# ASR1000 Introduction, troubleshooting and best practices

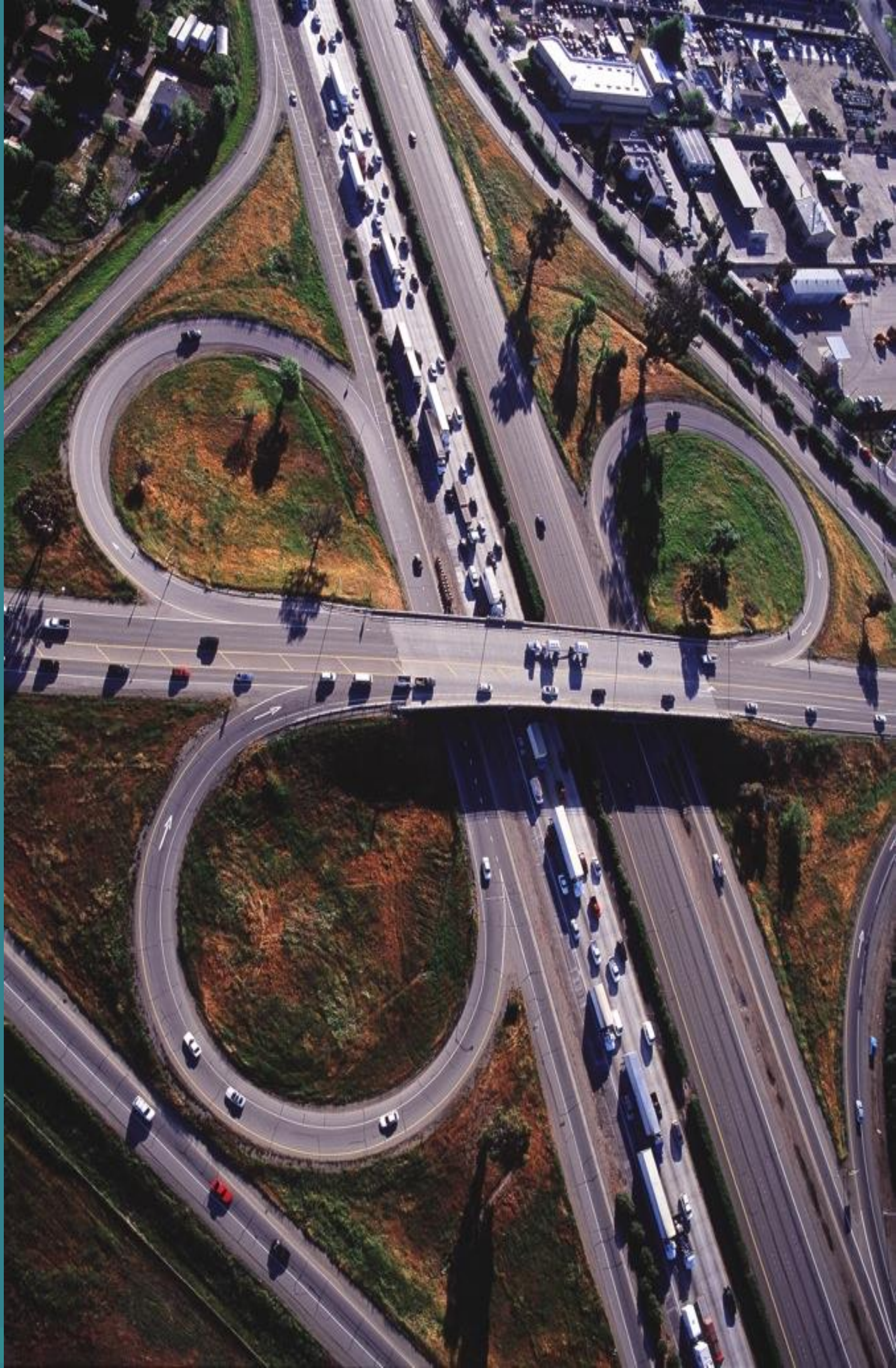
Wang Zhi Gang

March, 2015

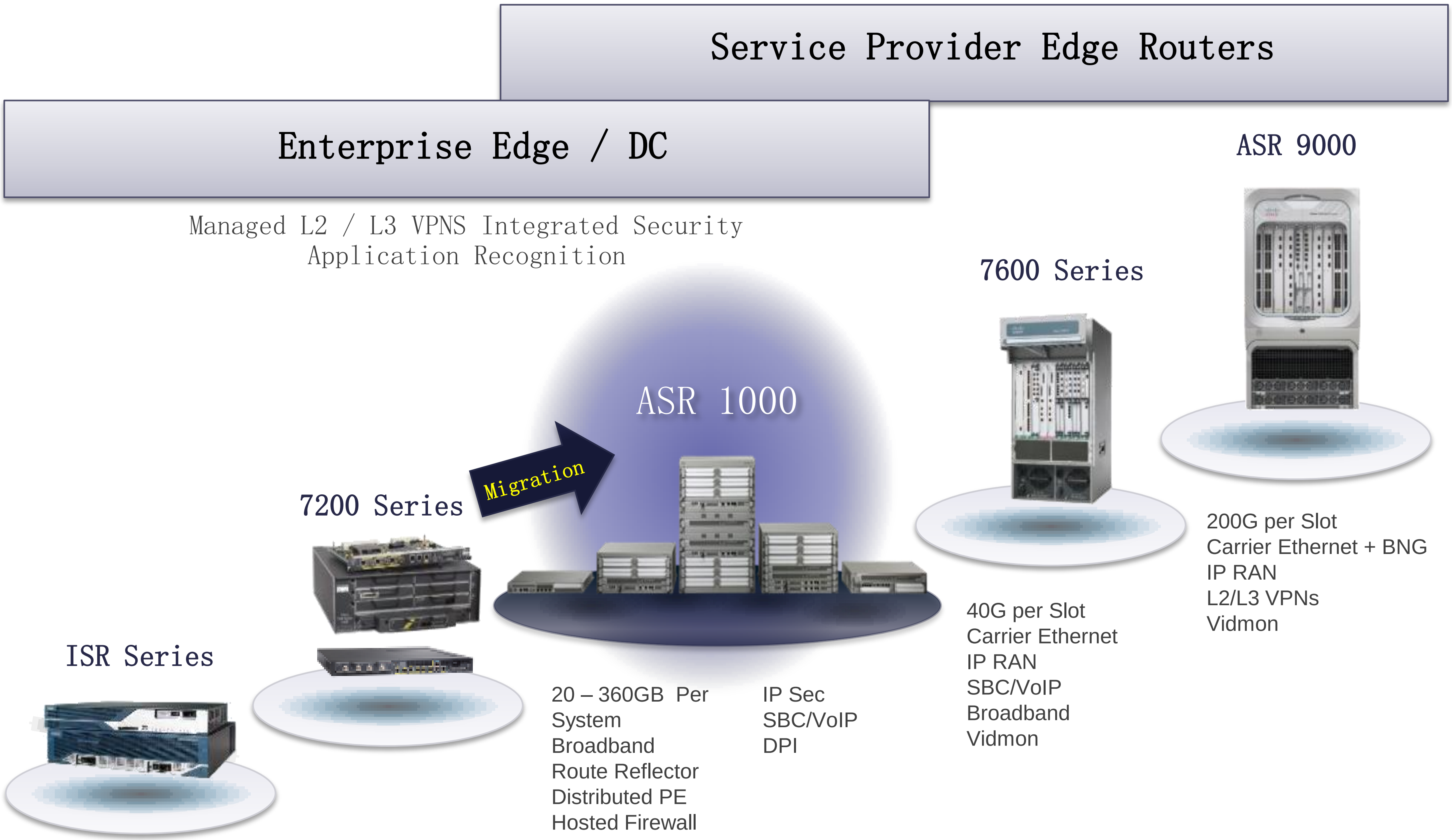
# Agenda

- ASR1000 introduction
- Hardware Component
- System Architecture
- ASR 1000 Software Architecture
- ASR 1000 Packet Flows
- IOS XE Releases and Packaging
- NAT+ZBW+HA
- DMVPN

