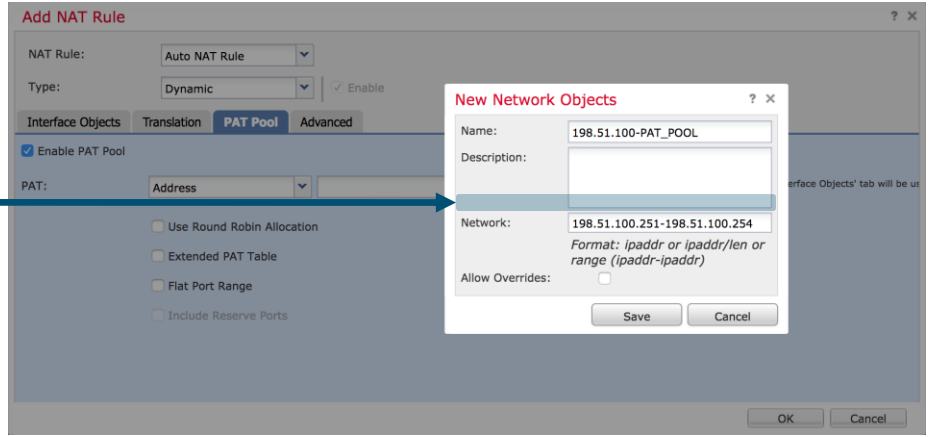
Check NAT pool allocations:

```
firepower# show nat pool
TCP PAT pool Outside, address 198.51.100.251, range 1-511, allocated 0
TCP PAT pool Outside, address 198.51.100.251, range 512-1023, allocated 0
TCP PAT pool Outside, address 198.51.100.251, range 1024-65535, allocated 64512
UDP PAT pool Outside, address 198.51.100.251, range 1-511, allocated 2
UDP PAT pool Outside, address 198.51.100.251, range 512-1023, allocated 0
UDP PAT pool Outside, address 198.51.100.251, range 1024-65535, allocated 23
firepower#
```

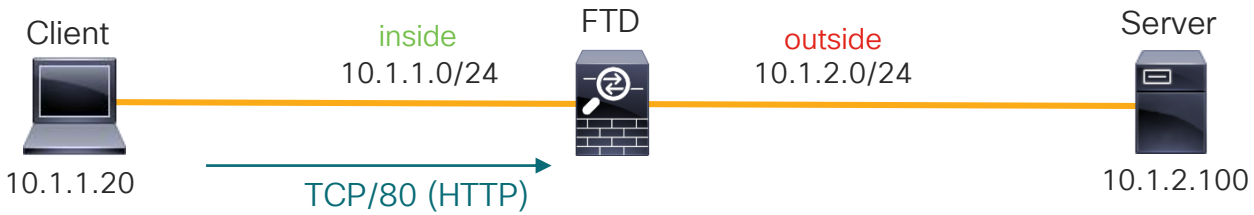
PAT pool port exhaustion

Solution:

Add more IP addresses to PAT pool



Scenario 3: Clients cannot reach server



The customer states that clients traversing FTD are not able to access an internal web server. However, other clients on the server subnet (10.1.2.0/24) are able to access the server.

Let's troubleshoot this using a systematic approach to FTD troubleshooting

Using our FTD troubleshooting tools

Remember: Always check events and syslogs! FMC: Analysis → Connections → Events

No events found! (Always make sure you're logging the rule that you expect to be hitting!)

Fortunately, we did enable Lina syslogs to an external server. Here's what we found:

```
%ASA-6-302013: Built inbound TCP connection 46927 for inside:10.1.1.20/2286 (10.1.1.20/2286) to outside:10.1.2.100/80 (10.1.2.100/80)
%ASA-6-302014: Teardown TCP connection 46927 for inside:10.1.1.20/2286 to outside:10.1.2.100/80 duration 0:00:30 bytes 0 SYN Timeout
```

So, now we know that we are receiving the packet but either the server is not responding or FTD is not forwarding it. Let's dig deeper. Maybe snort is dropping it…

```
> system support firewall-engine-debug

Please specify an IP protocol: tcp
Please specify a client IP address: 10.1.1.20
….
10.1.1.20-2286 > 10.1.2.100-80 6 AS 1 I 16 New session
10.1.1.20-2286 > 10.1.2.100-80 6 AS 1 I 16 using HW or preset rule
order 5, 'Allow_Inside_to_Outside', action Allow and prefilter rule 0
10.1.1.20-2286 > 10.1.2.100-80 6 AS 1 I 16 allow action
```

It looks like Snort allows it. So what next?

