
TOMORROW starts here.
cisco
Cisco live!

Cisco Advanced ASA Firewalls Inside-Out

BRKSEC-3660

Sasa Rasovic

Incident Manager, Cisco PSIRT

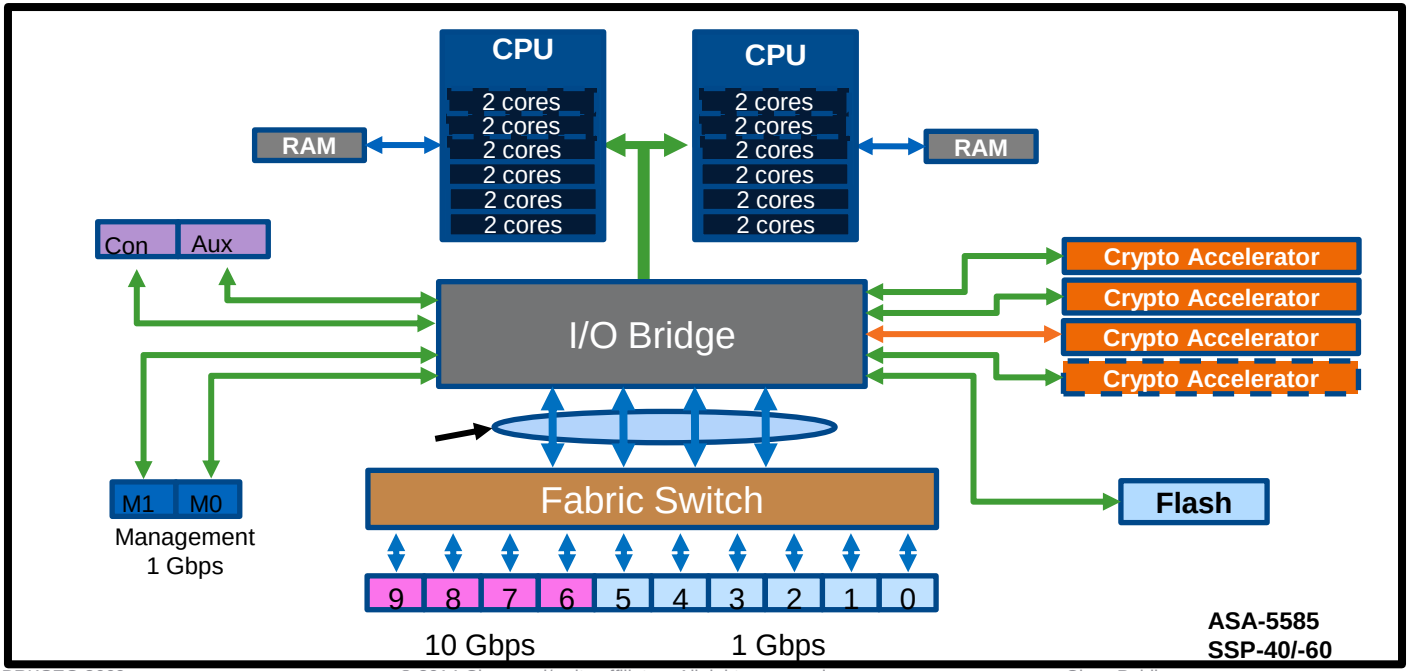
Cisco live!

Agenda

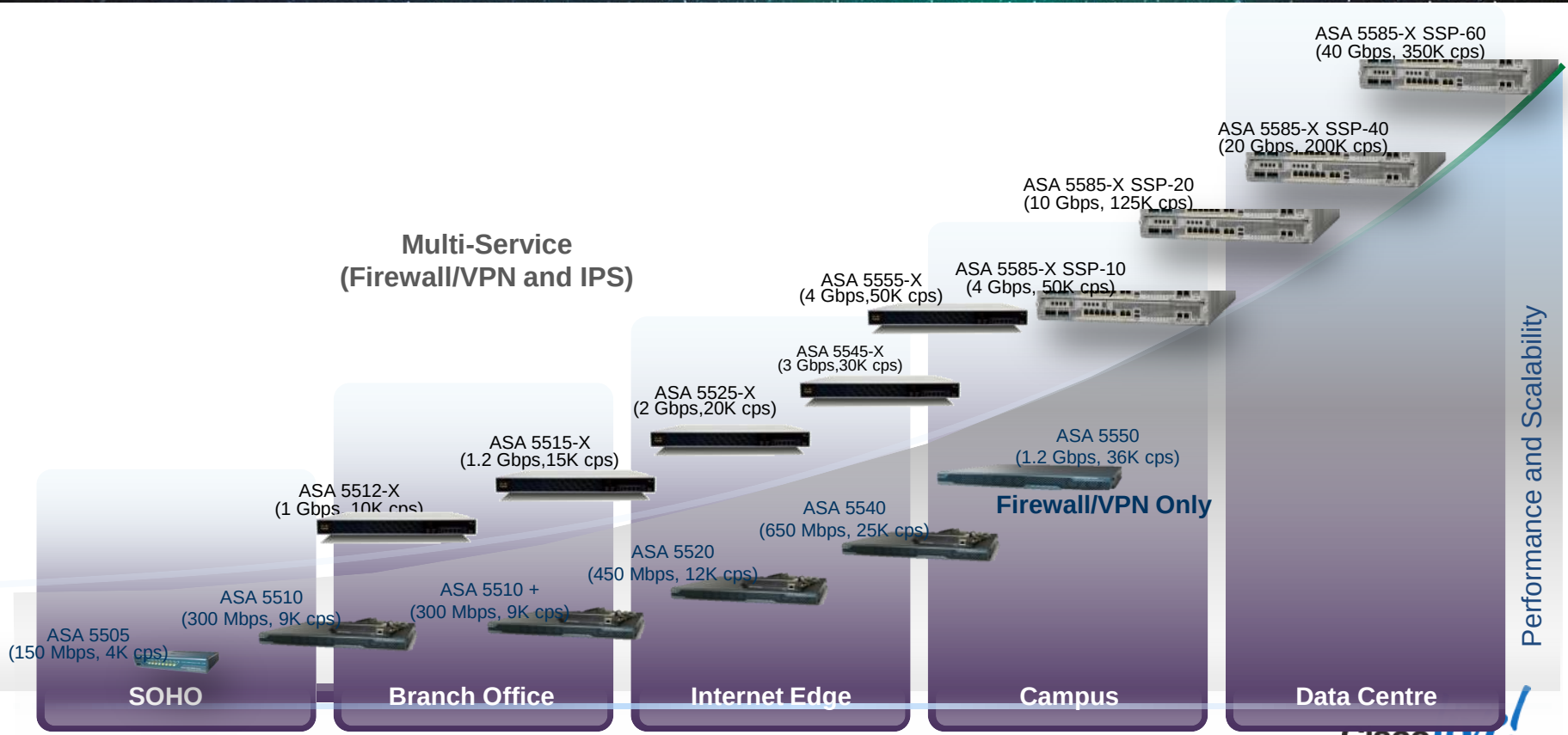
- Intro
- Architecture Overview
- NIC Driver
- Block Infrastructure (Multi-Core Platforms)
- Dispatch Layer
- ASP Media Layer
- Flow Lookup and Processing
- Troubleshooting Demo
- Control Plane Processing and Application Inspection
- Closing Remarks

ASA Inside-Out

- We will follow the path of the packet within the box in order to:
  - Understand how is the ASA structured Inside-Out
  - Understand how, what and when to troubleshoot
  - Understand how it all together affects performance