# Introducing the ASR1000

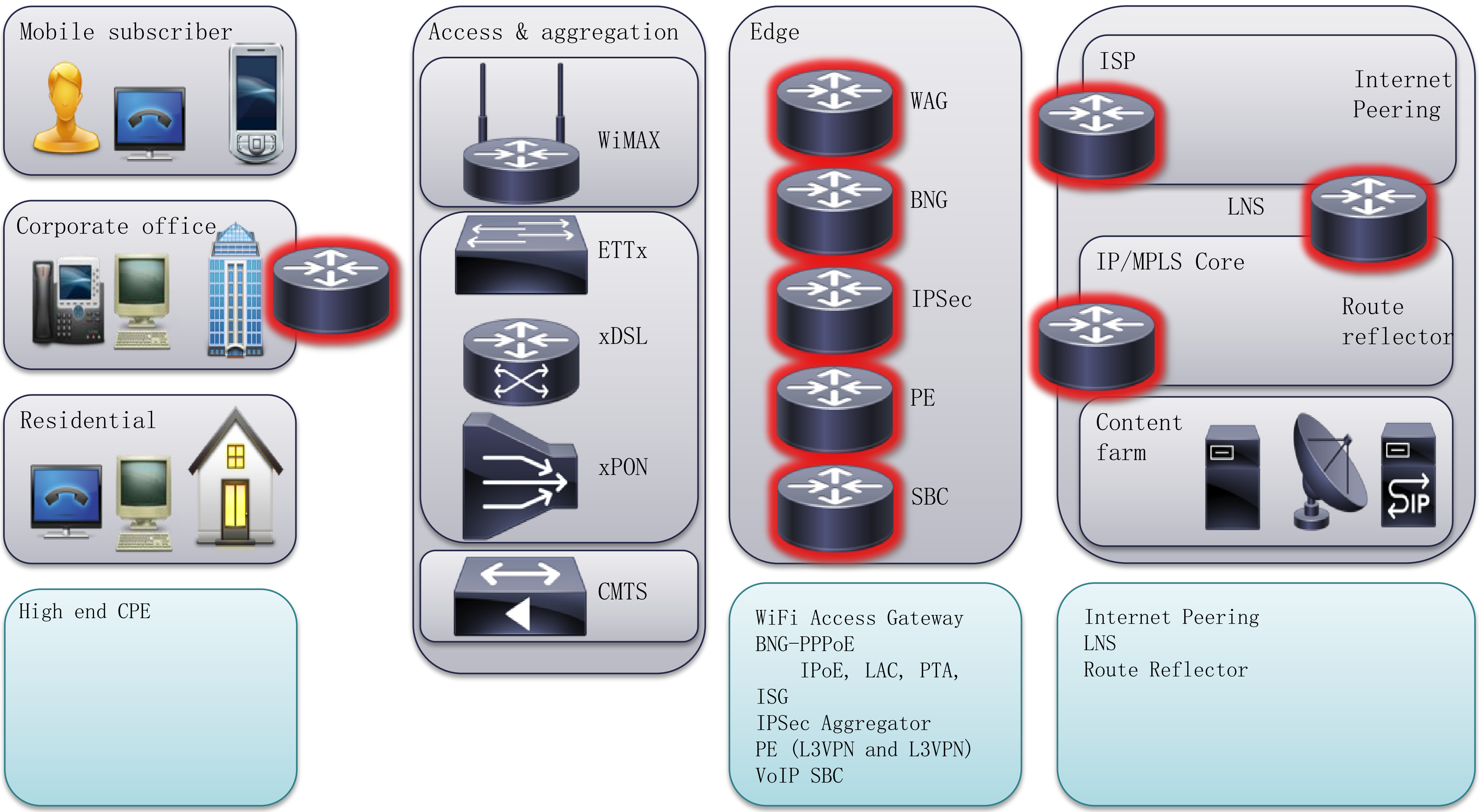


# Where the ASR1000 fits in the network



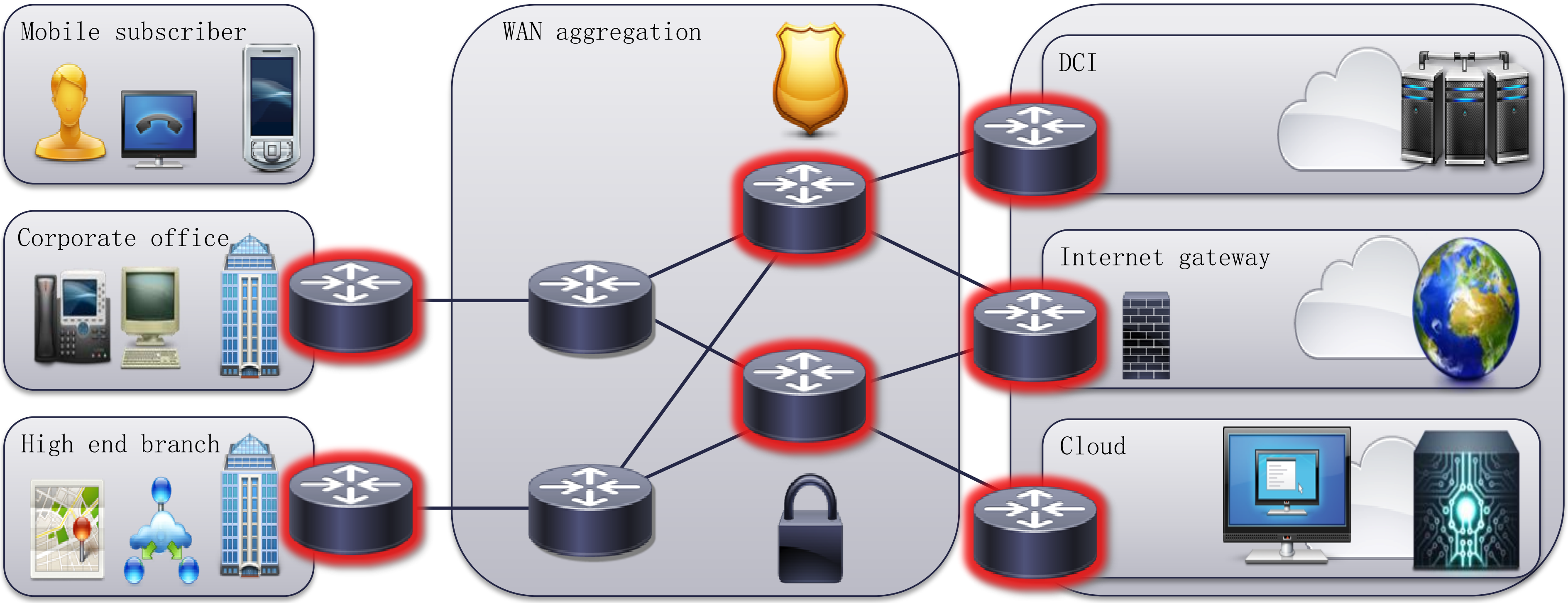

 

# ASR1000 in Service Provider next generation networks



# ASR1000 in enterprise deployments




High Speed CPE
High-end Branch

WAN Aggregation
IPSec
Internet Gateway

Data Center Interconnect
Internet gateway
Zone-Based Firewall
Cloud Services Edge

# Introducing the ASR1000 Series Routers

## Compact, Powerful Router

- Line-rate performance 2.5G to 200G+ with services enabled
- Hardware QoS engine with up 128K queues per ASIC
- Investment protection with modular engines, IOS CLI and SPAs for I/O

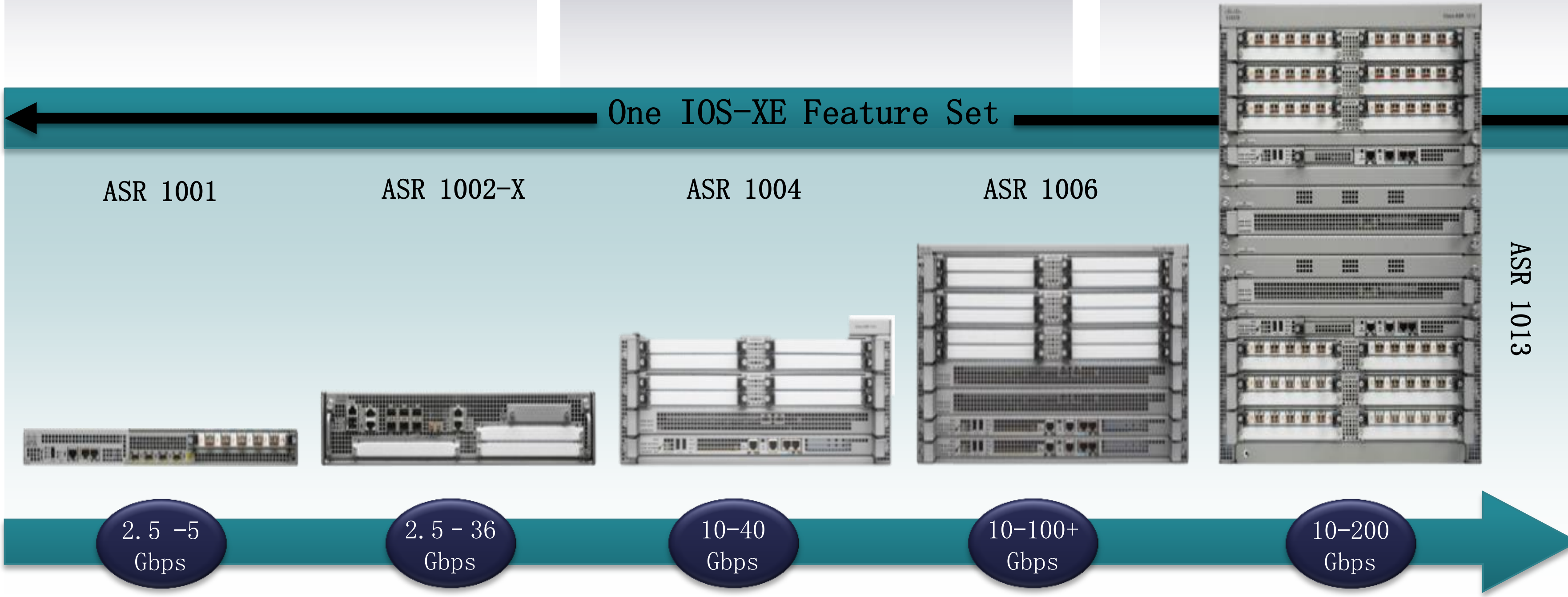
## Business-Critical Resiliency

- Fully separated control and forwarding planes
- Hardware and software redundancy
- In-service software upgrades

## Instant On Service Delivery

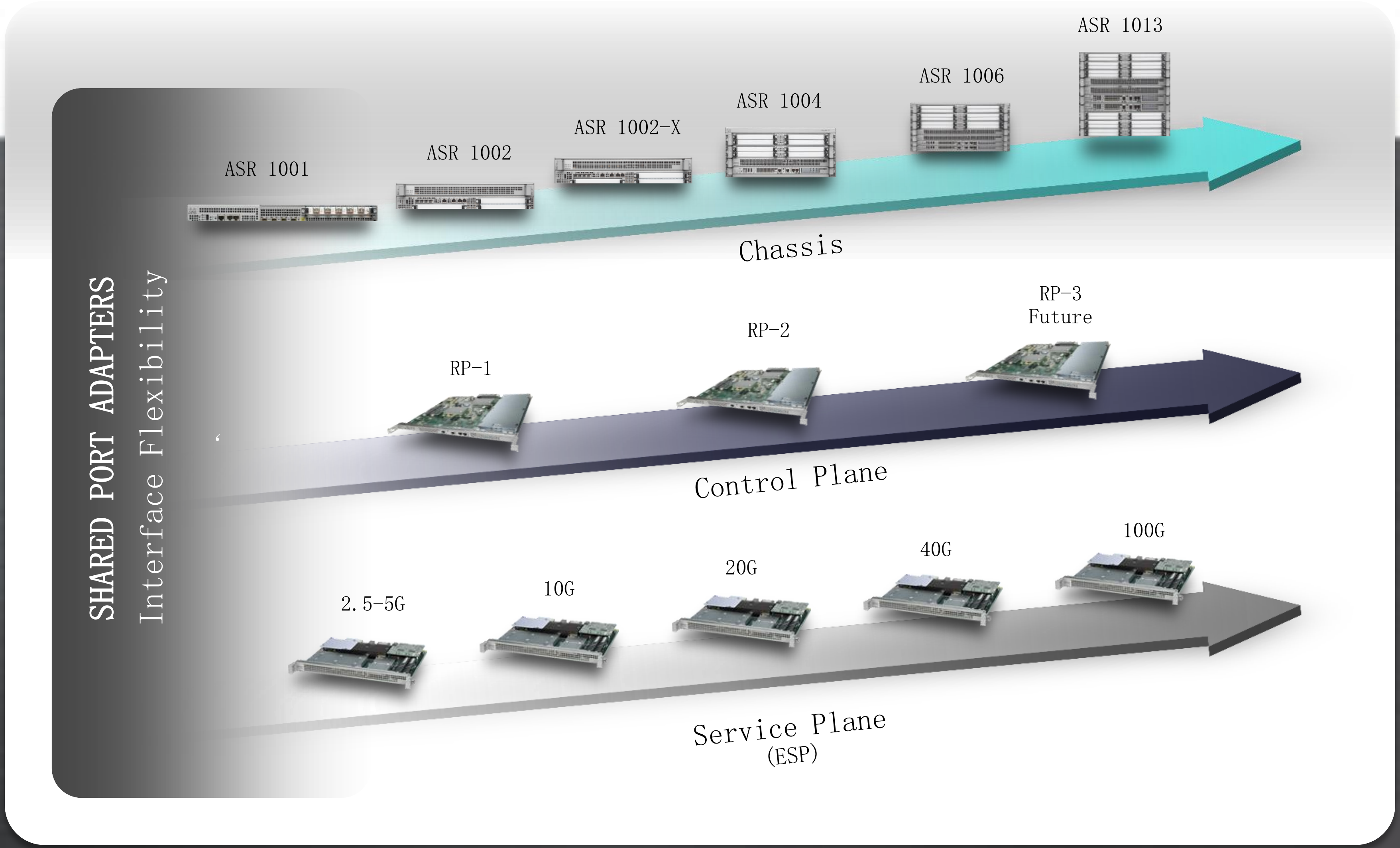
- Integrated firewall, VPN, encryption, NBAR, CUBE
- Scalable on-chip service provisioning through software licensing