Packet Captures – The single source of truth

What do we know at this point?

FTD is receiving the packet. We are building the TCP connection for the flow. Snort is NOT dropping the packet.

The next step here is to determine if FTD is actually forwarding the packet.
Let's use our awesome packet capture tools for this.

Verify ingress captures so we can line them up with egress captures:

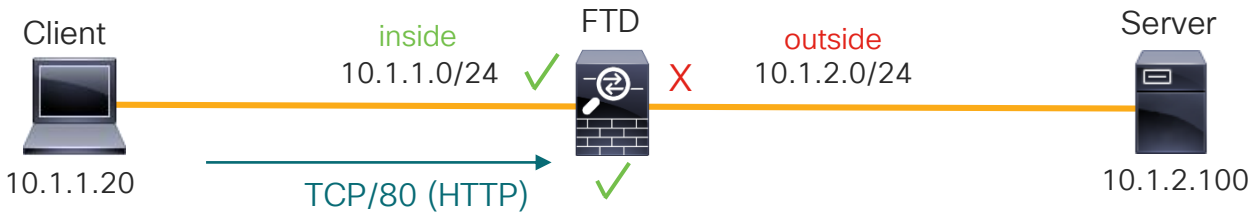
```
firepower# show capture
capture capin type raw-data trace interface inside [Buffer Full - 524216 bytes]
  match tcp host 10.1.1.20 host 10.1.2.100 eq www
firepower# sho cap capin | i 2286
 322: 13:04:56.926786       802.1Q vlan#36 P0 10.1.1.20.2286 > 10.1.2.100.80: S
1336706021:1336706021(0) win 512
firepower#
```

```
capture capout type raw-data interface outside
[Capturing - 0 bytes]
  match tcp any host 10.1.2.100 eq www
```

Houston...we have a problem.

No packets going to the destination server?

Visual troubleshooting recap



- Packet is received
- Lina is building connection
- Snort is not dropping
- However, FTD is not forwarding

Let's review! What are possible reasons that FTD may drop traffic without a Lina syslog or snort verdict indicating a drop?

Checking Lina inspection and L2 adjacency

Remember, we can use packet capture with the ‘trace’ command to see policy deci

```
firepower# show cap capin trace packet-number 1

7084 packets captured

   1: 13:04:12.548204       802.1Q vlan#36 P0 10.1.1.20.2286
> 10.1.2.100.80: S 1277167793:1277167793(0) win 512
...
Phase: 14
Type: ROUTE-LOOKUP
Subtype: Resolve Egress Interface
Result: ALLOW
Config:
Additional Information:
found next-hop 10.1.2.50 using egress ifc  outside

Result:
…
output-interface: outside
…
Action: allow
```

We can see that configured policies are not dropping the packet. However, it is strange that our next hop is not the directly-connected server.

Let’s investigate this…

Next-hop ARP resolution?

```
firepower# sh arp | i 10.1.2.50
firepower#
```

Check for ARP entry. Does not exist.

Reason for packet drop:

```
firepower# debug arp
debug arp  enabled at level 1
arp-req: generating request for 10.1.2.50 at interface outside
arp-req: request for 10.1.2.50 still  pending
```

We can see that ARP resolution is failing for this host. Therefore FTD cannot egress the packet.

Root cause:

```
firepower# show route
…
S        10.1.2.100 255.255.255.255 [1/0] via 10.1.2.50, outside
```

A static, more specific /32 route to the server via 10.1.2.50 is configured and that host is not responding to ARP.

You make **possible**