Cisco ASA Portfolio

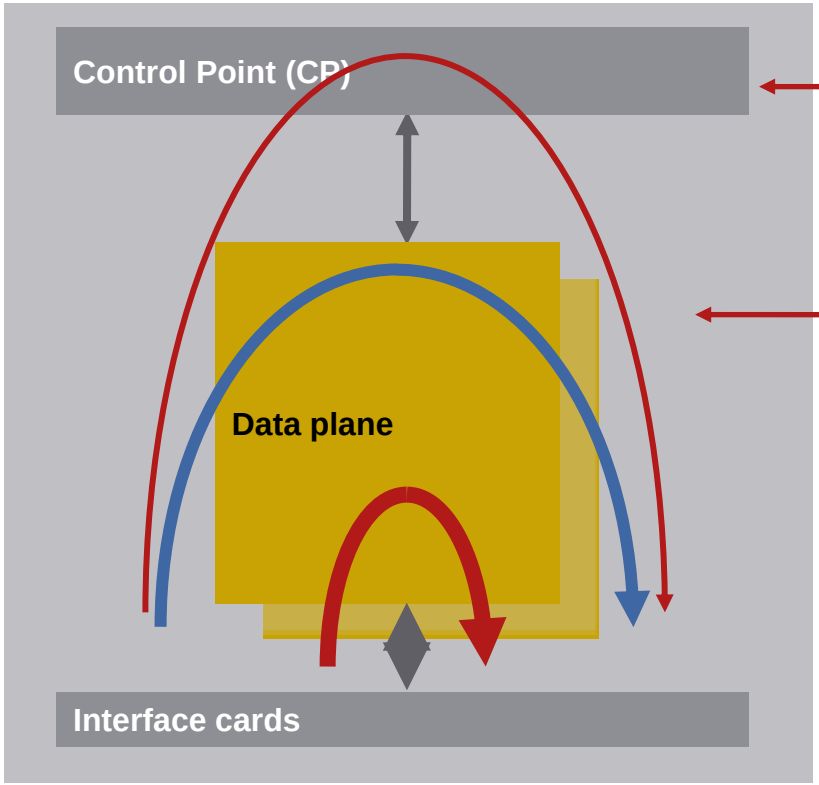





Architecture Overview

ASA Architectural Overview

Software



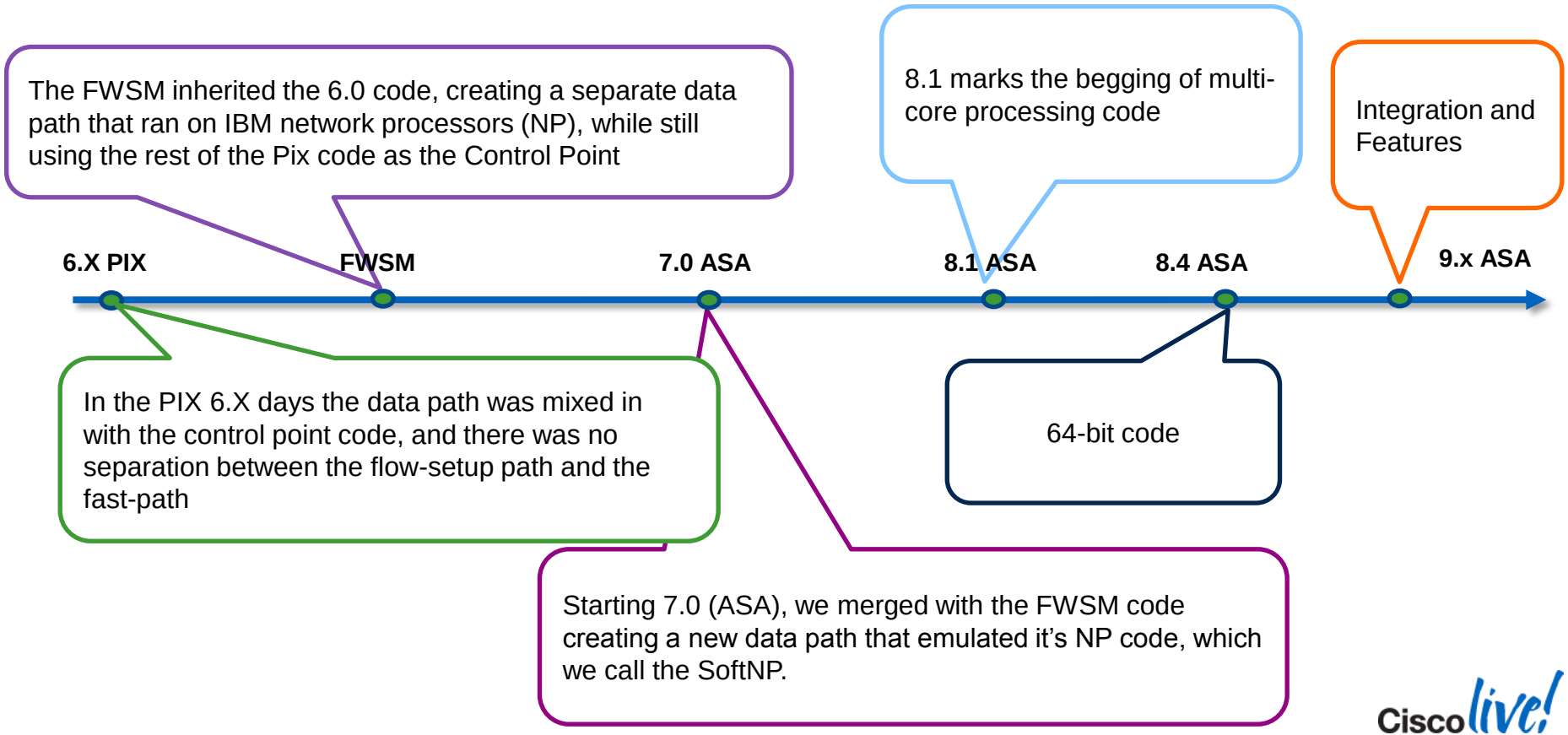
Control Point

ACL Compilation,
Fixups, Syslog, AAA

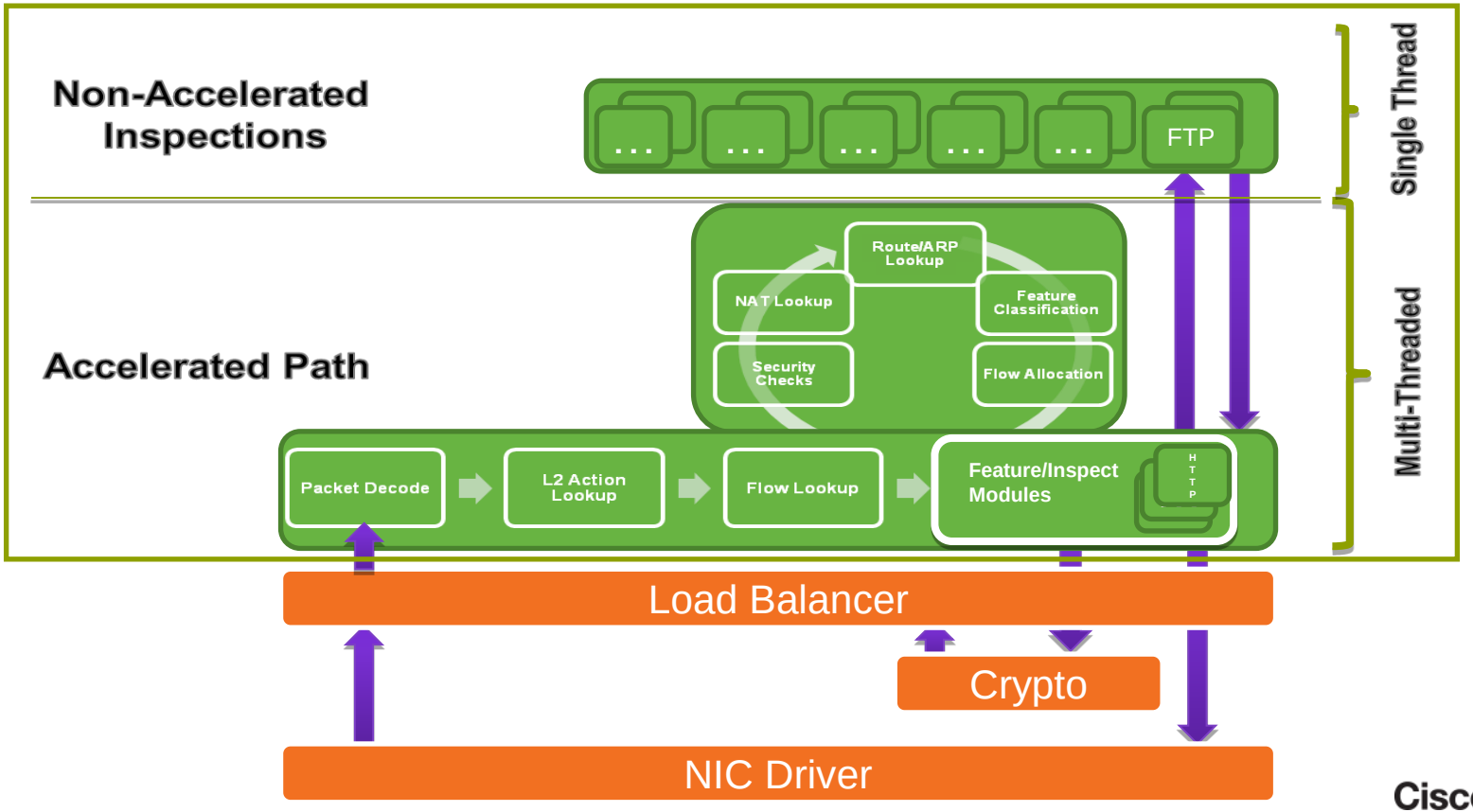
Accelerated Security Path (ASP) consists of SP and FP:

Session Establishment and Teardown, AAA Cache, ACLs, Flow Identification, Security Checks and NAT

ASA Data Path - Timeline



ASA Packet Flow






NIC Driver

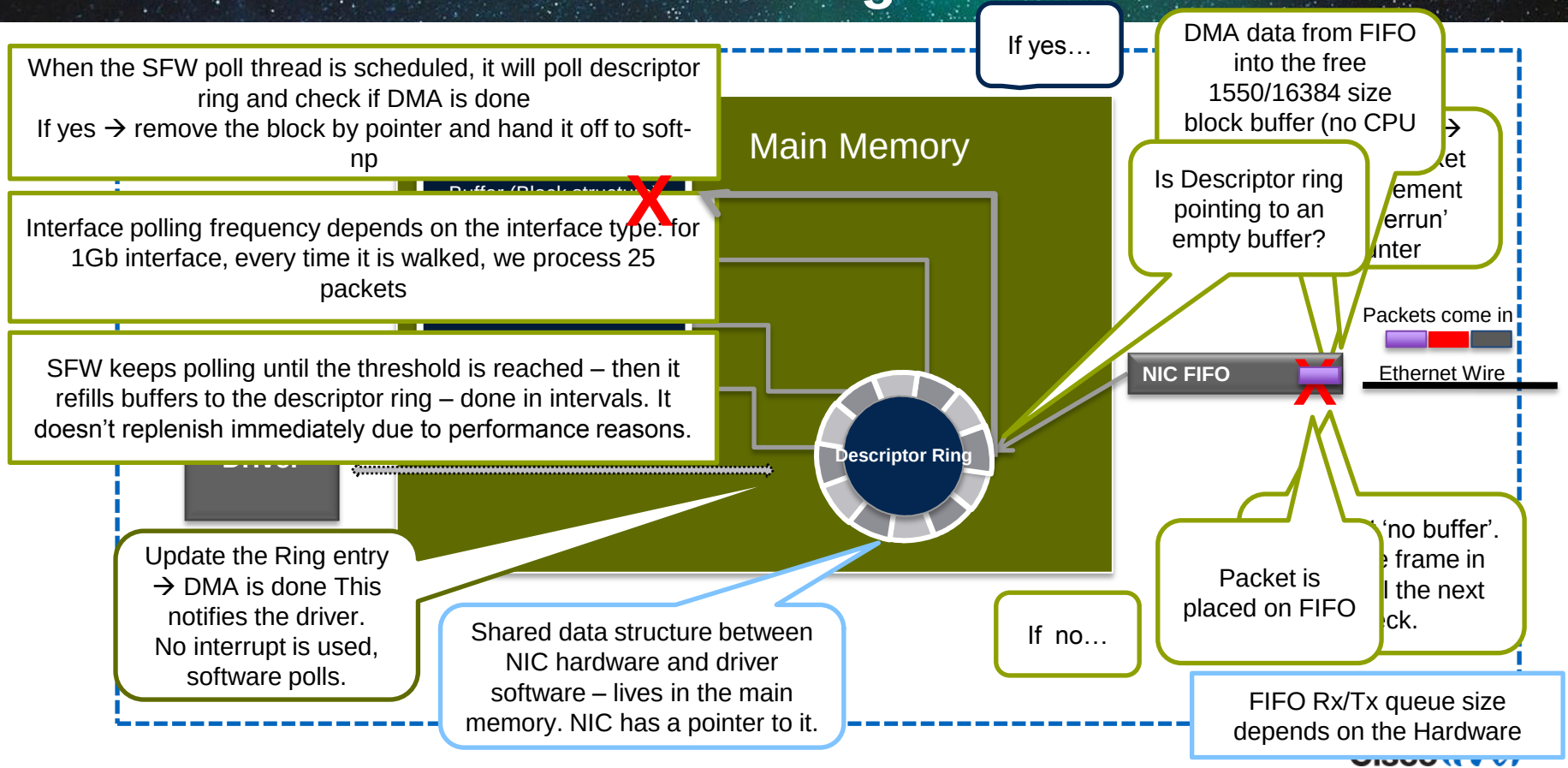
Hardware NIC Overview

- Network Interface Card (NIC) comes in several flavors
  - 100Mb, 1Gb and 10Gb interfaces
  - Uplinks and MAC Uplinks
- Fundamental characteristics:
  - FIFO queuing at the interface level
  - No CPU involvement for packet receive and transmit operations – DMA to and from the memory
  - Receive (Rx) and Transmit (Tx) descriptor rings are shared structures between physical (NIC) and memory (blocks/buffers) layer. They describe Physical Layer to CPU, and Memory to NIC

ASA Ingress Frame Processing Overview

- Frames are received from wire into ingress FIFO queues
  - FIFO Size: 48KB/16KB (Rx/Tx) on 1GE revenue ports, 4x512KB on 10GE
- NIC driver moves frames to main memory through Rx rings
  - Each ring slot points to a main memory address (“block” or “buffer”)
  - Rings are describing memory to interface, and the other way around
  - Single Rx ring per 1GE (255 or 512 slots) except ASA5585
  - Four Rx rings per 10GE (4x512 slots) with hashed load-balancing
  - Shared Rx rings on MACs (ASA5585) and 1GE uplink (ASA5505)
- CPU periodically scans the rings to pull packet blocks and refill slots with pointers to other free blocks

NIC Hardware Flow – Receiving a Packet



ASA Jumbo Frames

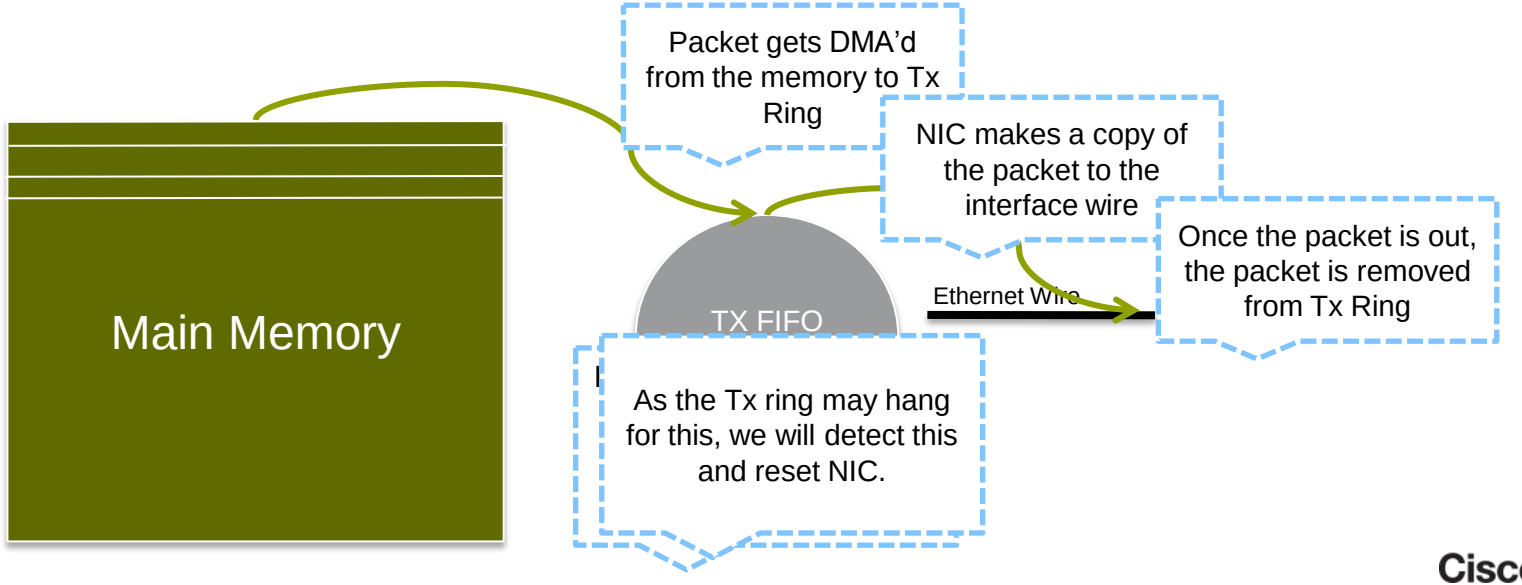
- ASAs 5580/5585 support Jumbo Ethernet frames (9216 bytes including L2 + FCS)
  - There is a separate DMA interface block pool for Jumbo frames
  - ASA uses 16KB blocks
  - It may limit the use of other features on the interface
  - It requires rebooting a device

```
asa(config)# mtu inside 9216
asa(config)# jumbo-frame reservation
WARNING: This command will take effect after the running-config is saved and the system
has been rebooted. Command accepted.
```

- Jumbo frames mean less cycles spent in fragmentation and reassembly
  - Requires end-to-end implementation
- From a processing perspective: Big Data chunk + a small packet header = less processing cycles for intermediate devices = more throughput

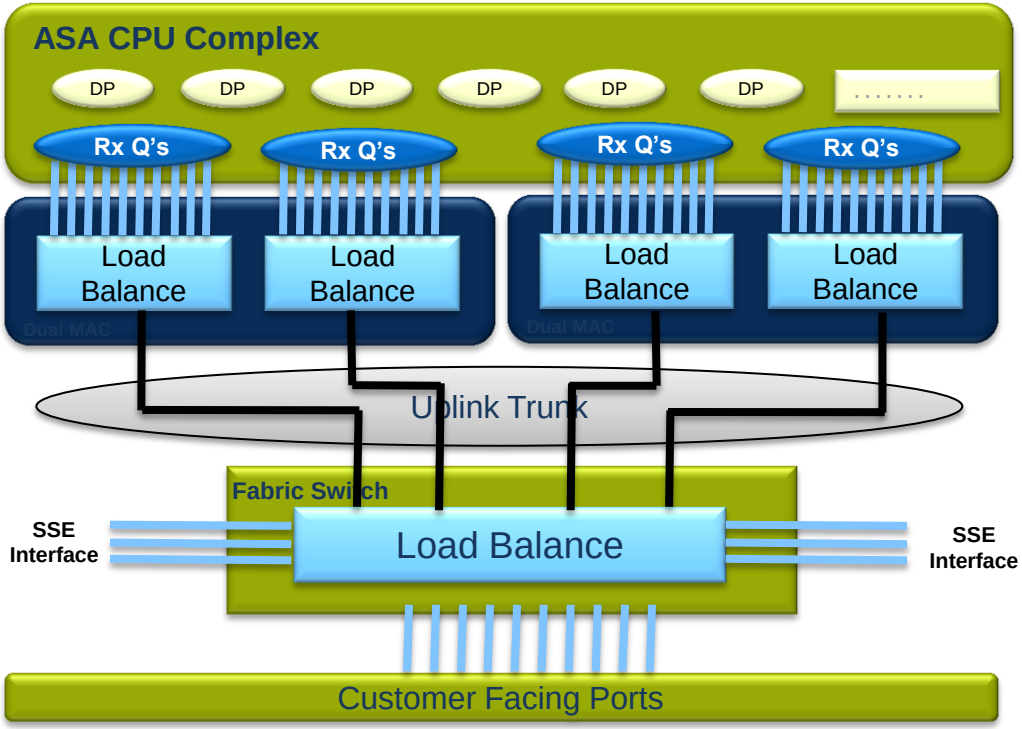
ASA Egress Frame Processing Overview

- For egress processing we follow the reverse path:
  - Once processed, packet is DMA'd to a Tx ring by means of a pointer
  - Shared rings on MACs (ASA5585) and 1GE uplink (ASA5505)
  - This is where we do LLQ before the packet is placed on the wire



ASA 5585 Packet Path to CPU

- Multi layered Load Balancing
  - Switch will do load balancing across the Uplinks towards the MAC
  - 2nd Level MAC: does the same (slighly different input for hash) towards the queues (MAC RX rings)
  - Cores then latch onto those rings and fetch packets according to their own LB scheme: ASP LB
  - Flows that can not be decoded, go to RX-0
- Flow control
  - Link pause is supported
  - Receive Flow Control – Triggered by congestion at Rx Queue. (Off by default)
  - Transmit Flow control – Controlled by MAC (Always enabled)