One IOS-XE Feature Set

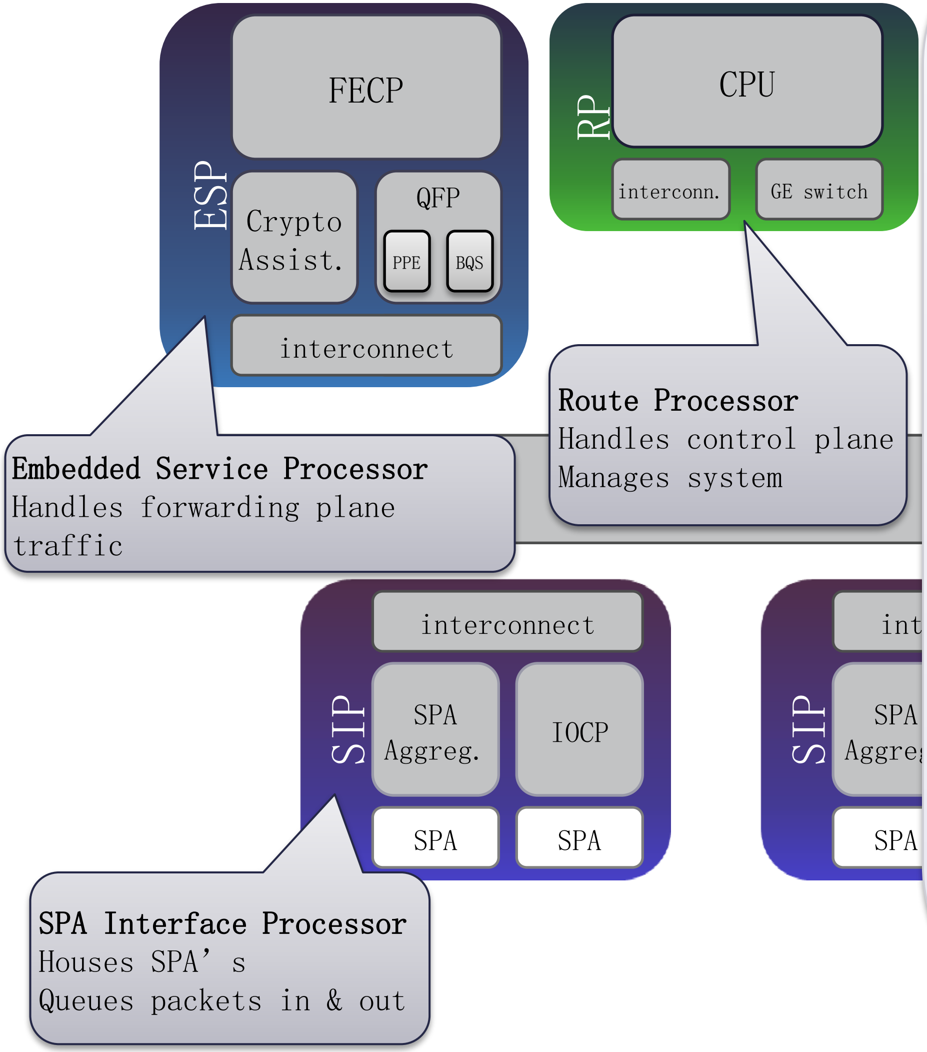


# Hardware Component

# ASR 1000 Series
# Investment Protection

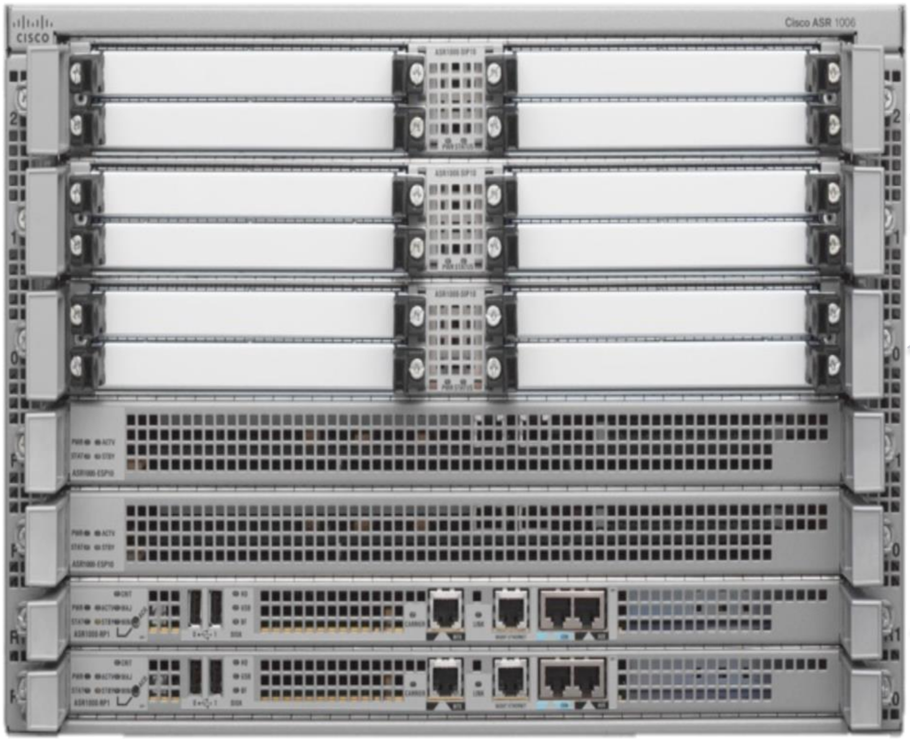


# ASR1000 building blocks




- Route Processor (RP)
  - Handles control plane traffic
  - Manages system
- Embedded Service Processor (ESP)
  - Handles forwarding plane traffic
- SPA Interface Processor (SIP)
  - Shared Port Adapters provide interface connectivity
- Centralized Forwarding Architecture
  - All traffic flows through the active ESP, standby is synchronized with all flow state with a dedicated 10-Gbps link
- Distributed Control Architecture
  - All major system components have a powerful control processor dedicated for control and management planes