ASA Interface Rings - 5580

```
asa5580-40-1# show interface TenGigabitEthernet 5/0 detail | beg Queue stats

Queue stats:
          RX[00]: 0 packets, 0 bytes
                  Blocks free curr/low: 511/0
          RX[01]: 0 packets, 0 bytes
                  Blocks free curr/low: 511/0
          RX[02]: 0 packets, 0 bytes
                  Blocks free curr/low: 511/0
          RX[03]: 0 packets, 0 bytes
                  Blocks free curr/low: 511/0
          TX[00]: 0 packets, 0 bytes, 0 underruns
                  Blocks free curr/low: 511/511
          TX[01]: 0 packets, 0 bytes, 0 underruns
                  Blocks free curr/low: 511/511
          TX[02]: 0 packets, 0 bytes, 0 underruns
                  Blocks free curr/low: 511/511
          TX[03]: 0 packets, 0 bytes, 0 underruns
                  Blocks free curr/low: 511/511
```

RX/TX Rings

- When the packet is dispatched from the RX FIFO, it will fit a particular Ring, based on it's 5-touple
- All the packets belonging to the same flow go to the same ring. (Caveat: too many single-flow packets can fill up a RX ring, causing a backpressure on Rx FIFO queue – other rings will thus suffer as well)

ASA Interface Rings – 5585 and ASA-SM

```
Asa5585-10(config)# show controller Internal-Data0/0 detail | inc Receive Ring
   Receive Ring[0]
   Receive Ring[1]
   Receive Ring[2]
   Receive Ring[3]

Asa5585-10(config)# show controller Internal-Data0/0 detail | beg Receive Ring[0]
   Receive Ring[0]
       Receive Ring [0]:
         Control (RXDCTL)       : 0x02000000
         Split RxCntrl (SRRCTL): 0x0200000a
         Base Address (RDBAH/L): 00000000-3f127000
         Length (RDLEN)         : 8192
         Num of RX Descriptors : 512
         Head/Tail (RDH/RDT)    : 424/423
         Driver Head/Tail       : 424/423
         rx_pkts                : 347048
         rx_bytes               : 50450564
         rx_drops               : 0
         rx_lec                 : 0
         num_rxbufs             : 511
Asa5585-10(config)# show inter detail | beg Interface Internal-Data0/0
Interface Internal-Data0/0 "", is up, line protocol is up
  Hardware is i82599 xaui rev01, BW 10000 Mbps, DLY 10 usec
```

Number of rings depends on platform

Number of packets hashed to this ring

RX Rings

For interface errors on 5585, check Internal-Data MAC.

ASA Interface and Traffic Stats

```
ASA# show interface g0/1
Interface GigabitEthernet0/1 "DMZ2", is up, line protocol is up
  Hardware is i82546GB rev03, BW 1000 Mbps, DLY 10 usec
        Auto-Duplex(Full-duplex), Auto-Speed(1000 Mbps)
        Input flow control is unsupported, output flow control is unsupported
        MAC address 0024.97f0.4edb, MTU 1500
        IP address 10.10.10.1, subnet mask 255.255.255.0
        39645 packets input, 4980966 bytes, 77 no buffer
        Received 192 broadcasts, 0 runts, 0 giants
        0 input errors, 0 CRC, 0 frame, 12 overrun, 0 ignored, 0 abort
        37599 L2 decode drops
        6011 packets output, 756890 bytes, 0 underruns
        0 pause output, 0 resume output
        0 output errors, 0 collisions, 1 interface resets
        0 late collisions, 0 deferred
...
```

Unable to move frame to main memory

Full Ingress FIFO -> dropped packets

No nameif configured or invalid VLAN frame

Full Egress FIFO -> dropped packets

Use this command to get the average size of the packet hitting the interface

```
ASA# show traffic
inside:
                received (in 499515.280 secs):
                         1390895 packets135993684 bytes
                         372 pkts/sec    521888 bytes/sec
                transmitted (in 499515.280 secs):
                         776339 packets  72598252 bytes
                         1 pkts/sec      7 bytes/sec
      1 minute input rate 200 pkts/sec, 28400 bytes/sec
      1 minute output rate 1 pkts/sec,  149 bytes/sec
      1 minute drop rate, 1 pkts/sec
      5 minute input rate 2 pkts/sec,  285 bytes/sec
      5 minute output rate 1 pkts/sec,  140 bytes/sec
      5 minute drop rate, 1 pkts/sec
```

Bursts can exhaust your firewall's FIFO/RX rings, but can be hinted if per-minute values are observed:
28400 b/s / 200 p/s = 143 B average size of the packet

NIC Troubleshooting Considerations

- NIC packet drops are caused by:
  - FIFO is full – `show interface` will show overruns = input errors

Small packets bursts can cause FIFO to overflow (even at low speeds)

```
ASA# show interface g0/1
4134256809 input errors, 0 CRC, 0 frame, 4134256809 overrun, 0 ignored, 0 abort
```

  - FIFO is full due to Descriptor ring not pointing to free buffer blocks – both 'overrun' and 'no buffers' indications will show up in `show interface`

Fixed block size (jumbo/no-jumbo)

```
ASA# show interface g0/1
49465365 packets input, 13850409151 bytes, 3570137 no buffer
```

  - CPU hog is causing delays in scheduling a thread for a Ring refill – Look for constant hogs in **`show process cpu-hog`**

```
ASA# show process cpu-hog
Process:      DATAPATH-0-1376, PROC_PC_TOTAL: 2417983, MAXHOG: 16, LASTHOG: 2
LASTHOG At:   19:00:12 KST Dec 14 2011
PC:           0x0000000000000000 (suspend)

Process:      DATAPATH-1-1377, PROC_PC_TOTAL: 1407242, MAXHOG: 25, LASTHOG: 1
LASTHOG At:   19:00:12 KST Dec 14 2011
PC:           0x0000000000000000 (suspend)
```

Ways to Fight Overruns 1: Flow Control

- IEEE 802.3x mechanism to inform the transmitter that the receiver is unable to keep up with the current data rate
  - Receiver sends a special Pause frame (XOFF) to temporary halt transmission and Resume (XON) frame to continue
  - XOFF is sent when buffer usage exceeds high-watermark – XON when it drops bellow low-watermark
  - The duration of the pause is specified in the frame – pause time unit is the amount of time to transmit 64 bytes
  - Link partner (L2 switch) will resume traffic when receiving XON or after advertised lifetime expires – adjacent L2 partner need to have it enabled!
  - Helps to eliminate overrun errors but it may cause packet drops/losses upstream
  - Tune watermarks for best performance

```
asa(config)# interface TenGigabitEthernet7/1
asa(config-if)# flowcontrol send on 64 128 26624
Changing flow-control parameters will reset the interface. Packets may
be lost during the reset. Proceed with flow-control changes?
```

It depends on the link speed

Optional low FIFO watermark in KB (0-511)

Optional high FIFO watermark in KB (0-511)

Optional duration (refresh interval)

Ways to Fight Overruns 2: EtherChannel

- Introduced in ASA 8.4 software
  - Up to 8 active and 8 standby port members per Ether-channel, up to 48 channels
  - Only same type/speed interfaces
  - Not supported on ASA5505 and 4GE-SSM ports
- Load balancing scheme
  - ASA load balances flows by hashing Src/Dst IP to a 3-bit value (8 possibilities) and round robin across members – this can be changed
  - As a result of the hashing scheme, best load distribution is with 2, 4 or 8 interfaces in the bundle (divided by 8)
  - Load is thus distributed across multiple FIFOs/RX rings, eliminating interface oversubscription
  - May help with unequal CPU load balancing on multi-core platforms
  - Single flow packets will always land on the same link




Dispatch Layer

Dispatch Unit/ASP Load Balancer

- Main poll loop for the system – runs continuously, never sleeps
- Goes to each interface in its poll-loop and tries to get a fixed number of packets before moving on to the next interface
  - The dispatch layer also polls loopback interfaces and crypto devices
- On multi-core platforms, it is responsible for removing the packet from the Rx, to be processed by a Core (per-core run)
- Each core is scheduled to search for an available interface Rx ring: when it does, it latches onto it for a time (no other core can serve that ring during that time)
- The core will release a ring after a scheduled time and look for another ring

ASA Packet Load Balancer

- A problem: what if the packet in the ring (serviced by a core) belongs to an already existing connection on a different core? (usually happens on systems with few interfaces and multiple cores)
- The load balancer must hand off the packet to that core, causing inter-core queuing and performance problems
- asp load-balance per-packet is a solution to this: release the ring after pulling a single packet from it.

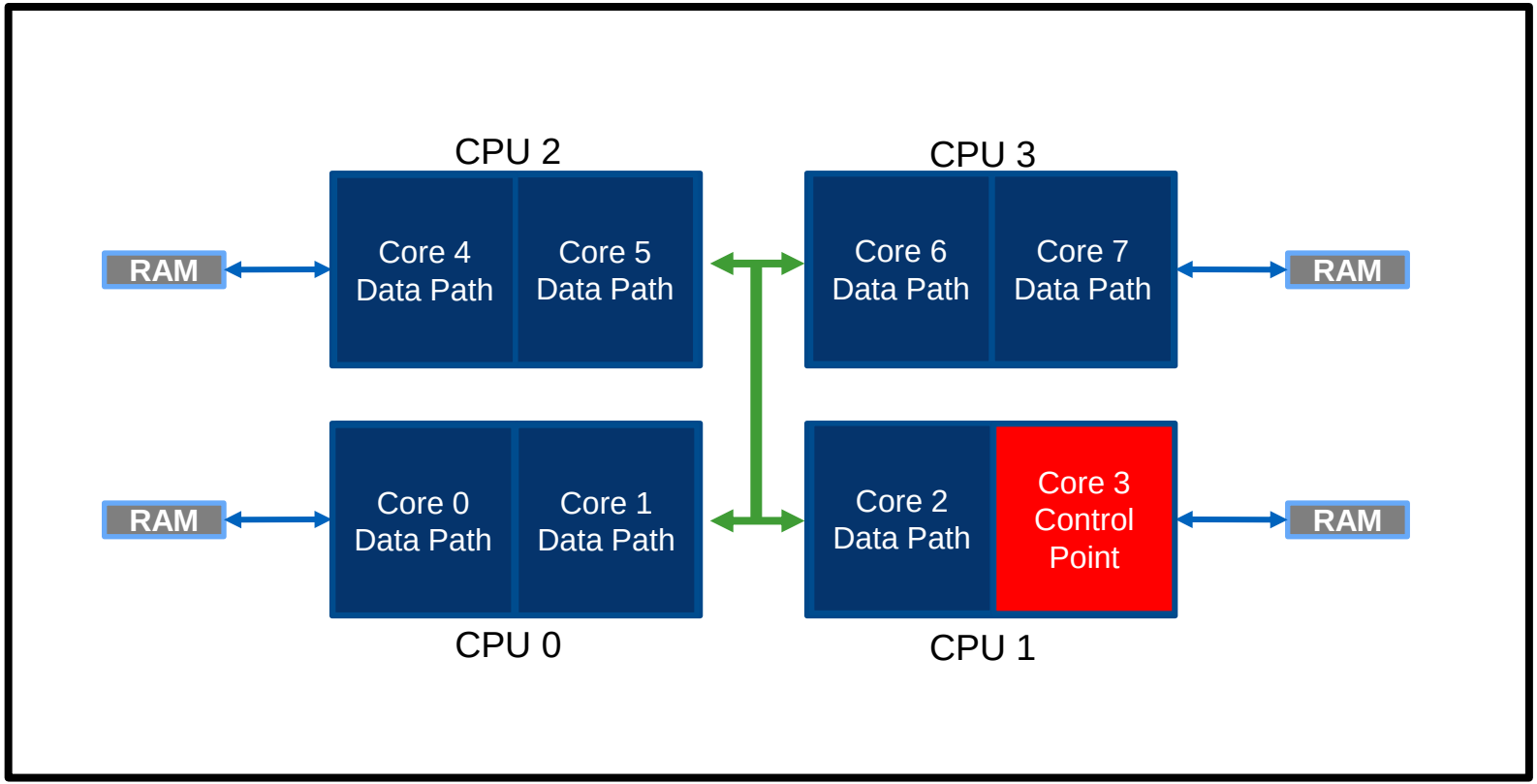
```
ASA5585# sho asp load-balance detail

Histogram of 'ASP load balancer queue sizes'
  64 buckets sampling from 1 to 65 (1 per bucket)
  0 samples within range (average=0)
<no data for 'ASP load balancer queue sizes' histogram>

Data points:
  bucket[1-1] = 457186 samples
  bucket[2-2] = 112793 samples
  bucket[3-3] = 0 samples
  bucket[4-4] = 0 samples
  bucket[5-5] = 0 samples
  bucket[6-6] = 0 samples
  bucket[7-7] = 0 samples
…
```

Blocks will get queued in case of inter-core hand-off

Single Flow Performance

- All packets within a single flow will always take the same path through the system
- All packets within a single flow will always be hashed to the same TX/RX rings
- All packets within a singe flow will always be serviced by the same Core
- This may, in effect, have detrimental effect on the traffic performance

ASA Multi-core Control Point Operation



Understanding CPU Usage and Scheduling

- Cores are scheduled for trading off the roles:
  - Control point - Basic system functions and some inspections [There's only one single-thread instance of the CP at a time]
  - DP (Data Path) - Processing packets from and to the interfaces [All other cores]

Example: System under negligible data-path load, but heavy control-point load.

Data Path – check for relevant imbalances between cores

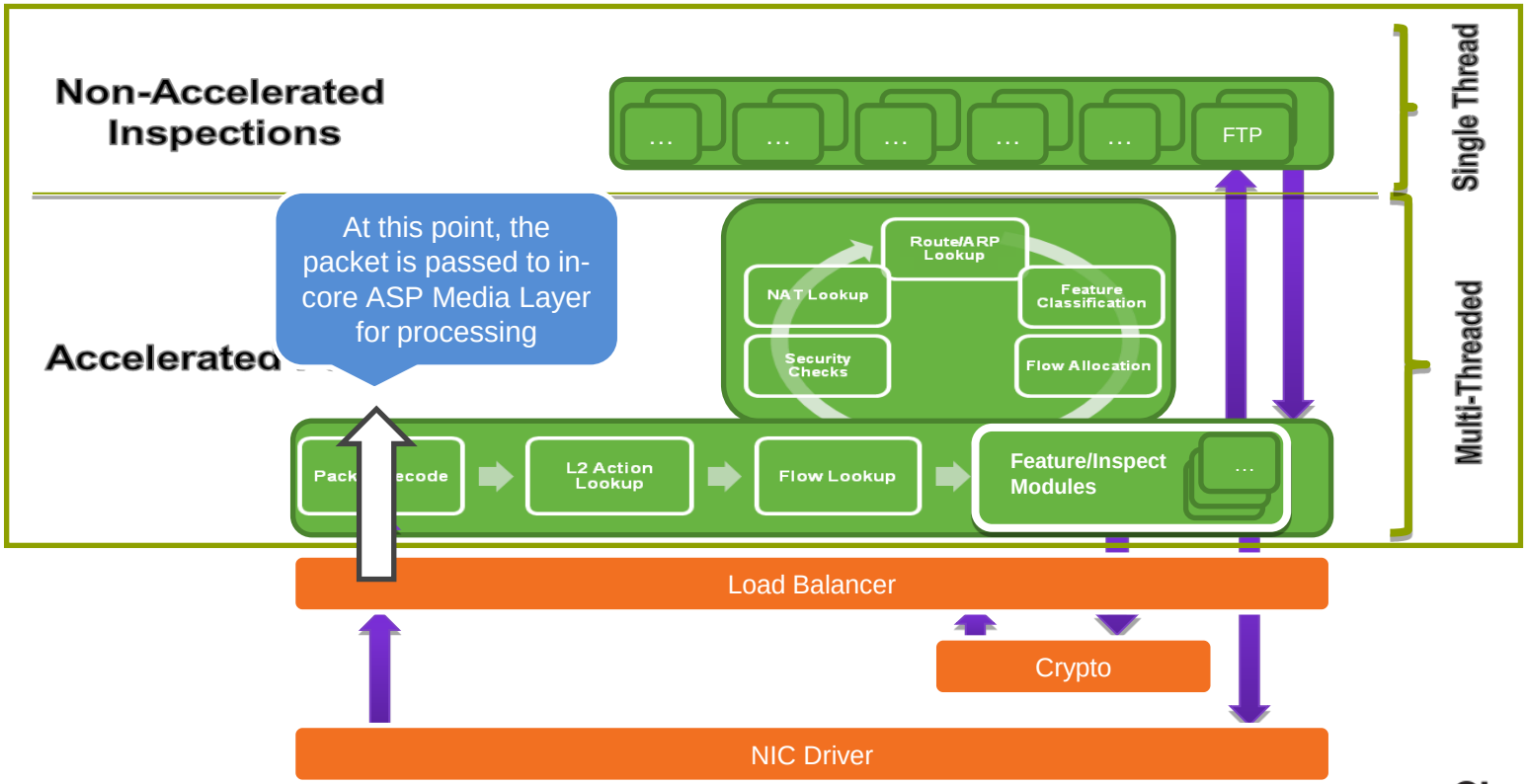
```
ASA-5585# show cpu usage detailed

Break down of per-core data path versus control point cpu usage:
Core                5 sec                1 min               5 min
Core 0        99.5 (99.5 + 0.0)  41.3 (41.2 + 0.0)  10.2 (10.2 + 0.0)
Core 1        99.5 (99.5 + 0.0)  40.9 (40.5 + 0.4)  10.0 ( 9.9 + 0.1)
Core 2        99.5 (99.5 + 0.0)  40.6 (40.5 + 0.1)  10.0 ( 9.9 + 0.0)
Core 3        99.5 (99.5 + 0.0)  40.5 (40.5 + 0.0)   9.9 ( 9.9 + 0.0)
Core 4        99.5 (99.5 + 0.0)  41.2 (40.3 + 0.8)  10.1 ( 9.9 + 0.2)
Core 5        99.5 (99.5 + 0.0)  41.7 (41.2 + 0.4)  10.3 (10.2 + 0.1)
Core 6        99.5 (99.5 + 0.0)  40.5 (40.5 + 0.0)   9.9 ( 9.9 + 0.0)
Core 7        99.5 (99.5 + 0.0)  41.6 (41.3 + 0.3)  10.3 (10.2 + 0.0)

Current Control Point load as a percentage of maximum Control Point load:
  for 5 seconds = 0.0%; 1 minute: 4.8%; 5 minutes: 1.5%
```

Useful Commands

For Your Reference

- `Show cpu core`
- `Show cpu core all`
- `Show cpu detail`
- `Show proc cpu-usage`
- `Show processes cpu-usage sorted busy-only`
- `Show asp load-balance detail`

ASA Packet Flow






ASP Media Layer

Media Layer

- Does a L2 decode
- On input, determines which VLAN the packet is on and thus on which virtual interface the packet arrived
- On output, use the output virtual interface to determine which VLAN the packet is to be sent on and write that in the L2 header (along with the source MAC)
- Higher layers are largely ignorant of VLANs

Packet Decode

- Parsing of the headers: Do L3 and L4 decode
- Store significant fields in the Block header or SoftNP Meta Data:
  - Addresses and Ports
  - L3/L4/L5 protocol offsets
- Detects simple L2/L3/L4 protocol violations (e.g. mismatched lengths or checksum errors) and drops those packets
- Fragment Reassembly – full or virtual
- Determine target context if in multimode

We don't capture packets that fail reassembly

```
ASA# show fragment
Interface: outside
    Size: 200, Chain: 24, Timeout: 5, Reassembly: virtual
    Queue: 0, Assembled: 207392, Fail: 2035, Overflow: 1937
Interface: dmz
    Size: 200, Chain: 24, Timeout: 5, Reassembly: virtual
    Queue: 0, Assembled: 0, Fail: 0, Overflow: 0
```

L2 Action Lookup

- ARP punts to control plane – the packet is buffered while waiting for a decision from CP (may have severe effect in case of ARP storms)
- DHCP punts to control plane
- MAC access-list (transparent mode)
- Configured captures – **captures may starve CPUs on SNP ASAs in case there is a lot of matching traffic – massive spin locks on multiple cores, have a potential of exhausting interface blocks**

```
ASA# show capture cap

4 packets captured

   1: 17:40:48.795613 802.1Q vlan#1527 P0 192.168.2.10.12345 > 192.0.4.126.80: S 0:492(492) win 8192
   2: 17:40:48.795613 802.1Q vlan#1527 P0
   3: 17:40:48.796818 802.1Q vlan#1527 P0 192.0.4.126.80 > 192.168.2.10.12345: S
3900802120:3900802120(0) ack 1000 win 3129 <mss 536>
```

Flow Lookup

Flow Lookup

- We perform flow lookup against our flow hash table for previously parsed 5-touple + incoming interface information
- As a result of flow lookup, the packet is either passed to ASP Data Path (SoftNP) for further processing/session establishment, or pushed towards CP.
- Flow lookup within SoftNP, can either yield a hit or miss for an existing session

  - If hit → session already exists → processing jumps to the first member of dispatch array: input QoS.

  - If it fails the lookup → it's a new connection → we continue processing against other security checks and build the flow

SoftNP Overview

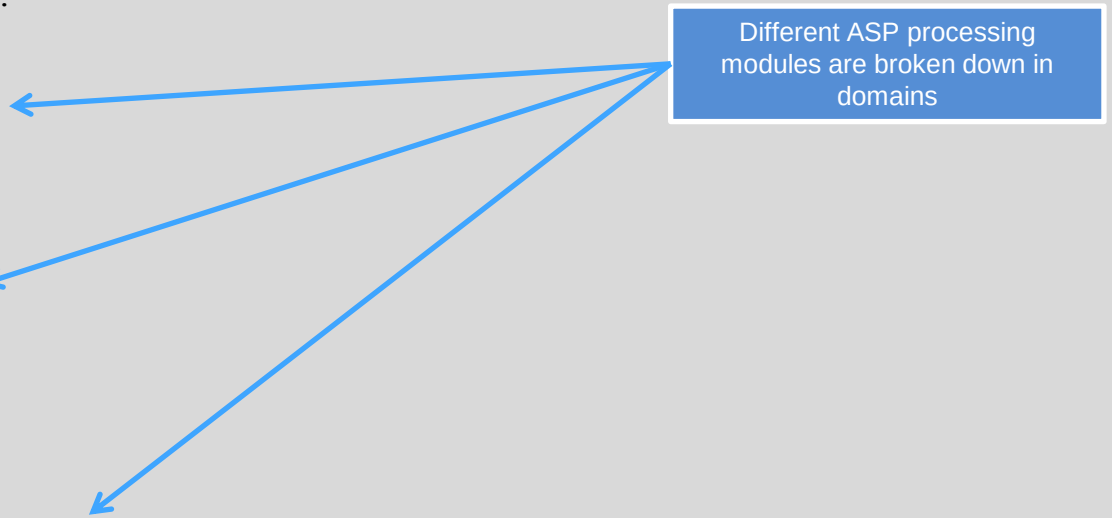
- SoftNP was designed as a multicore safe data path with a well defined API (NP-API) – software emulation of NP ASIC
- Control Point code is not multicore, and should not directly access SoftNP data structures and vice-versa. All communication is done through the NP-API queues
- Designed from the ground up to be high-performance and easily expandable
- Our processing architecture consists of Data Path (DP) and Control Path (CP)
- SoftNP is a base of our Data Path infrastructure that handles packets in either:
  - Fast Path (already existing sessions)
  - Or Slow Path (new sessions establishment)

How is DP Organised for Processing?

- DP is organised into multilevel ASP tables
- Each processing step touches on the appropriate ASP table – packet tracer will show this clearly
- ASP table content is pushed from CP during configuration/session establishment or dynamically updated (eg. ARP adjacencies)
- Mostly used for processing of new flows, but also by CP for routing, socket or interface lookups

ASP Tables

```
ASA (config)# sho asp table classify domain ?

exec mode commands/options:
  aaa-acct
  aaa-auth
  aaa-user
  accounting
  app-redirect
  arp
  autorp
  backup interface CLI
  capture
  conn-nailed
  conn-set
  ctcp
  debug-icmp-trace
  decrypt
  dhcp
  dynamic-filter
  eigrp
  encrypt
  established
  filter-activex
  filter-ftp
<--- More --->
```



ASP Tables – Example: Routing

```
ASA# show route
…
C    17.0.1.0 255.255.255.0 is directly connected, inside
C    10.0.0.0 255.0.0.0 is directly connected, outside
S*   0.0.0.0 0.0.0.0 [1/0] via 10.48.66.1, outside

ASA# show asp table routing | ex identity
in   17.0.1.0          255.255.255.0   inside
in   10.0.0.0          255.0.0.0       outside
in   0.0.0.0           0.0.0.0         outside
out  255.255.255.255   255.255.255.255 inside
out  17.0.1.0          255.255.255.0   inside
out  224.0.0.0         240.0.0.0       inside
out  255.255.255.255   255.255.255.255 outside
out  10.0.0.0          255.0.0.0       outside
out  224.0.0.0         240.0.0.0       outside
out  0.0.0.0           0.0.0.0         via 10.48.66.1, outside
```

Routing table as visible/configured on CP

Routing table as visible/pushed down on DP

in and out are used by different processing paths: Slow Path/Mid Path

The Identity Interface

- This is a special interface which represents the box itself or the control point
- Packets routed to the identity interface are to-the-box traffic
- Packets sent from-the-box enter the data-path with a source interface set to the identity interface
- The identity interface allows the to/from-the-box traffic to have all of the same features and checks as thru-the-box

```
ASA# show asp table interfaces
…
Soft-np interface 'identity' is up
    context single_vf, port np/identity, mtu 65535
            vlan <None>, Not shared, seclvl 101
            140724603457764 packets input, 4326 packets output
            flags 0x30

ASA# show asp table routing | in identity
in   255.255.255.255 255.255.255.255 identity
in   10.1.1.100      255.255.255.255 identity
in   10.1.2.100      255.255.255.255 identity
```

Broadcast destination – allows for punting to CP

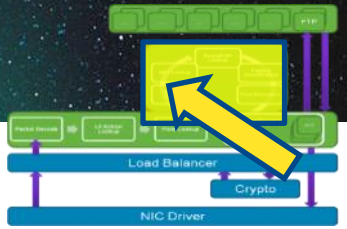
/32 interfaces – allow for punting to CP

Packet Tracer

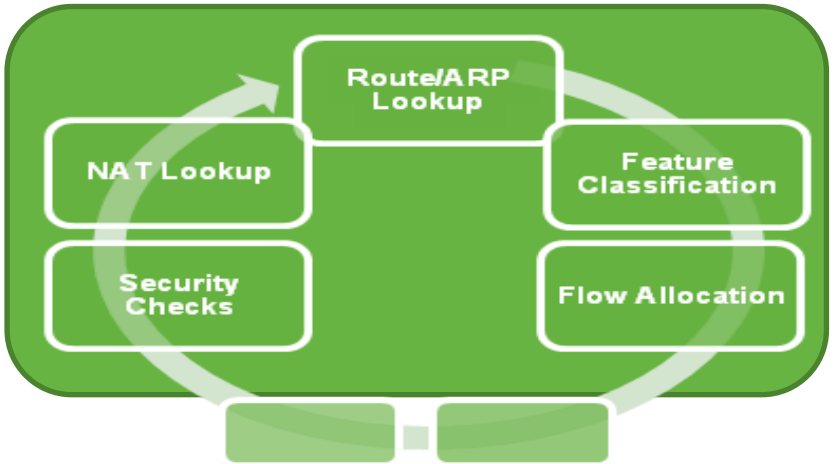
- Packet Tracer is known as a simulation tool to trace a packet as it moves through various processing steps/modules
- Little known fact is that this tool can be used in cooperation with Packet Capture, to trace the real/captured traffic
- It clearly shows which processing tables (ASP tables) packet touched, and why was it dropped/diverted: a powerful troubleshooting tool
- It may not show all of the ASP tables and possible drop reasons

```
ASA# packet-tracer input outside tcp 192.0.1.200 2222 192.0.1.100 80 detailed
...
Phase: 3
Type: ACCESS-LIST
Subtype:
Result: DROP
Config:
Implicit Rule
Additional Information:
 Forward Flow based lookup yields rule:
 in  id=0xd808dec0, priority=11, domain=permit, deny=true
        hits=1, user_data=0x5, cs_id=0x0, flags=0x0, protocol=0
        src ip/id=0.0.0.0, mask=0.0.0.0, port=0
        dst ip/id=0.0.0.0, mask=0.0.0.0, port=0, dscp=0x0
        input_ifc=outside, output_ifc=any

Drop-reason: (acl-drop) Flow is denied by configured rule
```

Flow Creation



- Every passed packet is part of a flow
- Main point of policy enforcement
- Decide on further flow processing (inspect)
- NAT in 8.3+ happens **after** routing



Processing Steps for New Flows

- NAT Untranslate/Routing
- ACL check
- IP options checks
- NAT
- RPF checks
- Additional security checks
- Crypto checks
- Bootnet filter check
- TCP intercept
- IPSec spi validation
- Create flow

First DP Processing Step: Un-NAT/Routing

- The first processing step inside the DP for new sessions is to perform Un-NAT for the destination of a flow!
- By performing it early in the flow lookup, we are able to achieve the following:
  - Get the Real IP address of the destination server
  - Get the routing information, pointing to the exact outgoing interface
  - Deliver this result to the awaiting modules (ACL, NAT, etc.) depending on it

First DP Processing Step: Un-NAT/Routing

```
ASA# sho nat detail
Manual NAT Policies (Section 1)
1 (inside) to (outside) source static inside_real inside_translated
    translate_hits = 0, untranslate_hits = 0
    Source - Real: 192.168.2.100/32, Mapped: 192.0.1.100/32


ASA# packet-tracer input outside tcp 192.0.1.200 2222 192.0.1.100 80 detailed
Phase: 1
Type: FLOW-LOOKUP
Subtype:
Result: ALLOW
Config:
Additional Information:
Found no matching flow, creating a new flow

Phase: 2
Type: UN-NAT
Subtype: static
Result: ALLOW
Config:
Additional Information:
NAT divert to egress interface inside
Untranslate 192.0.1.100/80 to 192.168.2.100/80
```

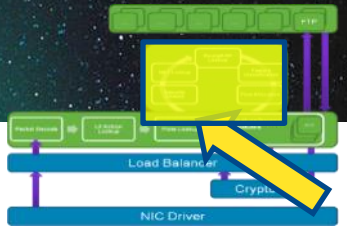
Address as visible on the Outside

Step 1 → lookup miss = new flow processing!