# ASR1006




**SIP0, SIP1 and SIP2**
SPA interface access

**FP0 and FP1**
data plane processing

**RP0 and RP1**
control plane processing

# ASR1000 – power supplies

ASR1002

ASR1001

ASR1004

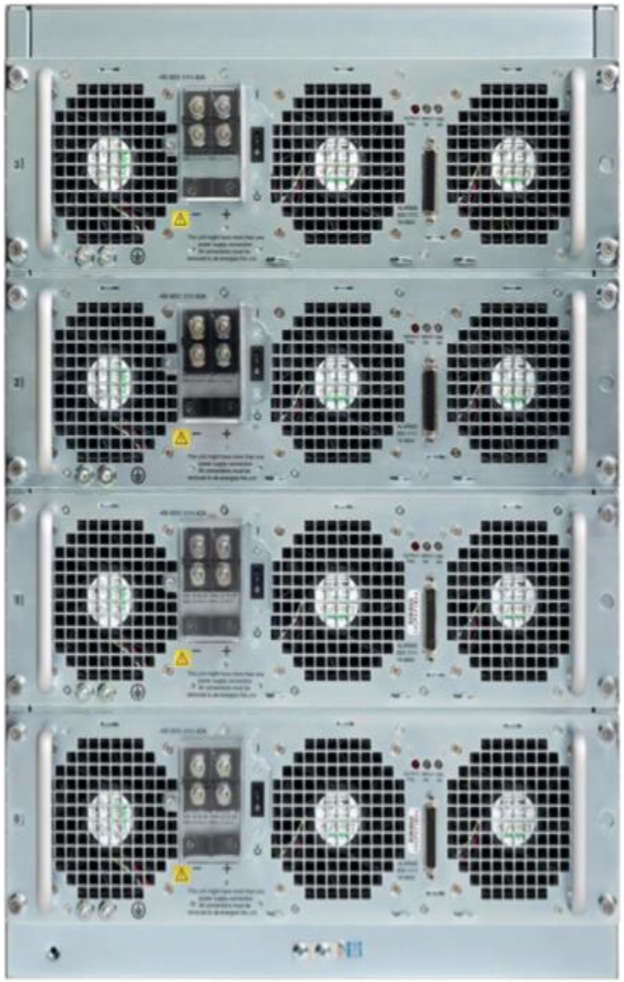
3x multispeed fan per PEM

2 PEM total

ASR1006

3x multispeed fan per PEM

2 PEM total



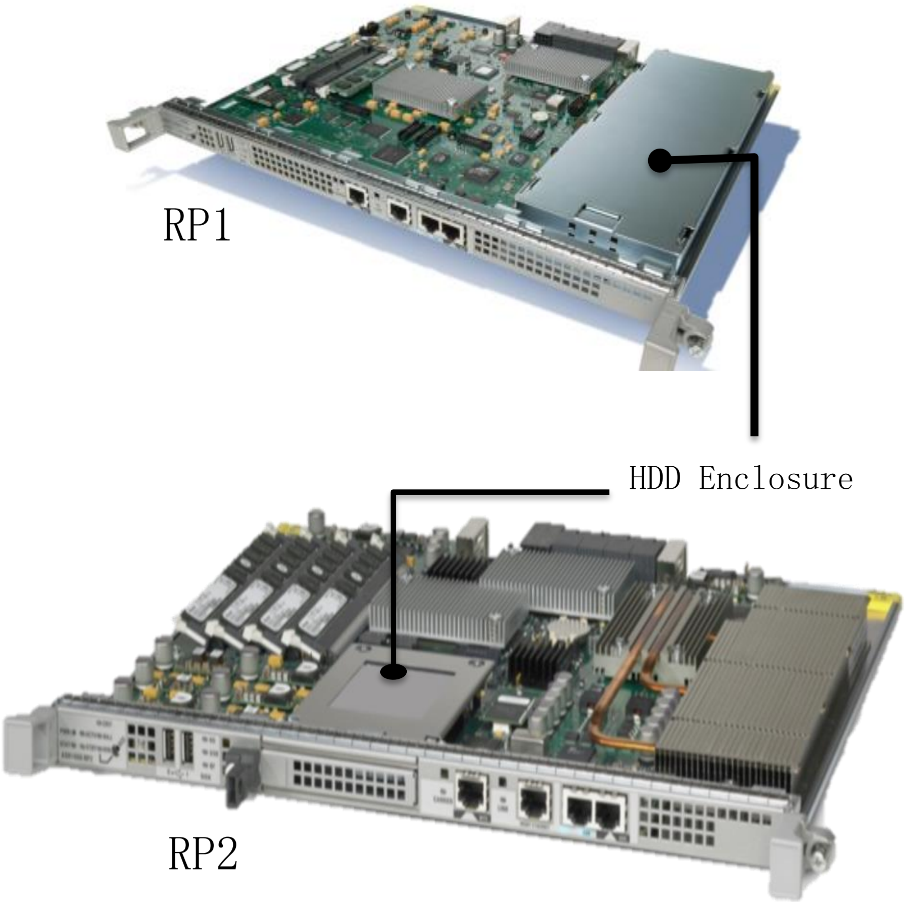
ASR1013

3x multispeed fan per PEM

4 PEMs total

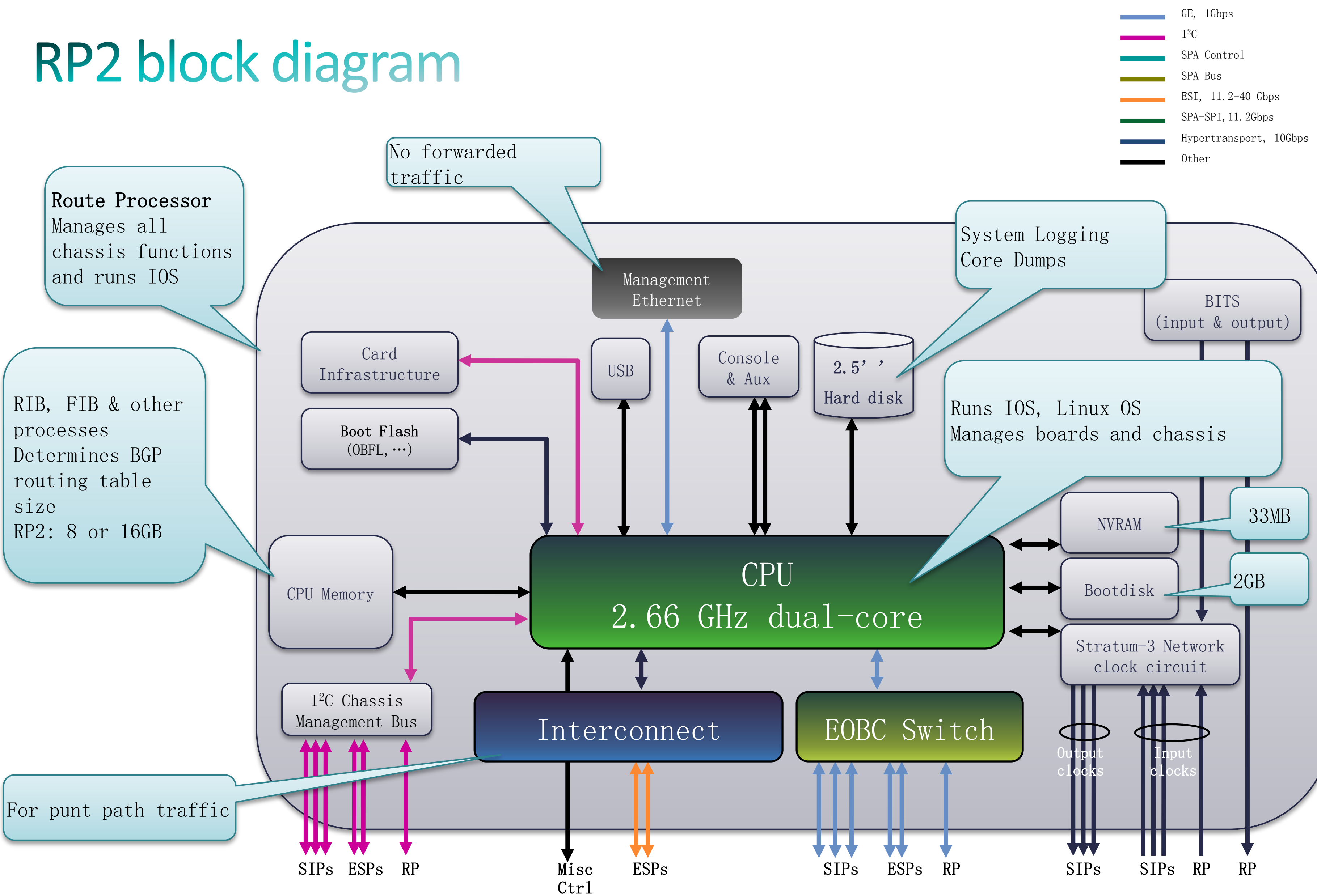
# Modular Route Processors: RP1 and RP2

- RP1
  - 1.5GHz PowerPC architecture
  - Up to 4GB IOS Memory
  - 1GB Bootflash
  - 33MB NVRAM
  - Fixed 40GB Hard Drive
- RP2
  - 2.66Ghz Intel dual-core architecture
  - 64-bit IOS XE
  - Up to 16GB IOS Memory
  - 2GB Bootflash (eUSB)
  - 33MB NVRAM
  - Hot swappable 80GB Hard Drive



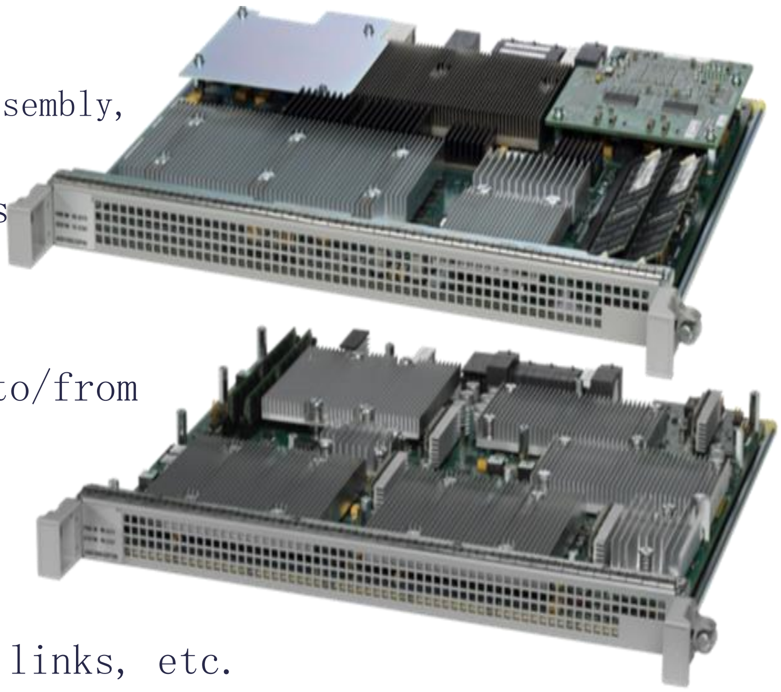
# RP2 block diagram



# ASR1000 Embedded Services Processor (ESP)

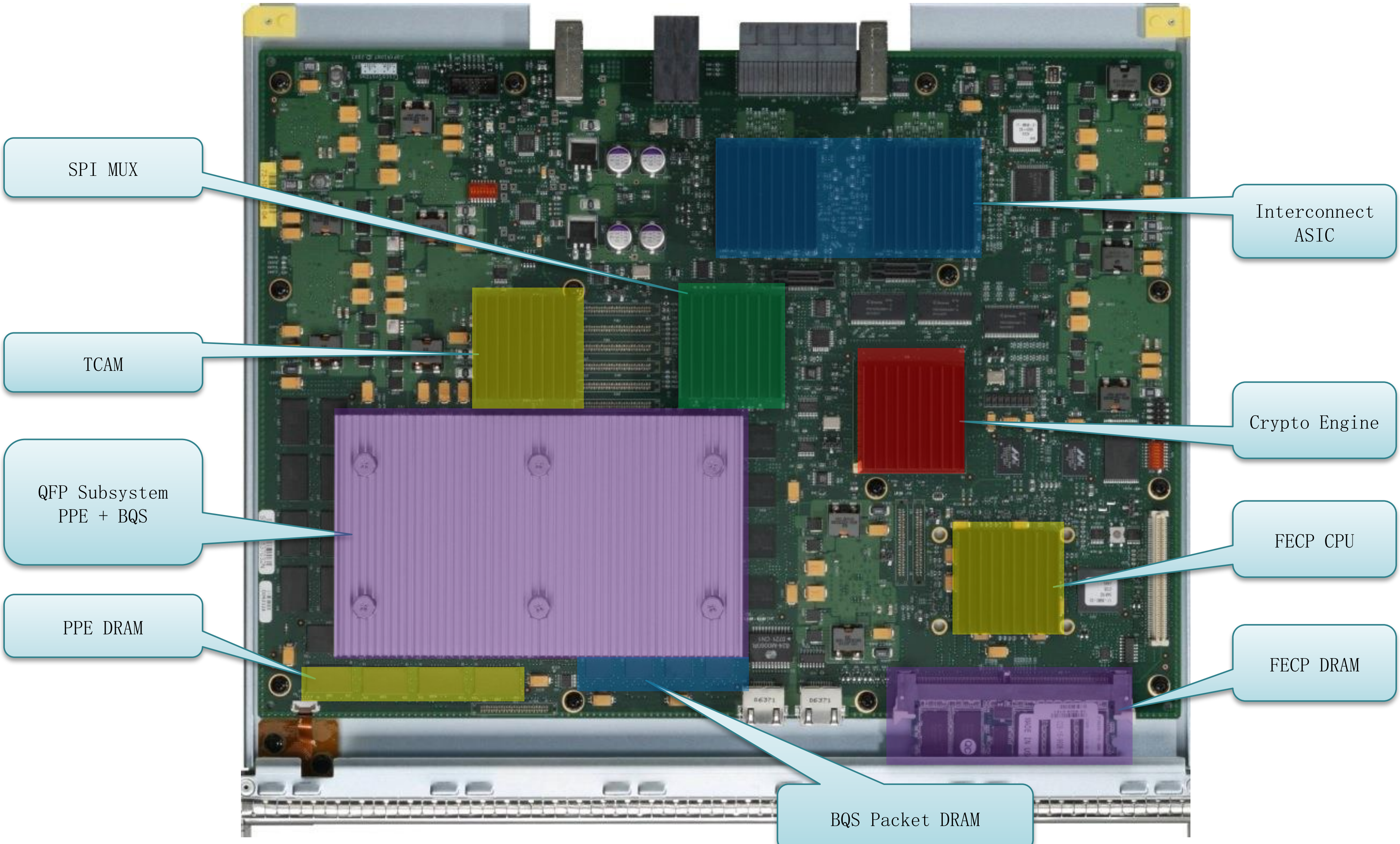
- Centralized, programmable forwarding engine providing full-packet processing
  - Packet Buffering and Queuing/Scheduling (BQS)
  - For output traffic to carrier cards/SPAs
  - For special features such as traffic shaping, reassembly, replication, punt to RP, crytptography, etc.
- 5 levels of HQoS scheduling, up to 464K Queues, Priority Propagation
- Dedicated crypto co-processor
- Interconnect providing data path links (ESI) to/from other cards over midplane
  - Transports traffic into and out of the Cisco Quantum Flow Processor (QFP)
  - Input scheduler for allocating QFP BW among ESIs
- FECP CPU manages QFP, crypto device, midplane links, etc.

ESP40

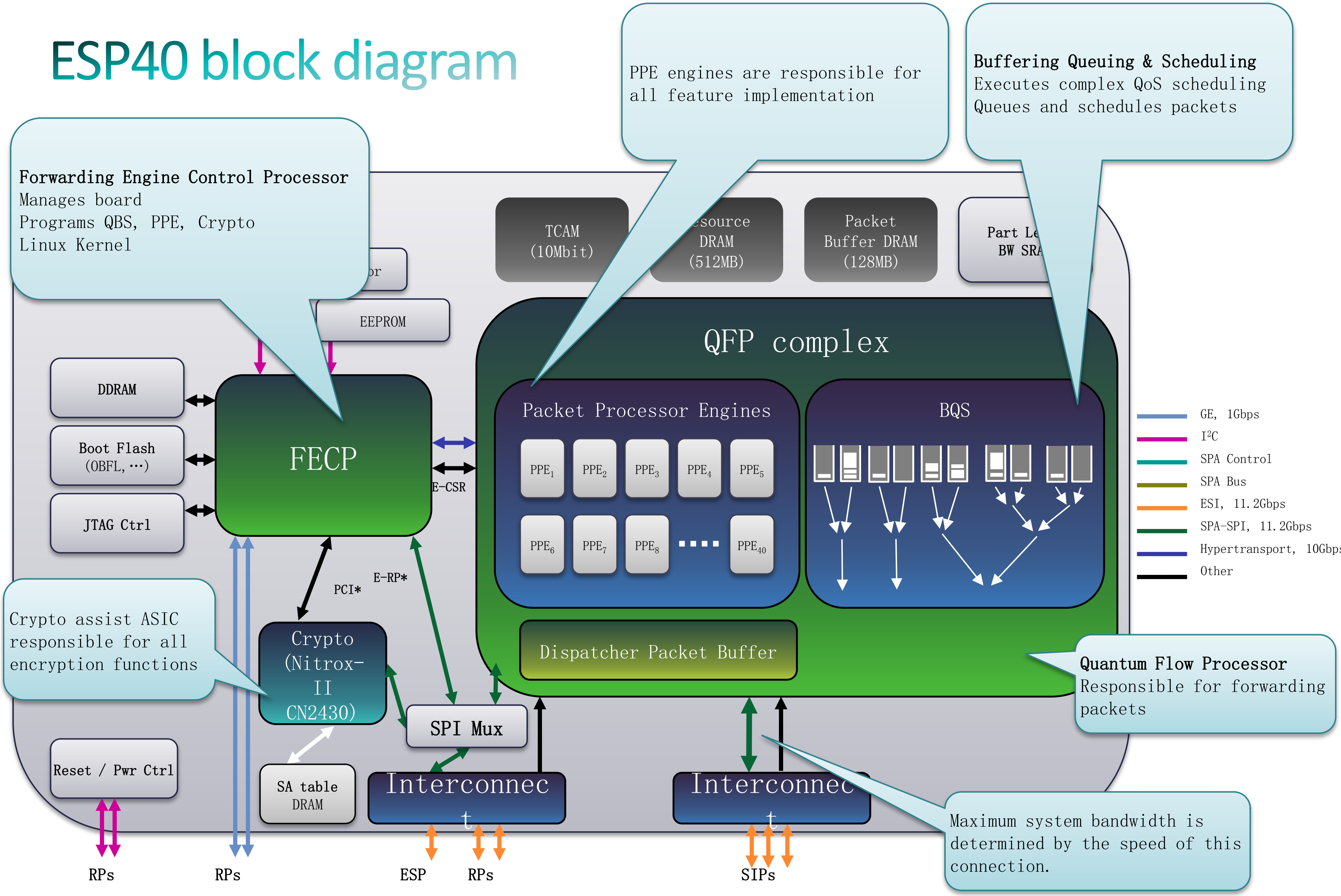


ESP100

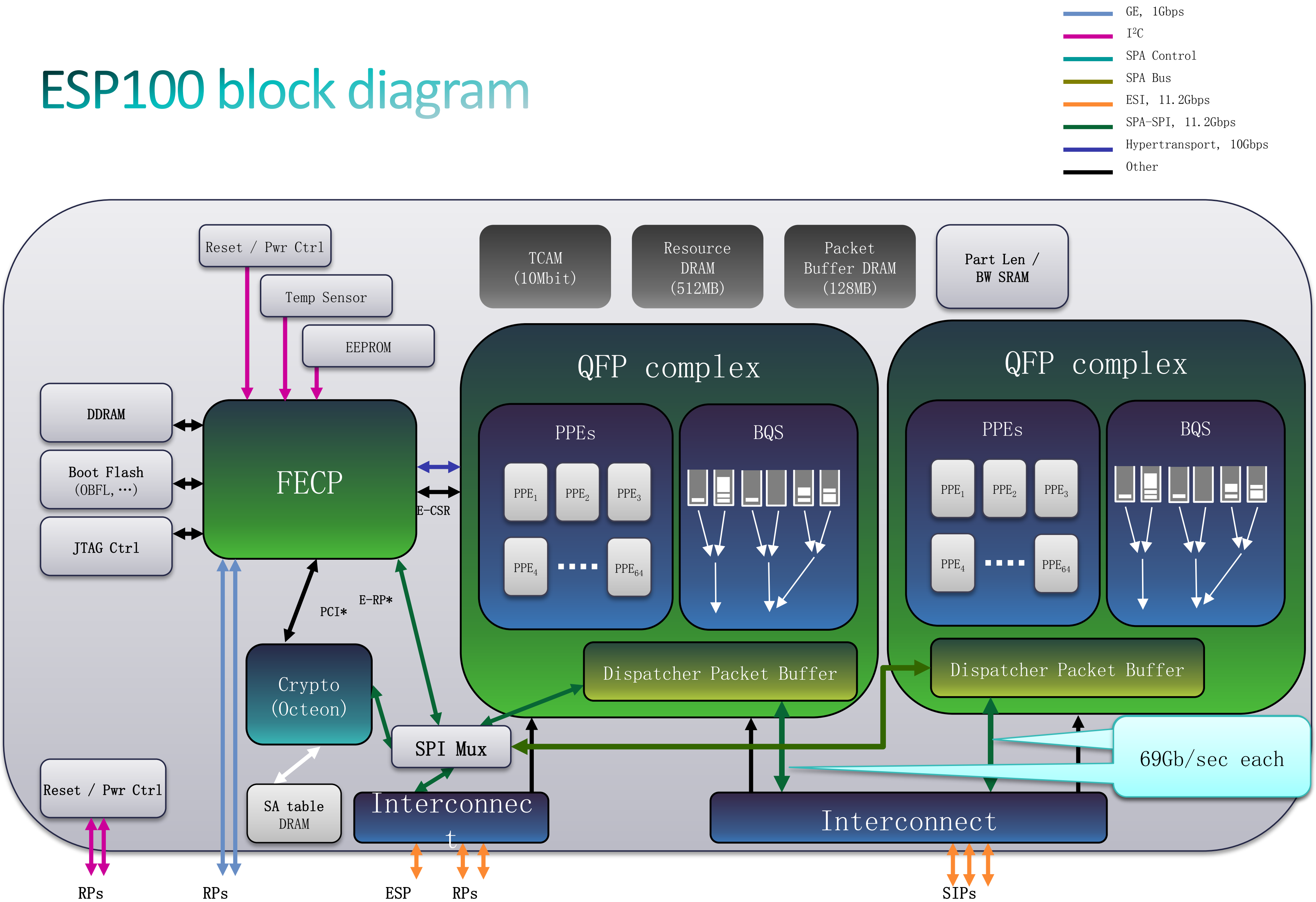
# ASR1000 Embedded Services Processor (ESP)



# ESP40 block diagram



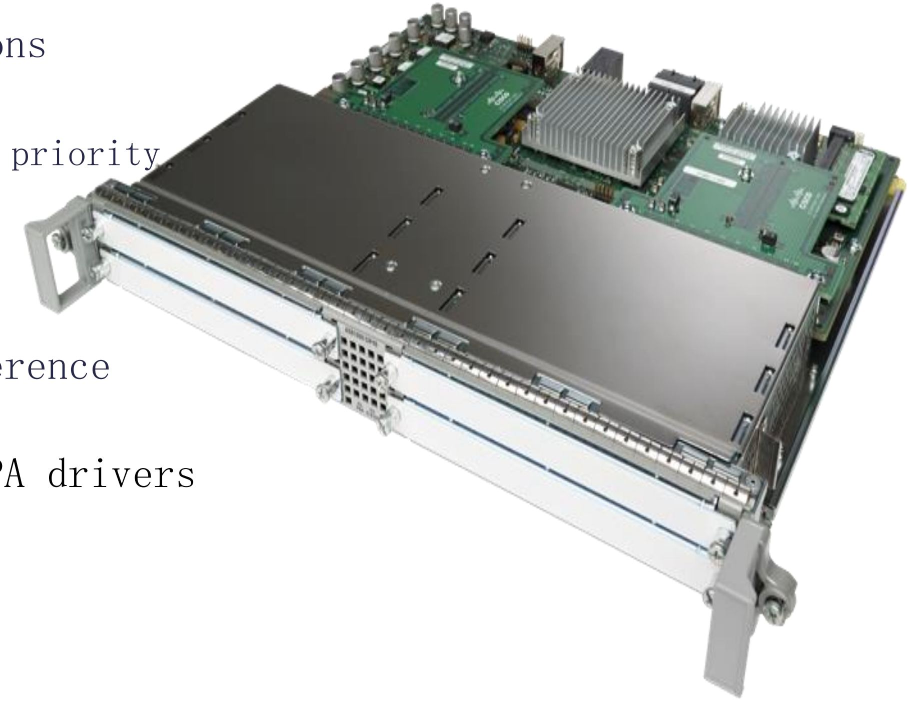
# ESP100 block diagram



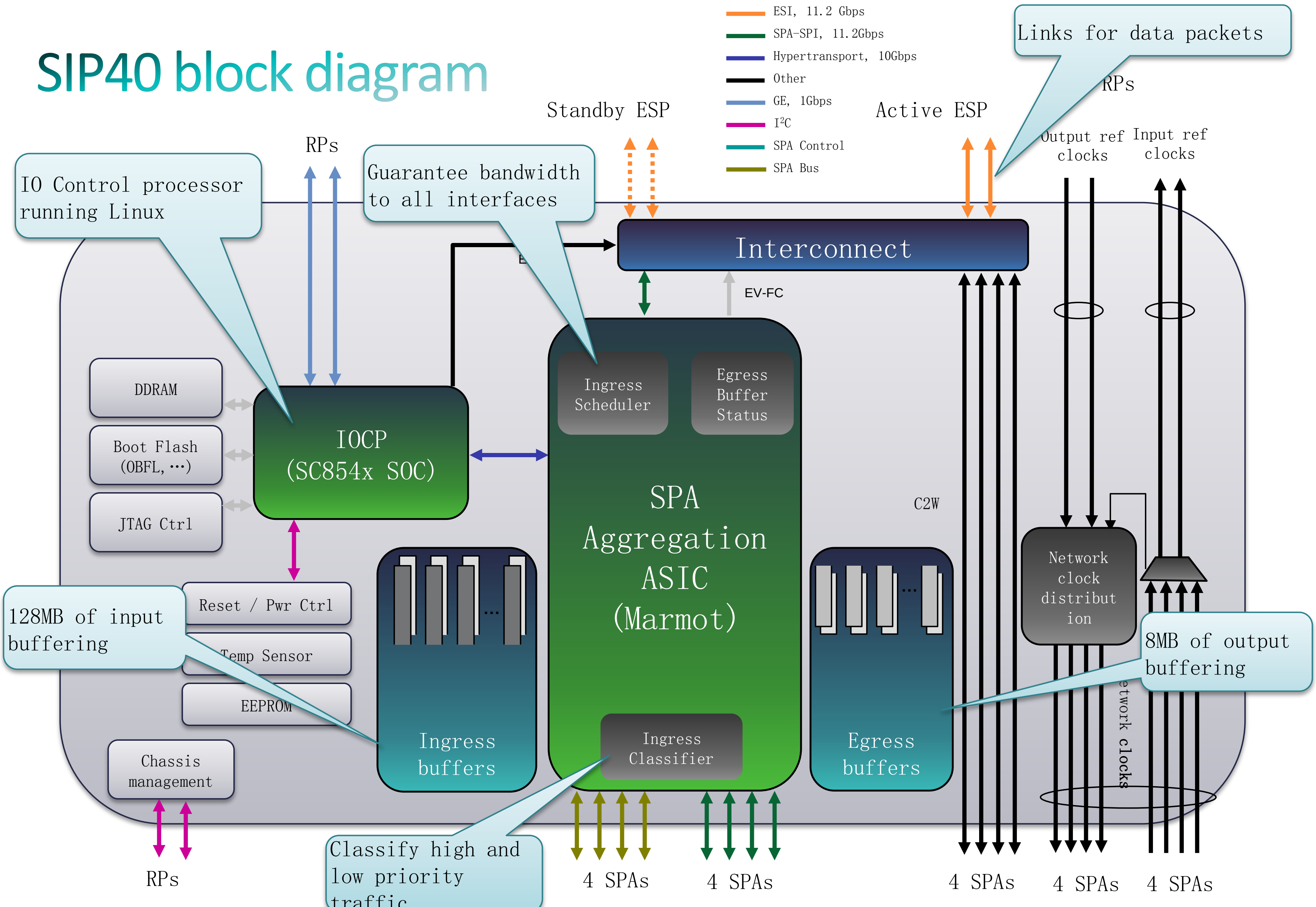

 