We Un-NAT the destination and we find out the Real IP/Routing Interface for it

This address is delivered to ACL check on Outside interface

Second DP Processing Step: Access-List



- General form: Extended, Standard, Webtype
- Starting 8.3, several features expect “Real IP” in ACLs
- Object-groups are used to configure large ACLs
  - Network, ICMP-Type, Service, Protocol
  - Change to object-group automatically reflects in ACL using it
- Starting 8.3, Global Access Rules can be used:
  - Applied to ingress traffic on all interfaces → processed after interface rules
  - Not replicated on all the interfaces in ASP → save memory space
  - May work as a last resort policy instead of implicit deny rules
  - Can be defined in conjunction with interface ACLs
  - Does NOT affect control traffic, output traffic, or DACLs
- Applies only to through traffic. If the ACL is intended for to-box, apply “control-plane” option at the end

```
access-group 101 in interface inside
access-group 202 out interface inside
access-group 303 in interface inside control-plane
access-group 404 global
```

Features that use Real IP

For Your Reference

- Access-group
- MPF
- WCCP redirect ACL
- Botnet traffic filter
- AAA match access-list

Soft-NP ACL Rule Structure

- All ACL rules are downloaded into soft-np via classify rules – permit domain
- During a lookup time, it returns an ID that is cached by CP

```
ASA(config)# sho run access-list
access-list outside extended permit ip any host 192.168.2.100
ASA(config)# sho run access-group
access-group outside in interface outside

ASA(config)# sho asp table classify domain permit

in  id=0xd80a09a8, priority=13, domain=permit, deny=false
                hits=1, user_data=0xd6e65140, cs_id=0x0, use_real_addr, flags=0x0, protocol=0
                src ip/id=0.0.0.0, mask=0.0.0.0, port=0
                dst ip/id=192.168.2.100, mask=255.255.255.255, port=0, dscp=0x0
                input_ifc=outside, output_ifc=any

ASA# packet-tracer input outside tcp 192.0.1.200 2222 192.0.1.100 80 detailed
...

Phase: 3
Type: ACCESS-LIST
Subtype: log
Result: ALLOW
Config:
access-group outside in interface outside
access-list outside extended permit ip any host 192.168.2.100
Additional Information:
Forward Flow based lookup yields rule:
in  id=0xd80a09a8, priority=13, domain=permit, deny=false
        hits=0, user_data=0xd6e65140, cs_id=0x0, use_real_addr, flags=0x0, protocol=0
        src ip/id=0.0.0.0, mask=0.0.0.0, port=0
        dst ip/id=192.168.2.100, mask=255.255.255.255, port=0, dscp=0x0
        input_ifc=outside, output_ifc=any
```

Cached rule ID returned during flow setup time

Real IP as delivered by Un-Nat module if part of flow

Unified ACLs in 9.0

- The object-groups used in ACLs can now be mixed containing both IPv4 and IPv6 entries:

```
object-group network inside_networks
   network-object 192.168.1.0 255.255.255.0
   network-object 2001:abcd::/64
```

- IPv4 and IPv6 tuples are unified under the same ACL
  - All types of combinations: IP44,IP46, IP64 and IP66
- Two new keywords, ‘any4’ and ‘any6’ are introduced
  - Keywords ‘any4’ and ‘any6’ are equivalent of ‘any’ used in IPv4 or IPv6 ACLs in 8.4
  - On 9.0 software, an ACL defined as “permit any any” is equivalent to:

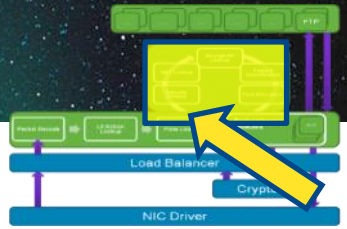
  permit ip any4 any4 + permit ip any4 any6 + permit ip any6 any4 + permit ip any6 any6
- Single ACL, containing both IPv4 and IPv6 will be applied to an interface
  - ‘ipv6 access-list’ command is deprecated
- During migrations to 9.0 software, all ACLs containing ‘any’ keyword, will be migrated to ‘any4’

ACL Limits

- Number of ACEs is limited only by memory to which they expand
- Each ACE uses a minimum of 212 bytes of RAM
- However, maximum performance may decrease (typically 10-15%) as you reach or exceed the Max Recommended ACEs.
- High number of ACEs may affect both session establishment and throughput
- **`show access-list | include elements`** will tell you how many ACEs are present

	5505	5510	5520	5540	5550	5580	5585/5512	ASA SM/5515-5555
Max Recommended ACEs	25k	80k	200k	500k	700k	750k	500k / 750k 1 / 2 million	2 million
Tested ACEs		80k	300k	700k	700k	1 million+	500k / 750k 1 / 2 million	2 million
Max Observed (from customers)					2.74 million	2.77 million		

Flow Creation: NAT in 8.3 and Later

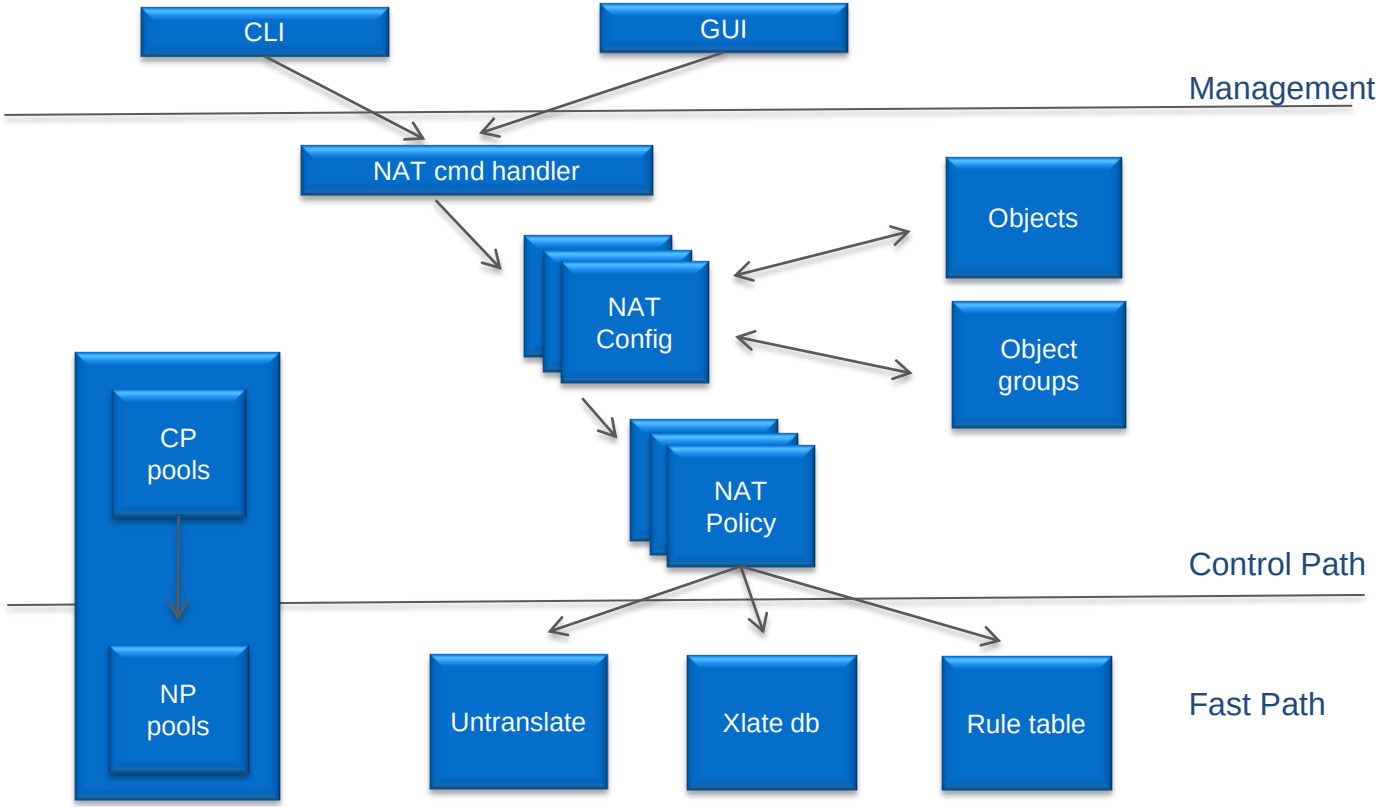


- Single translation rule table
- Manual NAT
  - Allows for bi-directional translation
  - Allows to specify both Source and Destination NAT within a single line
  - More flexibility in creating NAT rules (one-to-one, one-to-many, many-to-many, many-to-one)
- Automatic NAT
  - Single rule per object
  - Useful for less complex scenarios
  - Lexicographic order of statements within “auto” section
- Manual “after” NAT
  - Specifically positioned at the end of the NAT processing table

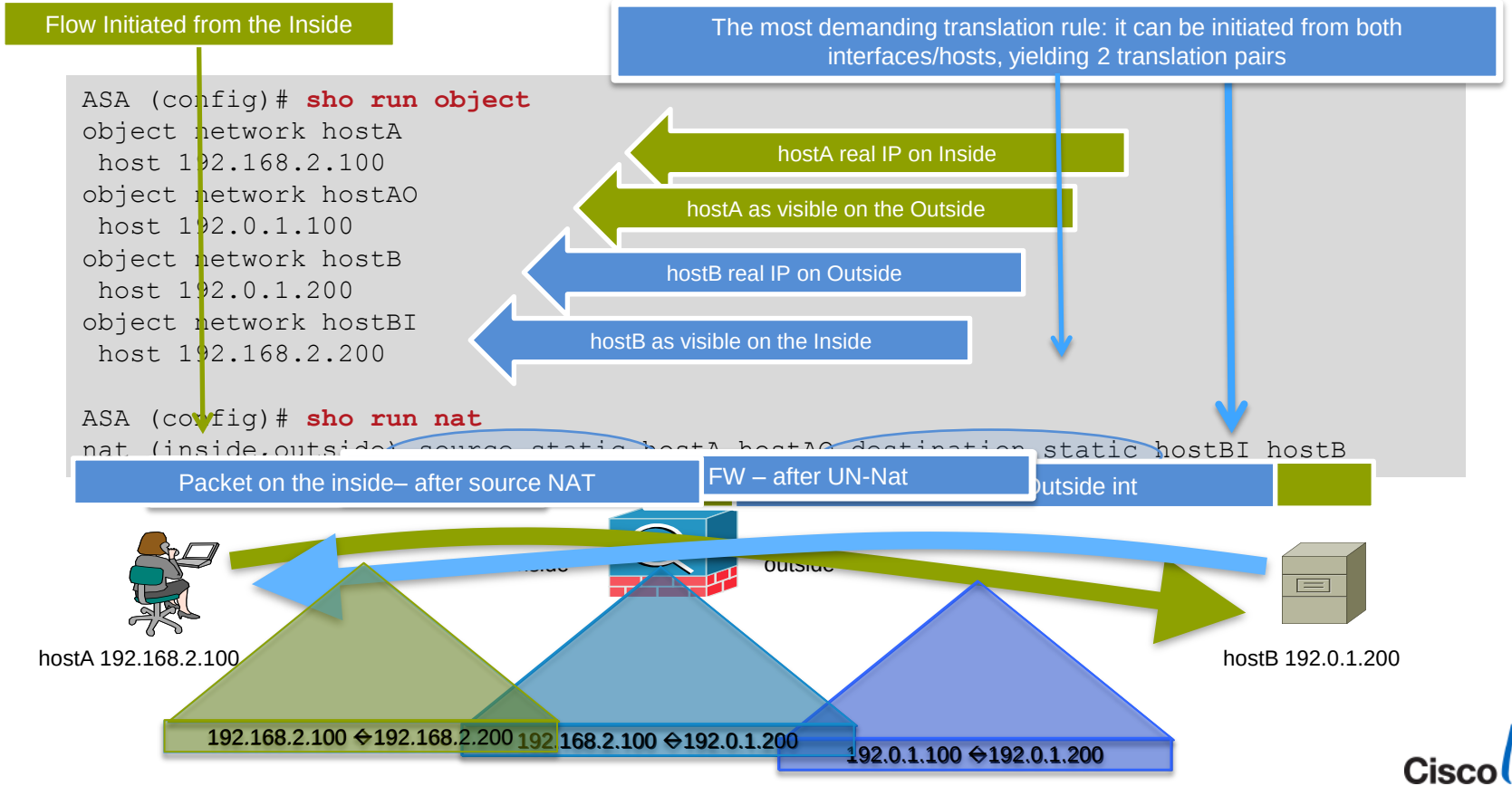
NAT in 8.3 and Later

- Rules are processed in order (like ACEs inside and ACL) – caching of those rules' IDs inside DP structures assures of this
- Rule ID is used to change it's place inside the list
- Manual NAT rules are always processed first
- Within Manual NAT rules list, only the order matters – it doesn't take into account dynamic/static nature of the statement
- Auto rules are processed next