# ASR1000 SPA interface processor (SIP)

- SIP10 and SIP40 models
  - 10 and 40 Gbit/sec throughput
- Supports up to 4 SPAs
  - 4 HH, 2 FH, 2 HH+1 FH
  - full OIR support
- Do not participate in forwarding decisions
- Preliminary QoS
  - Ingress packet classification – high & low priority
  - Ingress over-subscription buffering
  - 128MB of ingress oversubscription buffering
- Capture stats on dropped packets
- Network clock distribution to SPAs, reference selection from SPAs
- IOCP manages midplane links, SPA OIR, SPA drivers

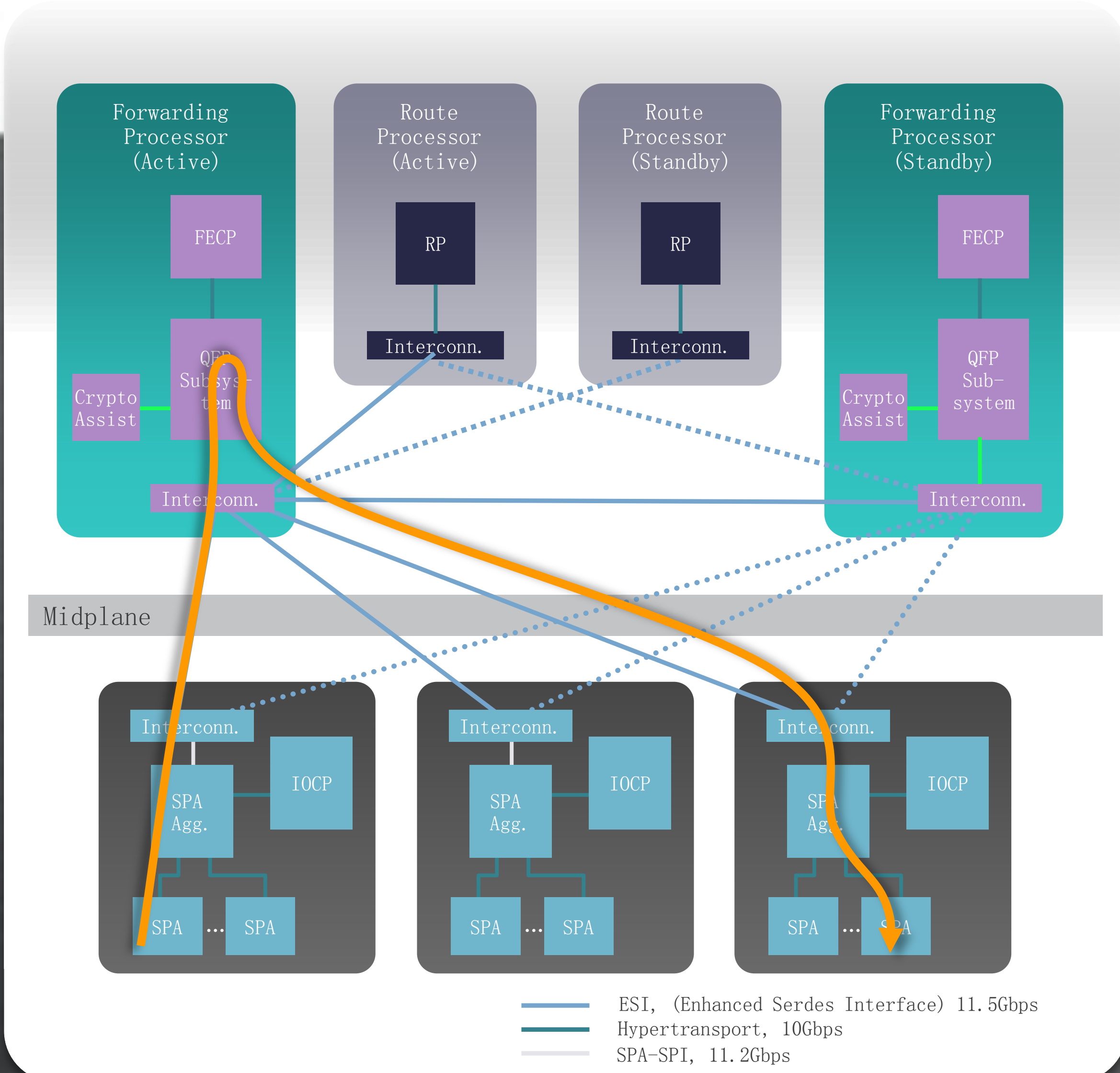


# SIP40 block diagram



# ASR1000 System Architecture

# ASR 1000 Building Blocks




RP (Route Processor)

- Handles control plane traffic
- Manages system

ESP (Embedded Services Processor)

- Handles forwarding plane traffic

SPA Interface Processor

- Shared Port Adapters provide interface connectivity

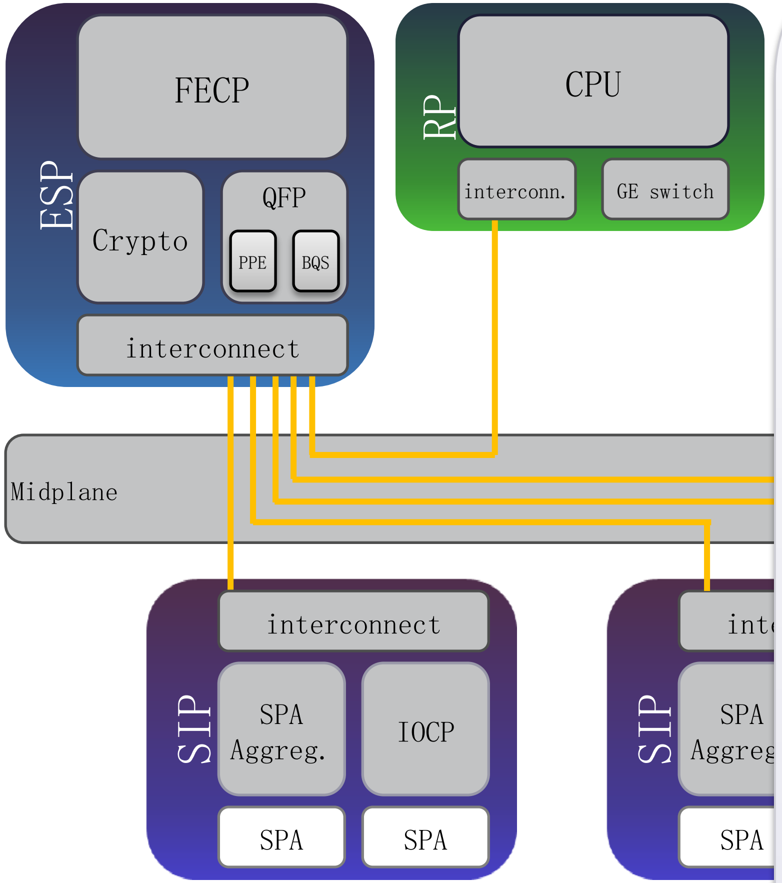
Centralized Forwarding Architecture

- All traffic flows through the active ESP, standby is synchronized with all flow state with a dedicated 10Gbps link

Distributed Control Architecture

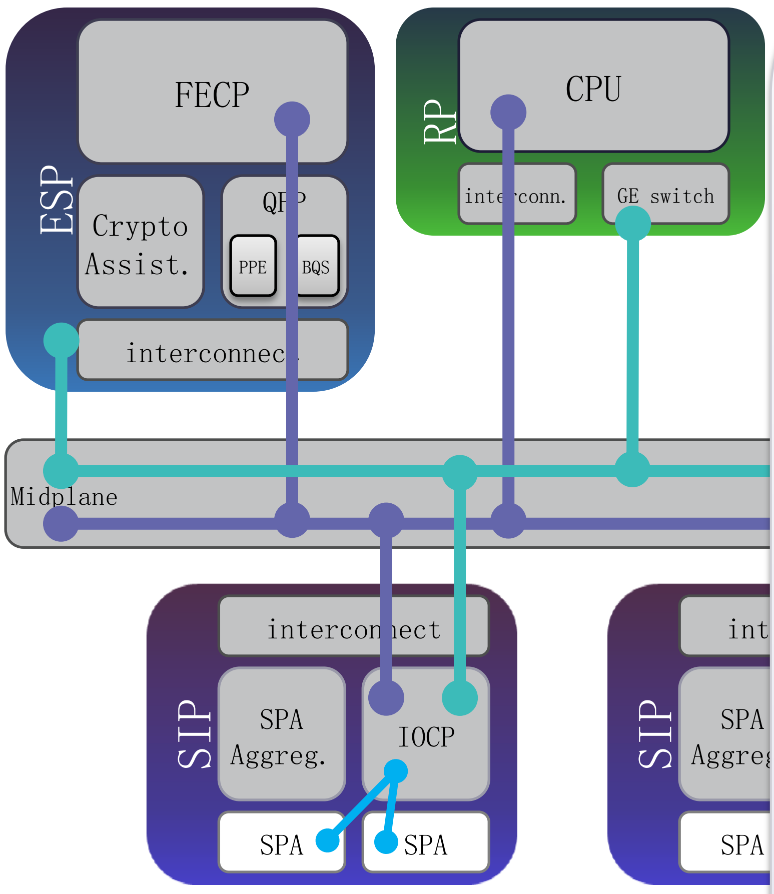
- All major system components have a powerful control processor dedicated for control and management planes

# ASR1000 data plane architecture




- Enhanced SerDes Interconnect (ESI)
  - serial communication via midplane
  - can run at 11.5Gbps or 23Gbps
- Provides data packet communication
  - data packets between ESPs and other linecards punt/inject traffic to/from RP
  - state synchronization between ESPs
  - two ESI links between each ESP and all linecards
  - Additional full set of ESI links to standby ESP CRC protection of packet contents

# ASR1000 control plane architecture




## Ethernet out-of-band channel (EOBC)

- indication if cards are installed and ready
- loading images, stats collection
- messages to program QFP