NAT Architecture



NAT Configuration Constructs



NAT Rule Tables

- This is how it translates down to NAT rule constructs on a DP level:

```
ASA(config)# sho asp table classify domain nat
Input Table
in   id=0xd88acd60, priority=6, domain=nat, deny=false
                hits=0, user_data=0xd86ea2a8, cs_id=0x0, use_real_addr, flag
                src ip/id=192.168.2.100, mask=255.255.255.255, port=0
                dst ip/id=192.0.1.200, mask=255.255.255.255, port=0, dscp=0x0
                input_ifc=inside, output_ifc=outside
in   id=0xd88ad500, priority=6, domain=nat, deny=false
                hits=0, user_data=0xd80b72f8, cs_id=0x0, use_real_addr, flags=0x0, protocol=0
                src ip/id=192.0.1.200, mask=255.255.255.255, port=0
                dst ip/id=192.168.2.100, mask=255.255.255.255, port=0, dscp=0x0
                input_ifc=outside, output_ifc=inside
```

This ID is valid only for translations initiated from the Inside – it will be cached as part of the xlate

This ID is valid for translations initiated from the Outside – bidirectional xlate

- Xlate/Untranslate database:

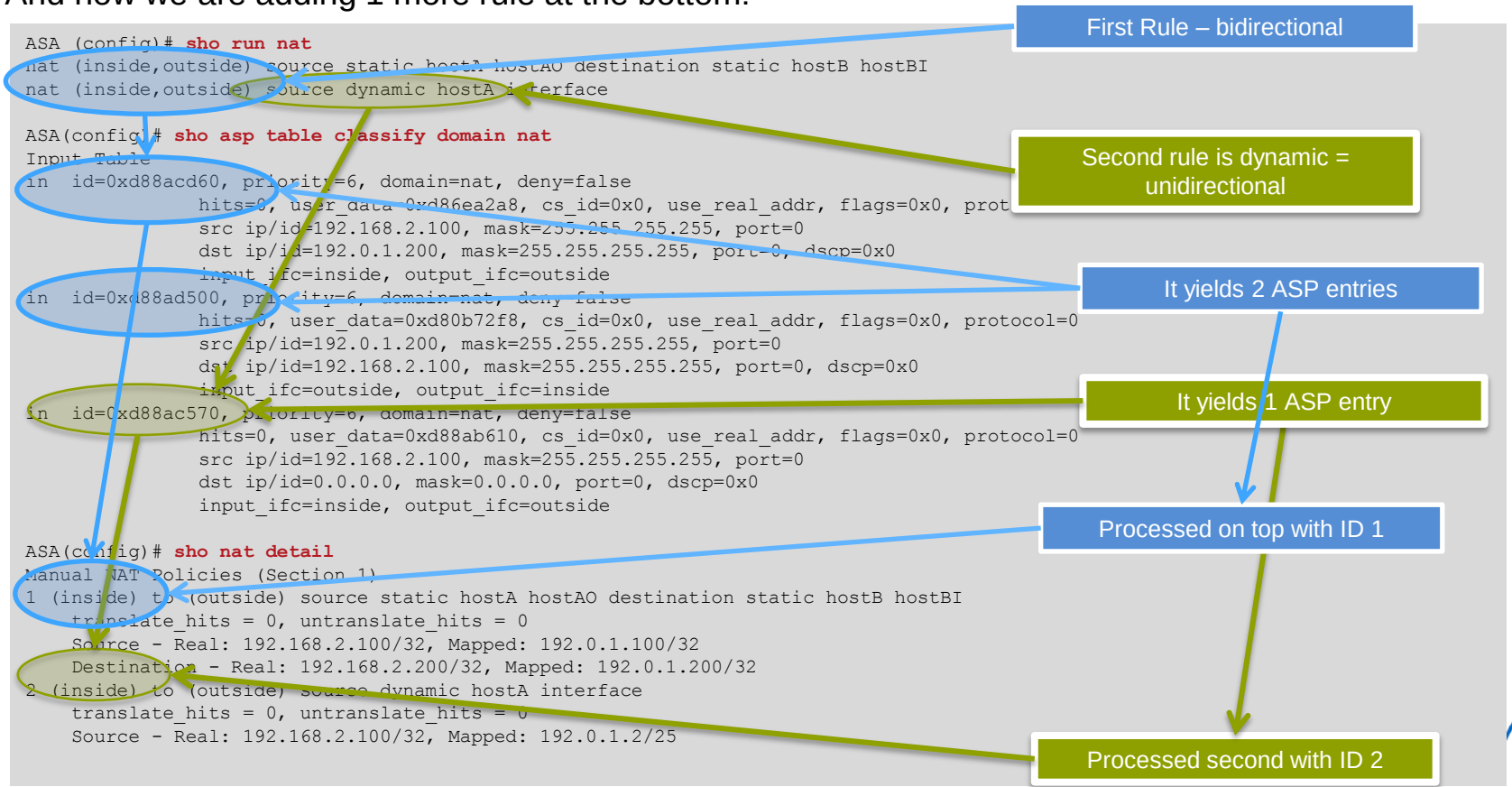
```
ASA(config)# sho nat detail
Manual NAT Policies (Section 1)
1 (inside) to (outside) source static hostA hostAO destination static hostB hostBI
    translate_hits = 0, untranslate_hits = 0
    Source - Real: 192.168.2.100/32, Mapped: 192.0.1.100/32
    Destination - Real: 192.168.2.200/32, Mapped: 192.0.1.200/32

ASA(config)# sh xlate
2 in use, 2 most used
Flags: D - DNS, i - dynamic, r - portmap, s - static, I - identity, T - twice
NAT from inside:192.168.2.100 to outside:192.0.1.100
    flags sT idle 0:14:33 timeout 0:00:00
NAT from outside:192.0.1.200 to inside:192.168.2.200
    flags sT idle 0:14:33 timeout 0:00:00
```

ID representing the order of processing – it will be the first element in the ASP table as well.

Cached rules will be used for flows in Fast Path processing

NAT Ordering

- And now we are adding 1 more rule at the bottom:



NAT Performance Considerations

- Identity or Static NAT is best for high performance
- Dynamic PAT and NAT require more CPU cycles and mostly affect connection setup rate
- PAT may affect throughput as well – usually up to 18%
- Additional performance drop if logging translation creation/tear up

Logging Messages

- Syslog messages may be initiated by Data Path, but are generated by Control Path
  - DP informs CP via a call upon DP-CP queue: there is a log to be sent
  - CP will pick up ‘material’ from the queue and put in a syslog form
  - CP will then send the syslog out, using DP
  - In case of UDP-based syslog this can be partly offloaded to DP

- This eats up more CPU cycles and more bandwidth
  - Most impact from logging conn creation events or SNMP polling
  - Use Netflow instead of Syslogs where applicable
  - Netflow minimises per-packet overhead by bundling data
  - Binary data takes up less space than ASCII strings

Case Study: Excessive Logging

For Your Reference

```
logging enable
logging buffered debugging
logging console debugging
logging trap debugging
logging history debugging
logging host inside 192.168.1.10
logging host inside 192.168.1.11
logging host DMZ 192.168.2.121

snmp-server host inside 192.168.1.10
snmp-server host inside 192.168.1.11
snmp-server host DMZ 192.168.2.121

flow-export destination inside 192.168.1.10
flow-export destination inside 192.168.1.11
flow-export destination DMZ 192.168.2.121
```

4 logging destinations (buffer, console, SNMP, and syslog)

+

3 syslog servers

+

3 SNMP servers

+

3 Netflow collectors

+

4 messages per PAT connection (over 550 bytes)

```
%ASA-6-305011: Built dynamic TCP translation from inside:192.168.1.101/4675 to
outside:172.16.171.125/34605
%ASA-6-302013: Built outbound TCP connection 3367663 for outside:
(198.133.219.25/80) to inside:192.168.1.101/4675 (172.16.171.125/
%ASA-6-302014: Teardown TCP connection 3367663 for outside:198.13
inside:192.168.1.101/4675 duration 0:00:00 bytes 1027 TCP FINs
%ASA-6-305012: Teardown dynamic TCP translation from inside:192.168.1.101/4675 to
outside:172.16.171.125/34605 duration 0:00:30
```

1 connection:
32 syslog messages
26+ packets sent
100K conn/sec:
2.8Gbps

Case Study: Logging Optimisation

For Your Reference

Not logging to buffer unless troubleshooting

Console logging is a bottleneck (low rate)

Using minimum number of syslog servers and Netflow collectors

Do not duplicate syslogs and Netflow data

Reduce severity level for syslogs

Send only certain syslogs as SNMP traps

Not all SNMP servers need to receive traps

```
logging enable
logging flow-export-syslogs disable
logging list FAILOVER message 104003
logging trap errors
logging history FAILOVER
logging host inside 192.168.1.10
logging host DMZ 192.168.2.121
snmp-server host inside 192.168.1.10
snmp-server host DMZ 192.168.2.121 poll
flow-export destination inside 192.168.1.10
flow-export destination DMZ 192.168.2.121
```

Queuing at the CP Level

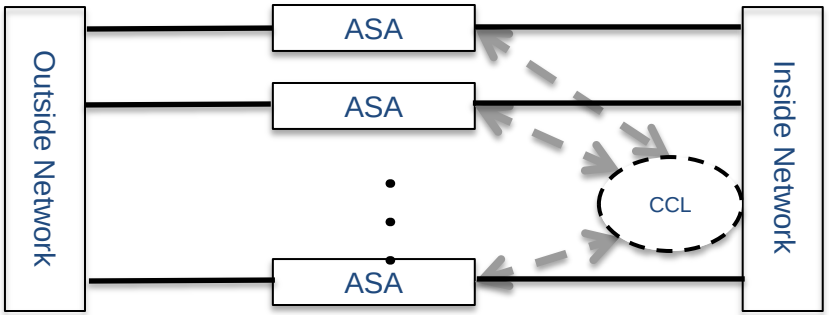
- Fast Path punts to CP by using event queues
- This event queue is split into categories and displays per-punted-feature counter
- Queues have upper limit, but no rate-limit
- Overloading one of them, may have significant effect on the DP to CP communication

```
ASA5585# sho asp event dp-cp
DP-CP EVENT QUEUE                    QUEUE-LEN  HIGH-WATER
Punt Event Queue                             0           0
Identity-Traffic Event Queue                 0           0
General Event Queue                          0           1
Syslog Event Queue                           0         310
Non-Blocking Event Queue                     0           0
Midpath High Event Queue                     0           2
Midpath Norm Event Queue                     0           3
SRTP Event Queue                             0           0

EVENT-TYPE          ALLOC ALLOC-FAIL ENQUEUED ENQ-FAIL  RETIRED 15SEC-RATE
midpath-norm          911          0      911        0      911           0
ips-cplane        3596848          0  3596848        0  3596848           0
arp-in                253          0      253        0      253           0
syslog            2511111       1321   249790   212123  2511111           0
```

Indicates the load on CP exceeded its processing capability

Clustering

- Clustering = connecting multiple ASAs to form a single firewall, transparent to users and scaling in a sub-linear fashion
- Capable of handling heavy asymmetric flows without performance penalty
- Positioned for Data Centre environments scaling to more than 100 Gbps firewall
- One unit is designated as a Master and the rest are Slave units
- A dedicated interface for Cluster Control Link (CCL)
  - Keepalive/CP/DP messages are sent over this link



Clustering Basics

- Can achieve a scaling factor of 0.7, assuming
  - N+1 redundancy
  - Consistent hashing algorithm to redirect packets within cluster
- Clustering is supported on all 5585-X and 5580 platforms
- The interfaces in a cluster of ASAs can be configured in either:
  - Layer-2 mode:
    - ASA interfaces are grouped together in an Etherchannel bundle
    - A switch uses Etherchannel load balancing mechanisms to send traffic between ASAs where all ASA units share a single system IP and system MAC, and appear as a single gateway in the network
  - Layer-3 mode:
    - Each interface on the ASA has it's own IP address and MAC address
    - A router can use PBR (Policy Based Routing) or ECMP (Equal Cost MultiPath routing) to balance traffic between ASAs.

Layer 2 Mode Configuration

ASA – Data Interface
interface GigabitEthernet0/1
channel-group 1 mode active

interface GigabitEthernet0/0
channel-group 2 mode active

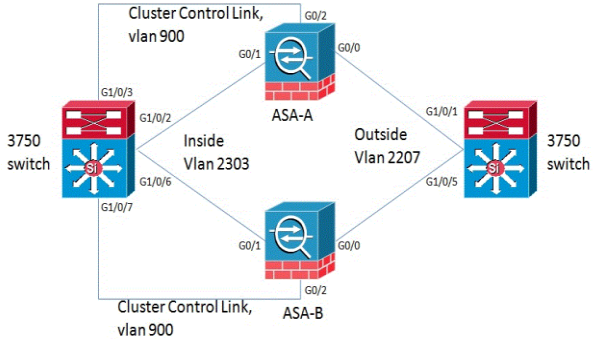
interface Port-channel1
port-channel span-cluster

interface Port-channel1.2303
mac-address aaaa.bbbb.aaaa
vlan 2303
nameif inside
ip address 10.1.1.1

interface Port-channel2
port-channel span-cluster

interface Port-channel2.2207
mac-address aaaa.cccc.aaaa
vlan 2207
nameif outside
ip address 10.2.2.1

Switch Configs For Cluster control link:
For int GigabitEthernet1/0/3 and GigabitEthernet1/0/7 :
switchport
switchport access vlan 900
switchport mode access
spanning-tree portfast



Switch Configurations:
For Inside -
int Port-channel100
switchport
switchport trunk encapsulation dot1q
switchport trunk allowed vlan 2303
switchport mode trunk

For int GigabitEthernet1/0/2 and int GigabitEthernet1/0/6 -
switchport
switchport trunk allowed vlan 2303
switchport mode trunk
channel-group 100 mode active

Switch Configurations:
For Outside-
int Port-channel200
switchport
switchport trunk encapsulation dot1q
switchport trunk allowe vlan 2207
switchport mode trunk

For int GigabitEthernet1/0/1 and int GigabitEthernet1/0/5 -
switchport
switchport trunk allowed vlan 2207
switchport mode trunk
channel-group 200 mode active

Clustering Flow Types

- The state for each connection is centralised on a unit called the **‘owner’**
  - All the packets belonging to the same connection must be processed by the owner
- If packets for a certain connection land on a non-owner unit, they are forwarded to owner over Cluster Control Link (CCL)
- The first ASA to receive traffic for a TCP/UDP connection (non-inspection) is a designated owner
- Connection state is backed up on a different ASA unit called the **‘director’**
  - This is a similar method used for stateful replication in Failover setups
- Director (unique) selected by performing hash on 5-tuple for connection
  - Any unit would get the same director by performing a hash over a 5-tuple – chord hashing
  - Any unit can get the owner for a connection by querying the director