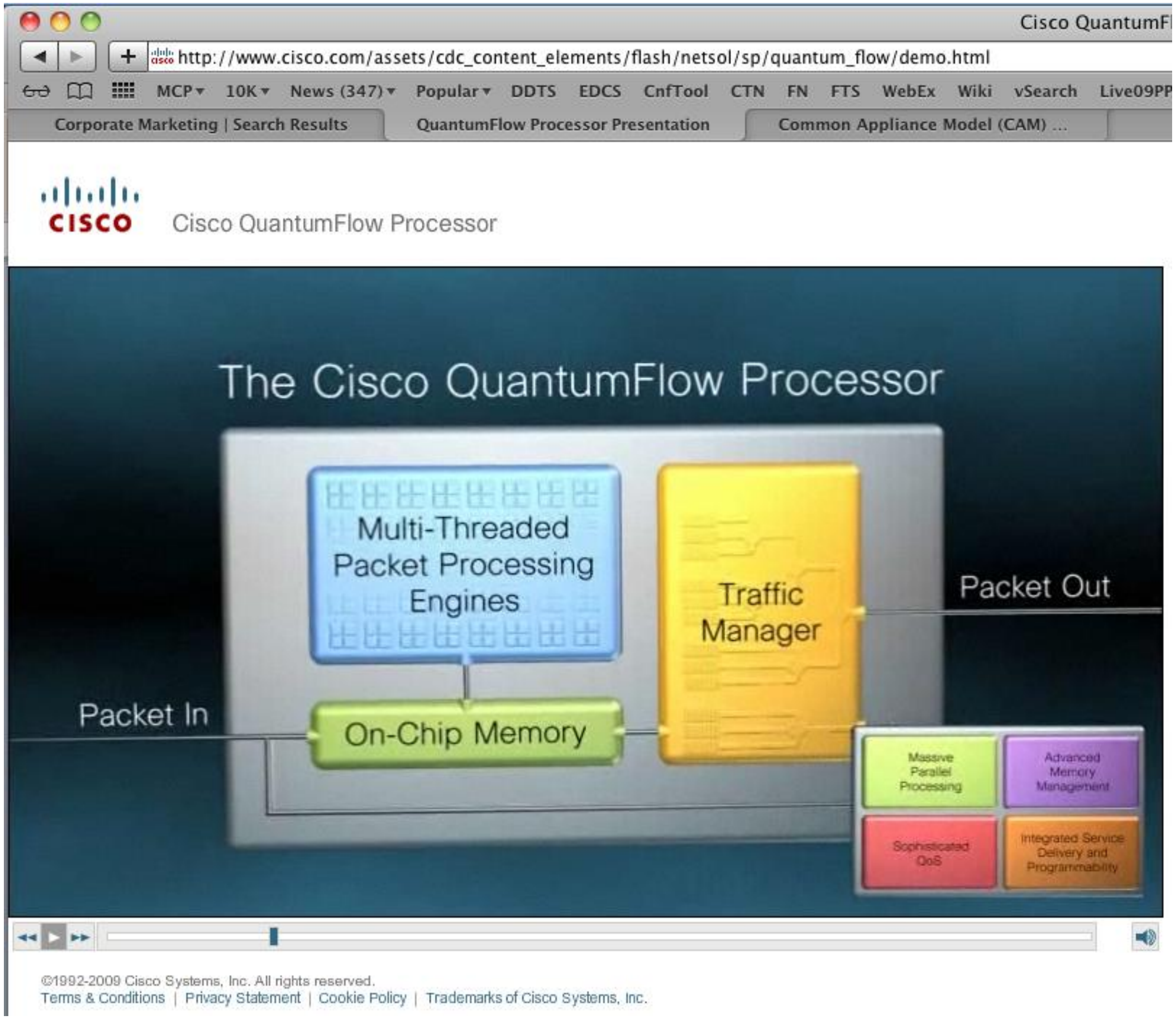
## Inter-Integrated Circuit ($I^2C$)

- monitor health of hardware components
- control resets
- communicate active/standby
- real time presence and ready indicators
- control the other RP (reset, power-down, etc.)
- report power-supply status
- signal ESP active/standby
- EEPROM access

## SPA control links

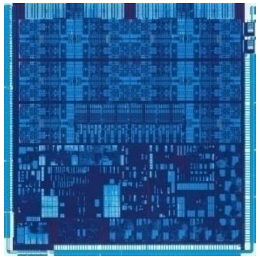
- detect SPA OIR
- reset SPAs (via $I^2C$)
- power-control SPAs (via $I^2C$)
- read EEPROMs

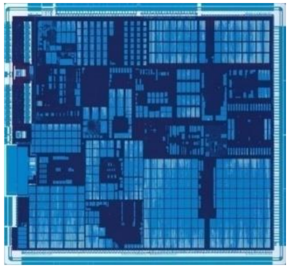
# QFP Flash




http://www.cisco.com/cdc_content_elements/flash/netsol/sp/quantum_flow/demo.html

# QFP Background





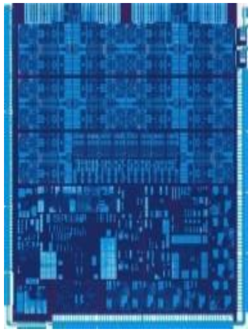
- COT design chosen
  - Existing CPUs did not offer forwarding power required
  - Memory architecture of general purpose CPUs relies on large caches (64B/128B) -> Inefficient mapping for network features
- QFP uses small memory access sizes (16B)
  - minimizes wasted memory reads and increases memory access
  - for the same raw memory BW, a 16B read allows 4-8 times the number of memory accesses/sec as a CPU using 64/128B accesses
- Preserves C-language programming support
  - Including stacking for nested procedures
  - Differentiator as compared to NPUs
  - Key to feature velocity
  - Support for portable, large-scale development
- Addition of hardware assists to further boost performance
  - TCAM, PLU, HMR…
  - Trade-off power requirement vs. board space

| | Cisco QFP | Sun Ultrasparc T2 | Intel Core 2 Mobile U7600 |
|---|---|---|---|
| Total number processes (cores x threads) | 160 | 64 | 2 |
| Power per process | 0.51W | 1.01W | 5W |
| Scalable traffic management | 128k queues | None | None |

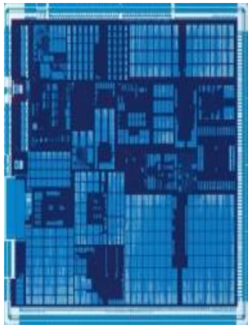
# Cisco Quantum Flow Processor
# – ASR1000 series innovation

1st generation
QFP Chip Set

- Five year design and continued evolution – now on 2nd generation
- Massively parallel: 64 cores, 4 threads per core for 256 packets in
- QFP Architecture designed to scale to beyond 100Gbit/sec
- High-priority traffic path throughout forwarding processing
- Packet replication capabilities for Lawful Intercept
- Full visibility of entire L2 frame
- Latency: tens of microseconds with features enabled
- Interfaces on-chip for external cryptographic engine
- 2nd generation QFP is capable of 70Gbit/sec, 32Mpps processing
- Can cascade 1, 2 or 4 chips to build higher capacity ESPs

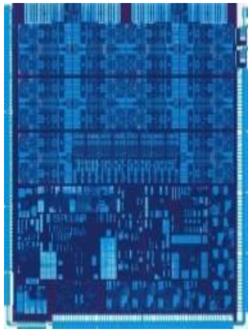
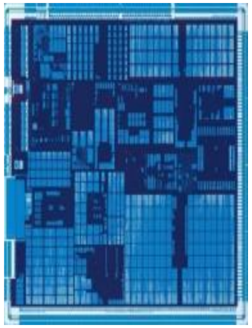


Cisco QFP
Packet Processor



Cisco QFP Traffic Manager
(Buffering, Queueing, Scheduling)

# Cisco Quantum Flow Processor – 2nd generation details

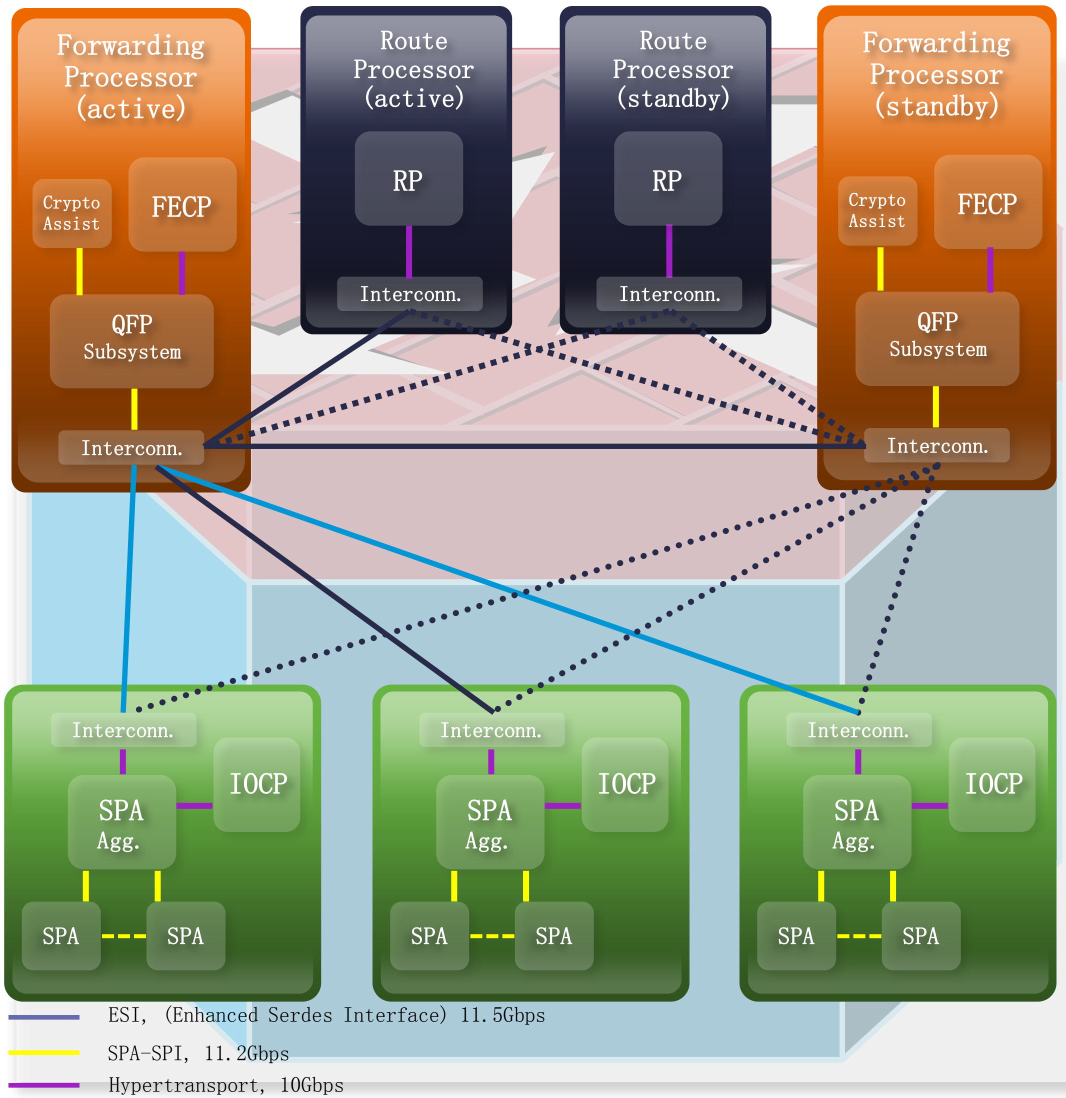



- Used on ASR1002-X, ESP100 & ESP200
- 2$^{nd}$ gen QFP integrates both the PPE engine and the Traffic manager into a single ASIC
  - 64 PPEs
  - 116K queues per 2$^{nd}$ gen QFP
    128K queues for 1$^{st}$ gen QFP
  - 3$^{rd}$ gen QFP can be in a matrix
    ESP100 has 232K queues
    ESP200 has 464K queues
- 1$^{st}$ and 2$^{nd}$ gen QFPs run the same code
  - Maintains identical feature behavior between QFP hardware releases
- Full configuration consistency
- Identical feature behavior (NAT, FW, etc)
- In-service hardware upgrade from ESP40 to ESP100 supported
- Differences
  - Minor behavioral show-command differences
  - Deployment differences in deployments with large number of BQS schedules