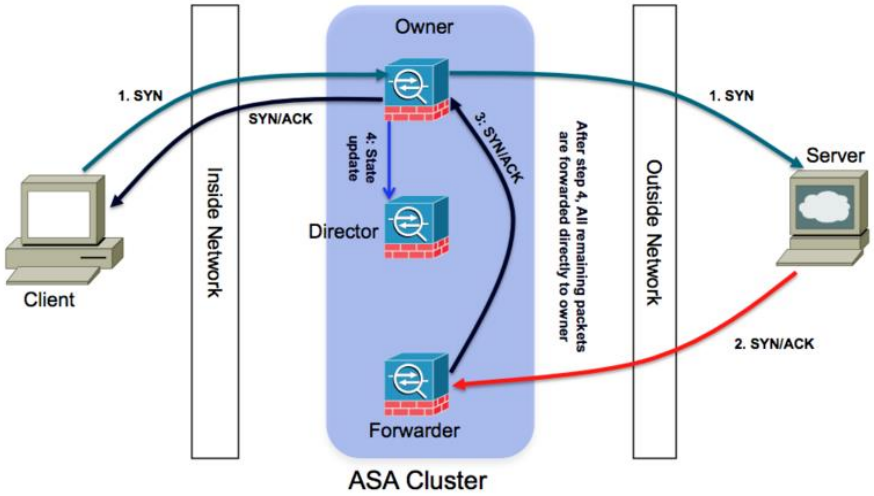
Types of Flows

- Forwarding flow
  - The unit that handles the first packet will be an owner (dependent on the Etherchannel hashing)
  - Since it finds out who the owner is, it will create a forwarding flow → forward packets to the owner using CCL
  - Flags – U/I/O/B: normal connection flags
  - Forwarder flows is not subscribed to state updates (the unit is simply forwarding to owner)
  - Short-lived flows (eg. DNS, ICMP) do not have forwarding flows

```
ASA1# show conn detail
...
TCP outside: 192.168.0.100/22 inside: 192.168.1.131/35481,
    flags  y, idle 0s, uptime 23m37s, timeout -, bytes 0, cluster
sent/rcvd bytes 38923511/0, cluster sent/rcvd total bytes 40694839/0,
owners (2,255)

    flags  Y, idle 10s, uptime 10m37s, timeout -, bytes 0, cluster
sent/rcvd bytes 21992775/3680, cluster sent/rcvd total bytes
23178895/3888, owners (1,255)
```

TCP Connection Build-up

1. When a SYN packet reaches the owner, the owner creates a flow, encoding owner information into SYN cookie
   - Provided that the sequence randomisation is ON
2. Owner unit forwards a SYN packet to server
   - If sequence randomisation is OFF for the cluster, owner unit needs to update a director about the sate of this flow immediately
3. SYNACK packet originates from server and is delivered to forwarder unit
4. Forwarder unit decodes owner information from SYN cookie and creates forwarding flow to direct packets to owner unit
   - If the sequence randomisation is OFF for the cluster, forwarder unit will have to query a director to find the owner for this packet
5. Owner unit sends state update to director unit, and forwards SYNACK to client
6. After above process, TCP connection is established and backed by director unit



Effects on Other Features

- The features that are supported on the ASA are either **centralised** or **distributed** in clustering
- Centralised features require a task to be completed by a cluster Master

Centralised

- Inspect (DCERPC, ESMTP, IGMP, NetBios, PPTP, Radius, RSH, SNMP, SUNRPC, TFTP, XDMCP)
- IGMP, PIM, L3 Multicast Data Traffic
- L2 Dynamic Routing
- VPN: L3/IKEv1 and L3/IKEv2, VPN management access

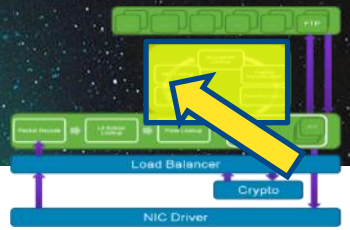
Distributed

- DNS
- NAT
- TCP intercept, others…..

Unsupported

- Inspect CTIQBE, WCCP, SIP, Skinny, WAAS, RTSP, MGCP, RAS, H323/H325, GTP
- Failover
- VPN RA, IPSec pass-through, VPN Load Balancing
- NAC, TLS Proxy
- DHCP CLient, Server, Proxy

Drops in ASP

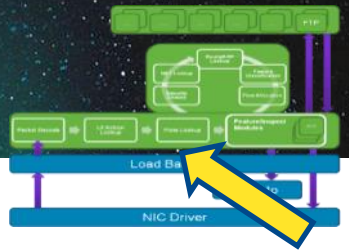


- Accelerated security path counts all drops
- Not all drops are visible with the exact reason in the packet tracer or in syslog
- Frame drops: per packet, Flow drops: per flow
- Drops in ASP can be captured as well (**`capture type asp-drop`**)
- Drop counters are documented in the command reference, under **`show asp drop`**

```
ASA# show asp drop

Frame drop:
  Flow is denied by configured rule (acl-drop)                          42
  First TCP packet not SYN (tcp-not-syn)                               438
  TCP RST/FIN out of order (tcp-rstfin-ooo)                            199
  Expired flow (flow-expired)                                          280
  Interface is down (interface-down)                                     4
  Dropped pending packets in a closed socket (np-socket-closed)        522

Last clearing: Never

Flow drop:
  NAT reverse path failed (nat-rpf-failed)                               4
  SSL handshake failed (ssl-handshake-failed)                           14
  SSL received close alert (ssl-received-close-alert)                    1
```

Troubleshooting NAT

Processing steps for existing flows

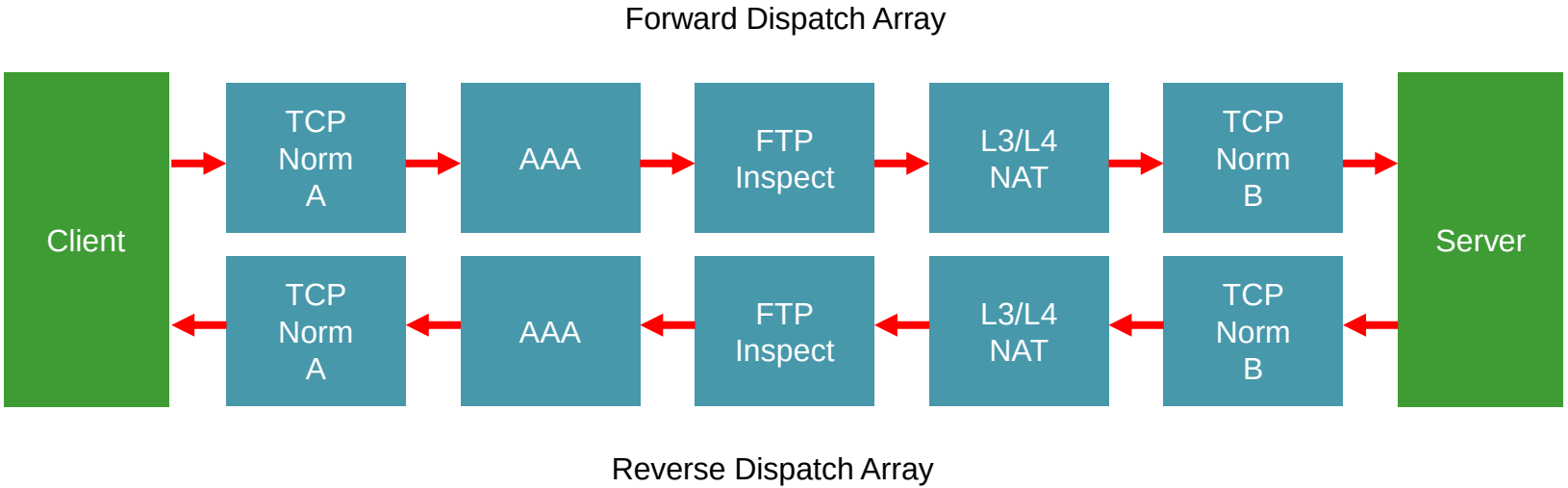


- Input QOS
- IPSec Tunnel Processing
- TCP Intercept Processing
- TCP Security Engine
- IP Option Processing
- NP Inspect Engine Processing (ICMP/DNS/RTP/RTCP)
- DNS Guard
- Pinhole Processing
- Multicast processing
- CSC Module Processing (optional)
- Inspection Engine Processing/AAA punts/IPsec over TCP punts
- IPSec NAT-T Processing
- Decrypt
- Address Update and Checksum Adjustments
- TCP Security Engine
- IPS - AIP Module processing (optional)
- Adjacency Look-up if necessary
- Output QOS
- Encrypt
- Fragment
- Output Capture
- Output L2 ACL
- Queue processing and Transmit

Dispatch Array

- How do I process this flow after setup?
- Every flow has a dispatch array of channels, with one for every action: Compiled information from ACL, MPF, etc.
- Array of actions for a flow is compiled during flow setup/creation
- As the flow has 2 wings (one for each direction), actions on an array usually have 2 directions in order to preserve flow consistency: Forward and Reverse array
- Some of those actions may be dynamically disabled during the life of a flow, for instance an inspect that concludes that it doesn't need to see any more data may remove itself
- Packets may be injected into any point in the array by the CP or inspects

Dispatch Array Example



Flow Management

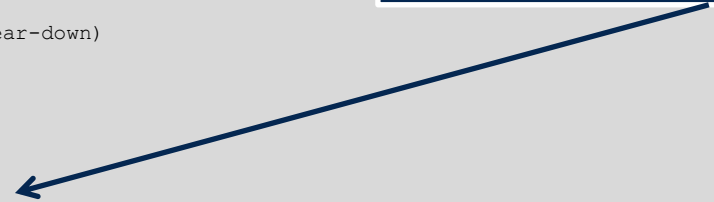
```
ASA# sho conn detail
…
TCP dmz:192.168.1.2/22 inside:192.168.2.10/56661,
    flags UIO, idle 59s, uptime 59s, timeout 1h0m, bytes 2934

ASA# sho conn long
..
TCP dmz:192.168.1.2/22 (192.168.1.2/22) inside:192.168.2.10/36217 (192.168.1.1/52395), flags UIO, idle 11s, uptime 11s, timeout
1h0m, bytes 2934

ASA# sho local-host
Interface inside: 2 active, 4 maximum active, 0 denied
local host: <192.168.2.10>,
    TCP flow count/limit = 1/unlimited
    TCP embryonic count to host = 0
    TCP intercept watermark = unlimited
    UDP flow count/limit = 0/unlimited

  Xlate:
    TCP PAT from inside:192.168.2.10/43473 to dmz:192.168.1.1/7233 flags ri idle 0:00:01 timeout 0:00:30

  Conn:
    TCP dmz 192.168.1.2:22 inside 192.168.2.10:43473, idle 0:00:01, bytes 2934, flags UIO
Interface dmz: 2 active, 4 maximum active, 0 denied
local host: <192.168.1.2>,
    TCP flow count/limit = 1/unlimited
    TCP embryonic count to host = 0
    TCP intercept watermark = unlimited
    UDP flow count/limit = 0/unlimited

  Conn:
    TCP dmz 192.168.1.2:22 inside 192.168.2.10:43473, idle 0:00:01, bytes 2934, flags UIO
Interface outside: 2 active, 4 maximum active, 0 denied
```

Control Plane Processing and Application Inspection

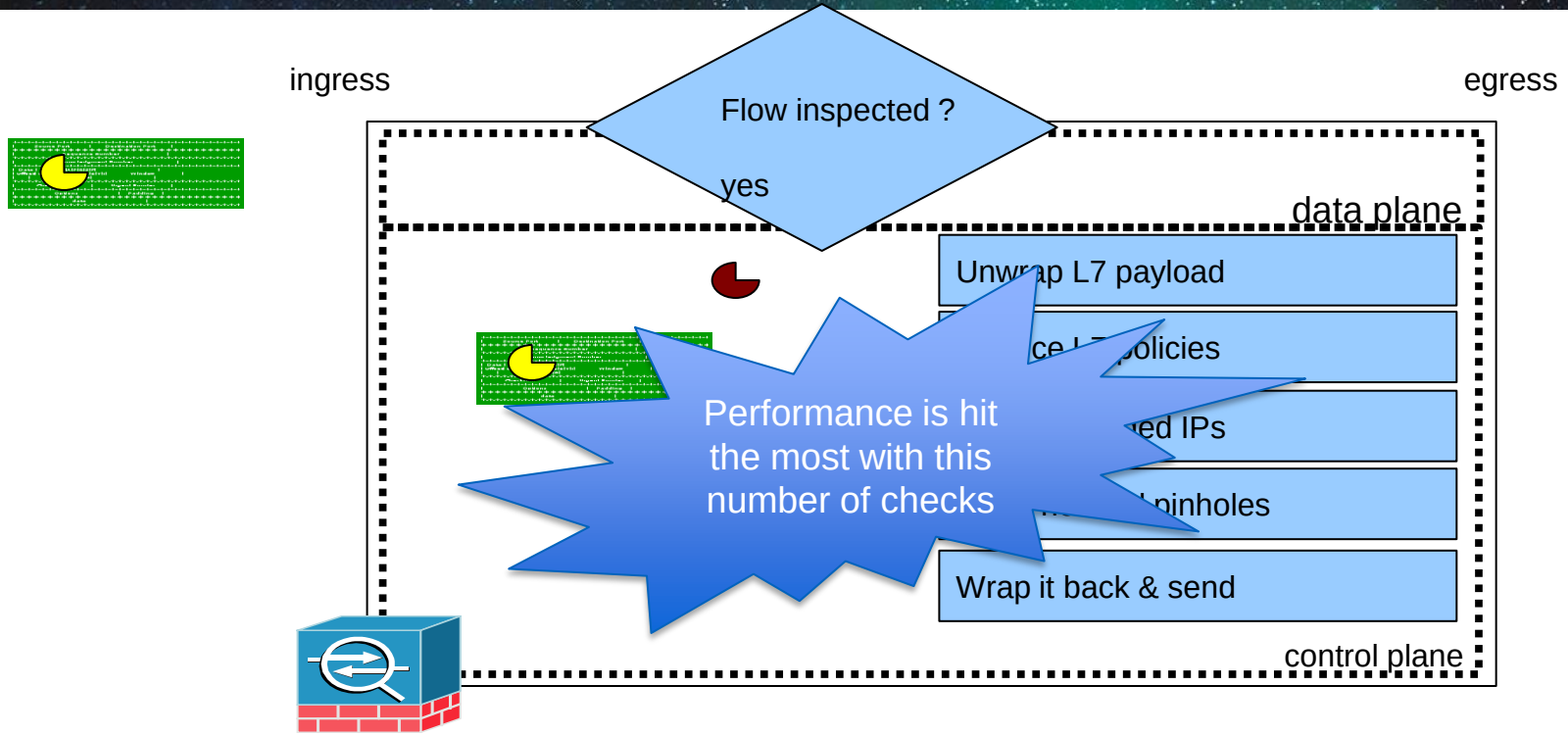
DP/CP Accelerated Features

```
ASA5585# sho asp multiprocessor accelerated-features
MultiProcessor accelerated feature list:
        Access Lists
        DNS Guard
        Failover Stateful Updates
        Flow Operations(create, update, and tear-down)
        Inspect HTTP URL Logging
        Inspect HTTP (AIC)
        Inspect IPSec Pass through
        Inspect ICMP and ICMP error
        Inspect RTP/RTCP
        IP Audit
        IP Fragmentation & Re-assembly
        IPSec data-path
        MPF L2-L4 Classify
        Multicast forwarding
        NAT/PAT
        Netflow using UDP transport
        Non-AIC Inspect DNS
        Packet Capture
        QOS
        Resource Management
        Routing Lookup
        Shun
        Syslogging using UDP transport
        TCP Intercept
        TCP Security Engine
        TCP Transport
        Threat Detection
        Unicast RPF
        WCCP Re-direct
Above list applies to routed, transparent, single and multi mode.
```

Features outside this list can not be processed in DP, and need to be punted to CP – single thread!

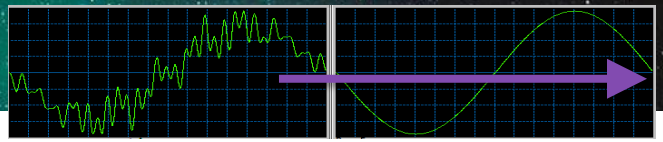


Application Inspection Engines



The flows are classified for inspection based on the configured service-policy.
The inspected TCP flows are usually subject to TCP normaliser before the inspection.

TCP Normaliser

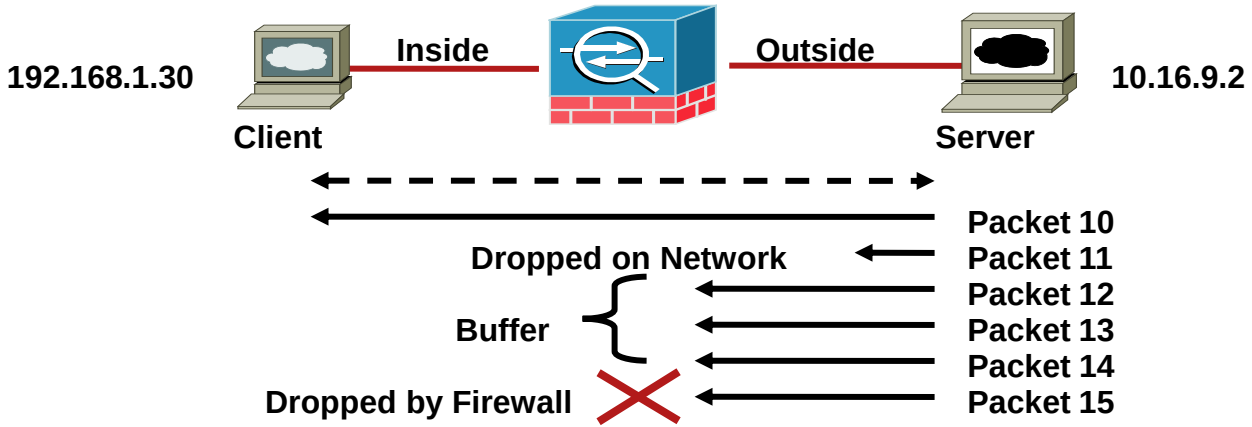


- Out-of-order TCP segments reduce performance
  - Re-assembly effort by transit devices and receiver
  - Possibility of dropped OOO packets calls for TCP retransmits → slow start
- For inspected and traffic sent to SSM, this causes a problem
  - ASA can put packets in order if needed (queuing)
  - The queue is kept per-flow – 3 packets by default
  - Buffering can be increased on ASA by using the **`queue-limit`** option under the **`tcp-map`**

```
ASA# sho asp drop frame tcp-buffer-full

TCP packet buffer full                                  90943
Last clearing: 06:39:06 UTC Jan 2 2012 by enable_15

ASA# sho asp drop frame tcp-buffer-timeout

TCP packet buffer full                                    902
Last clearing: 06:39:06 UTC Jan 2 2012 by enable_15
```

Case Study: Out-of-Order Packets

- Lots of out-of-order packets seen
- Default out-of-order buffer too small to hold
- Poor TCP throughput due to lot of retransmits



Case Study: Out-of-Order Packets

- How to detect

```
ASA# show asp drop
Frame drop:
  ...
  TCP packet buffer full                                  90943
  ...
```

- How to fix

```
access-list OOB-nets permit tcp any 10.16.9.0 255.255.255.0
!
tcp-map OOO-Buffer
     queue-limit 6
!
class-map tcp-options
   match access-list OOB-nets
!
policy-map global_policy
   class tcp-options
      set connection advanced-options OOO-Buffer
!
service-policy global_policy global
```

Case Study: Out-of-Order Packets

- How to verify?

Inspects applied and packets matching them

```
ASA# show service-policy

Global policy:
  Service-policy: global_policy
    Class-map: inspection_default
      Inspect: dns maximum-length 512, packet 92, drop 0, reset-drop 0
      Inspect: ftp, packet 43, drop 0, reset-drop 0
      Inspect: h323 h225, packet 0, drop 0, reset-drop 0
      Inspect: h323 ras, packet 0, drop 0, reset-drop 0
      Inspect: http, packet 562, drop 0, reset-drop 0
      Inspect: netbios, packet 0, drop 0, reset-drop 0
      Inspect: rsh, packet 0, drop 0, reset-drop 0
      Inspect: rtsp, packet 0, drop 0, reset-drop 0
      Inspect: skinny, packet 349, drop 0, reset-drop 0
      Inspect: esmtp, packet 0, drop 0, reset-drop 0

    Class-map: tcp-options
      Set connection policy:
      Set connection advanced-options: OOB-Buffer
        Retransmission drops: 0              TCP checksum drops : 0
        Exceeded MSS drops  : 0              SYN with data drops: 0
        Out-of-order packets: 2340           No buffer drops    : 0
```

No more normaliser buffer drops

HTTP Inspect

For Your Reference

- HTTP Inspect without HTTP-map
  - Logs URLs into syslogs
  - Helps AAA authen/author (if configured)
  - Helps URL filtering (if configured)
  - Basic protocol sanity checks
- HTTP Inspect with HTTP-map
  - Parse the HTTP headers fully
  - Monitor violations
  - Enforce L7-based actions
  - More resource-intensive

```
%ASA-6-302013: Built outbound TCP connection 764 for dmz:192.168.1.2/8080 (192.168.1.2/8080) to
inside:192.168.2.10/60886 (192.168.2.10/60886)
%ASA-5-304001: 192.168.2.10 Accessed URL 192.168.1.2:http://192.168.1.2:8080/
%ASA-6-302014: Teardown TCP connection 764 for dmz:192.168.1.2/8080 to inside:192.168.2.10/60886
duration 0:00:00 bytes 3778 TCP FINs
```

TCP Proxy

For Your Reference

- ASA gathers segmented TCP packets into a single buffer before inspection
  - ASA 8.4: IM, H.225, SIP, Skinny, RTSP, CTIQBE, SunRPC, DCERPC
  - Full TCP Proxy
  - Spoof ACK packets for the received data
  - Keeping the separate TCP Window for 2 wings of the flow and updates accordingly
- The total size of received TCP data and untransmitted TCP data is limited to 8192/64K bytes
  - This limit is imposed so that an inspection process can not over-subscribe system resources

VoIP Protocol Inspection

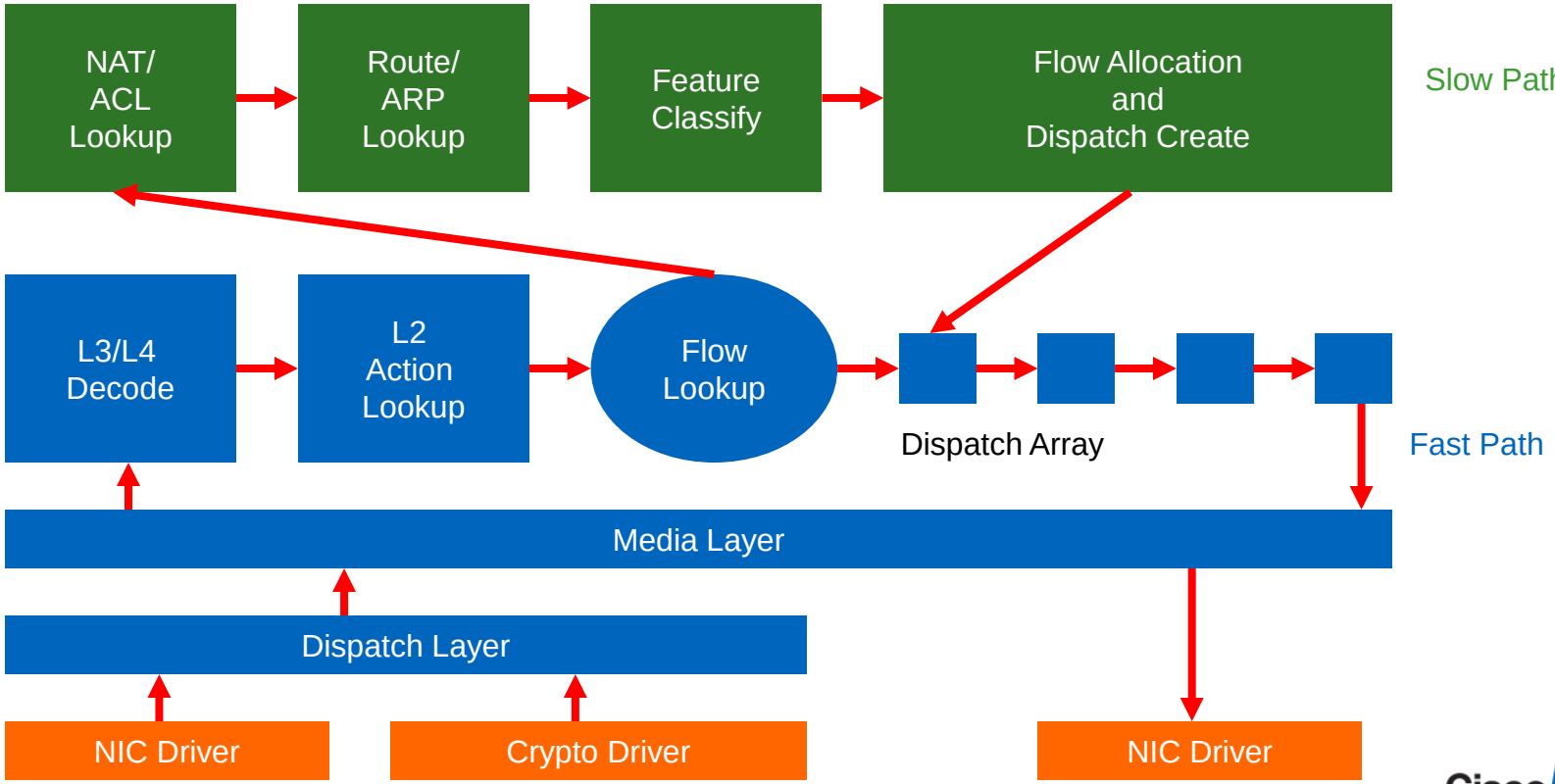
For Your Reference

- Most impact during phone registration and call setup
  - SIP performs better than Skinny due to less overhead
  - Limited advantage with multi-core due to single Control Path thread
- Media connections (RTP/RTCP) are handled in Data Path
  - High rate of small UDP datagrams
  - Control and associated media conns handled by same core
- Further registration and call setup rate hit with TLS Proxy
  - PKI module dependence




Closing Remarks

A Day in the Life of a Packet Diagram



Knowing Your Network

- Know your device to optimise performance and avoid problems
- Avoid congestion at early stages of packet processing
  - Take care of the interface and memory queues
- Optimise for balanced load between cores
  - Single flow will always end up on the same core
  - Use of Clustering is an additional layer of load balancing
- Chose Fast Path accelerated features over Control Path for best performance
  - Where necessary, use appropriate modules/appliances: WSA, ASA-CX, IPS, etc.
- Selectively apply features

How Do We West Performance?

- Maximum Throughput is achieved with
  - UDP packets of large size (less overhead)
  - Moderate numbe[illegible]f connections so that all c[illegible]es are leverag[illegible]
- Max PPS is achieved [illegible]mall UDP P[illegible] (64 b[illegible] moderate number of connections
- Max Connections is achiev[illegible] [illegible]ctions
- MAX CPS tests take regul[illegible] [illegible]mediate closure

TMIMIX traffic testing is based on Real World multi-protocol traffic profile and usually differs from those numbers

Network Location	SSI[illegible]	[illegible]X	5585-X P-40	ASA 5585-X SSP-60
Performance Max Firewall (Multi)	3 Gbp[illegible]	7 Gb[illegible]	12 Gbps	20 Gbps
Max Firewall Conns	1,000,00[illegible]	2,000,000	4,000,000	10,000,000
Max Conns/Second	65,000	140,000	240,000	350,000
PPS (64 byte)	1,500,000	3,200,000	6,000,000	10,500,000

Recommended Sessions

- BRKSEC-2021 - Firewall Architectures
- BRKSEC-3020 - Troubleshooting ASA Firewalls
- BRKSEC-3771 - Advanced Web Security Deployment with WSA and ASA-CX




Q & A

Complete Your Online Session Evaluation

Give us your feedback and receive a Cisco Live 2014 Polo Shirt!

Complete your Overall Event Survey and 5 Session Evaluations.

- Directly from your mobile device on the Cisco Live Mobile App
- By visiting the Cisco Live Mobile Site www.ciscoliveaustralia.com/mobile
- Visit any Cisco Live Internet Station located throughout the venue

Polo Shirts can be collected in the World of Solutions on Friday 21 March 12:00pm - 2:00pm

Learn online with Cisco Live!

Visit us online after the conference for full access to session videos and presentations.
www.CiscoLiveAPAC.com


CISCO
TM