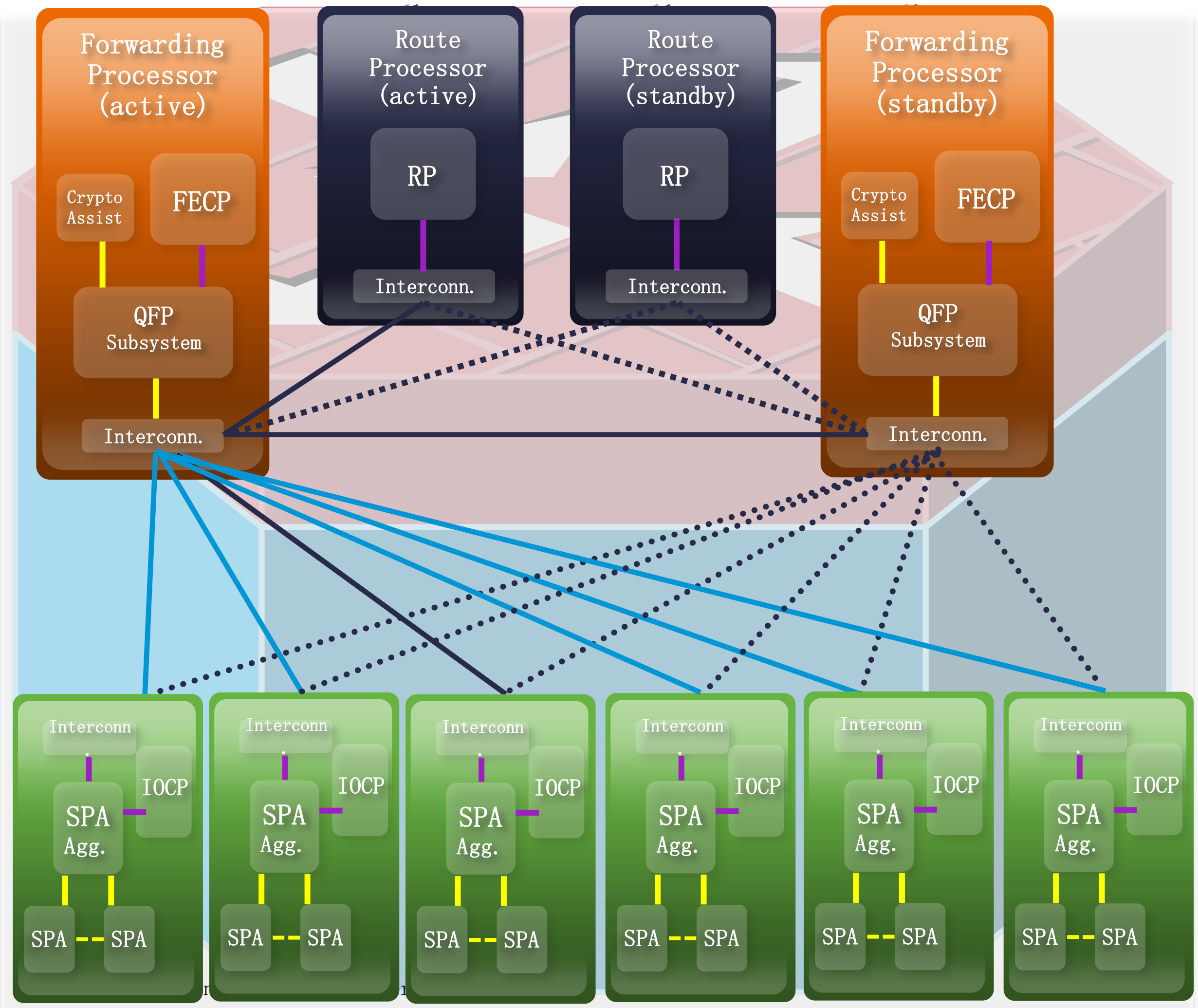
# ASR1000 – designed to gracefully handle oversubscription

- Total bandwidth of the system is determined by the following factors
  - Type of forwarding engine: eg. ESP10, ESP20, ESP40, ESP100 or ESP200
  - Type of SIP: SIP10 or SIP40
- Step 1: SPA-to-SIP Oversubscription
  - Up to 4 x 10Gbps SPAs per SIP 10 = 4:1 Oversubscription Max
  - No over subscription for SIP40 = 1:1
  - Calculate your configured SPA BW to SIP oversubscription ratio
- Step 2: SIP-to-ESP Oversubscription
  - All SIPs in a chassis share the ESP bandwidth
  - Calculate configured SIP BW to ESP capacity ratio
- Total Oversubscription = Step1 x Step2

# ASR 1006 System Architecture



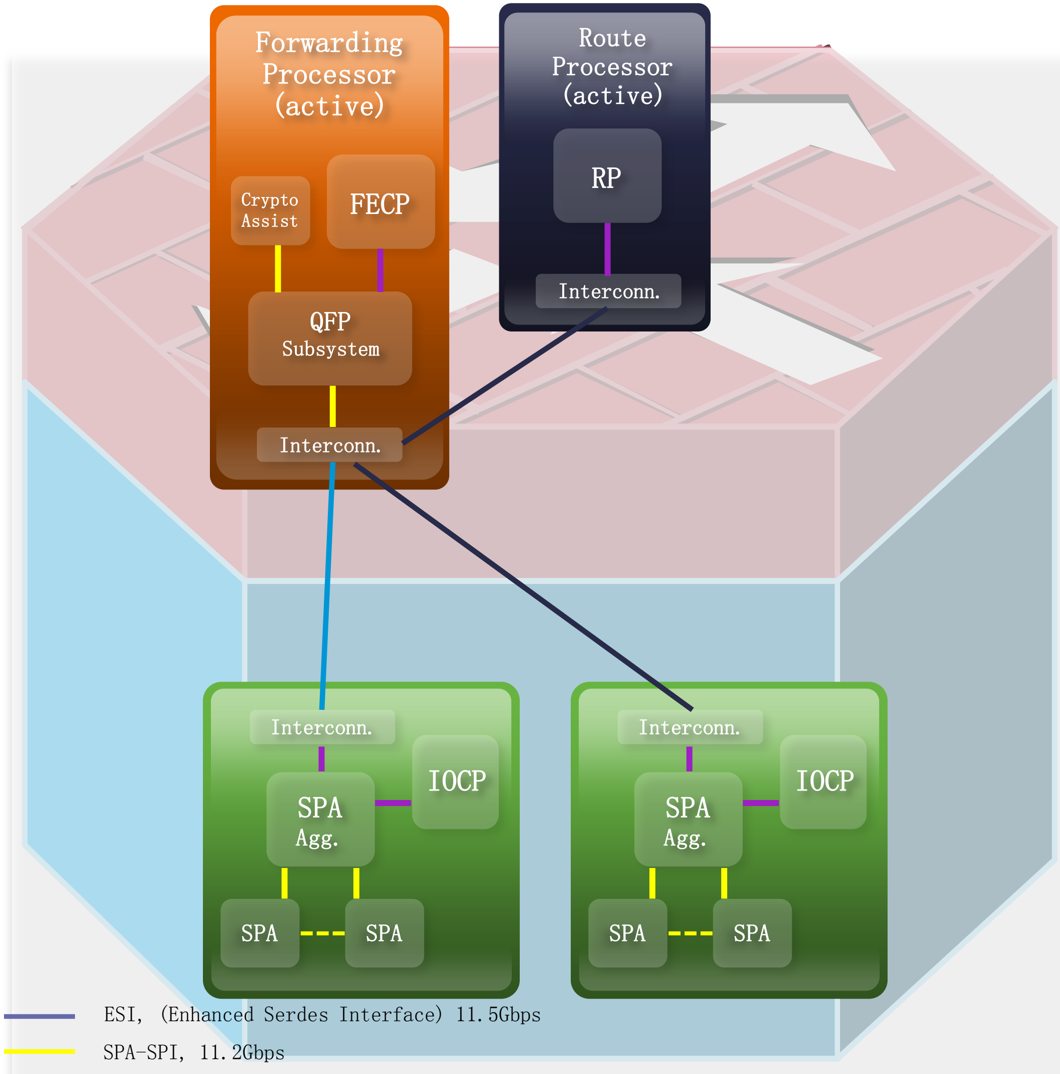
# ASR 1013 System Architecture



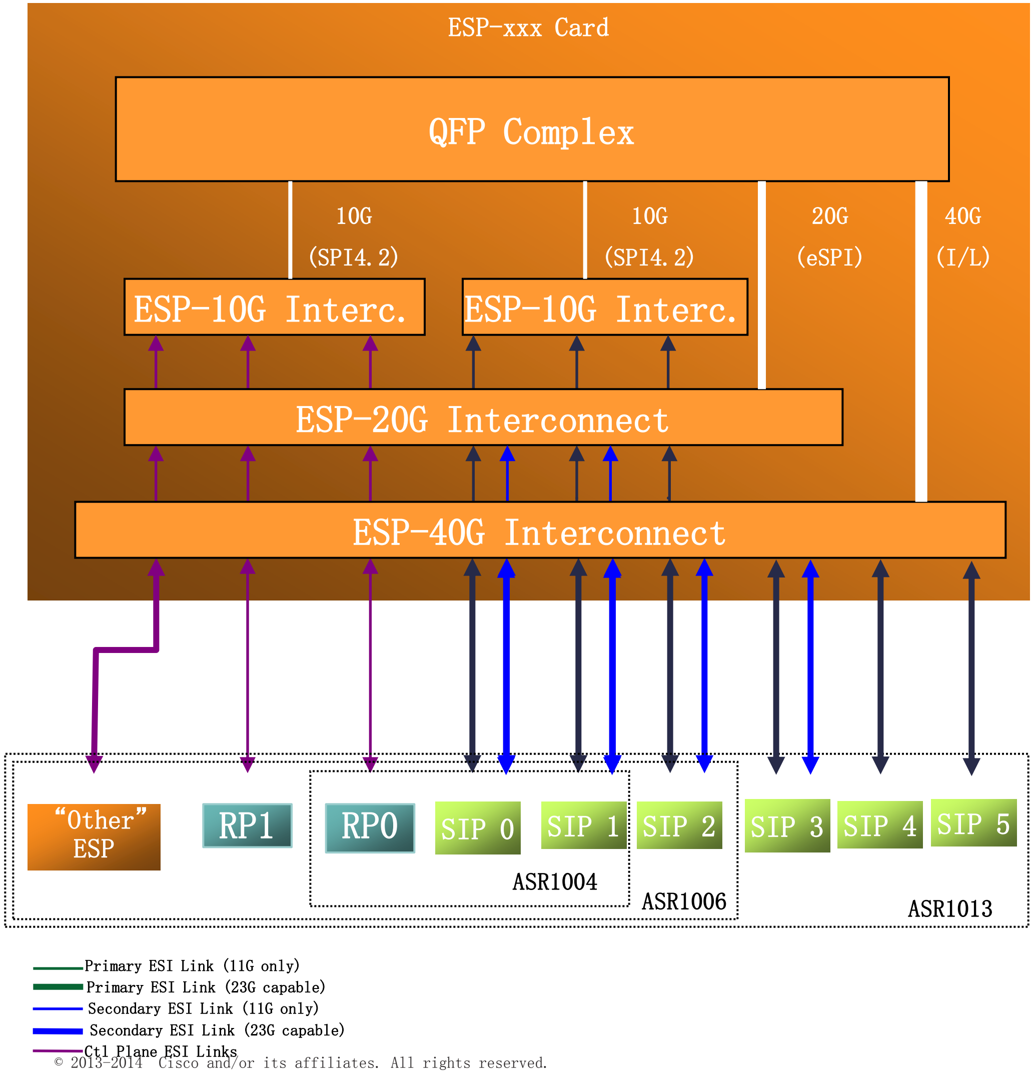

SPA-SPI, 11.2Gbps

Hypertransport, 10Gbps

# ASR 1004 System Architecture

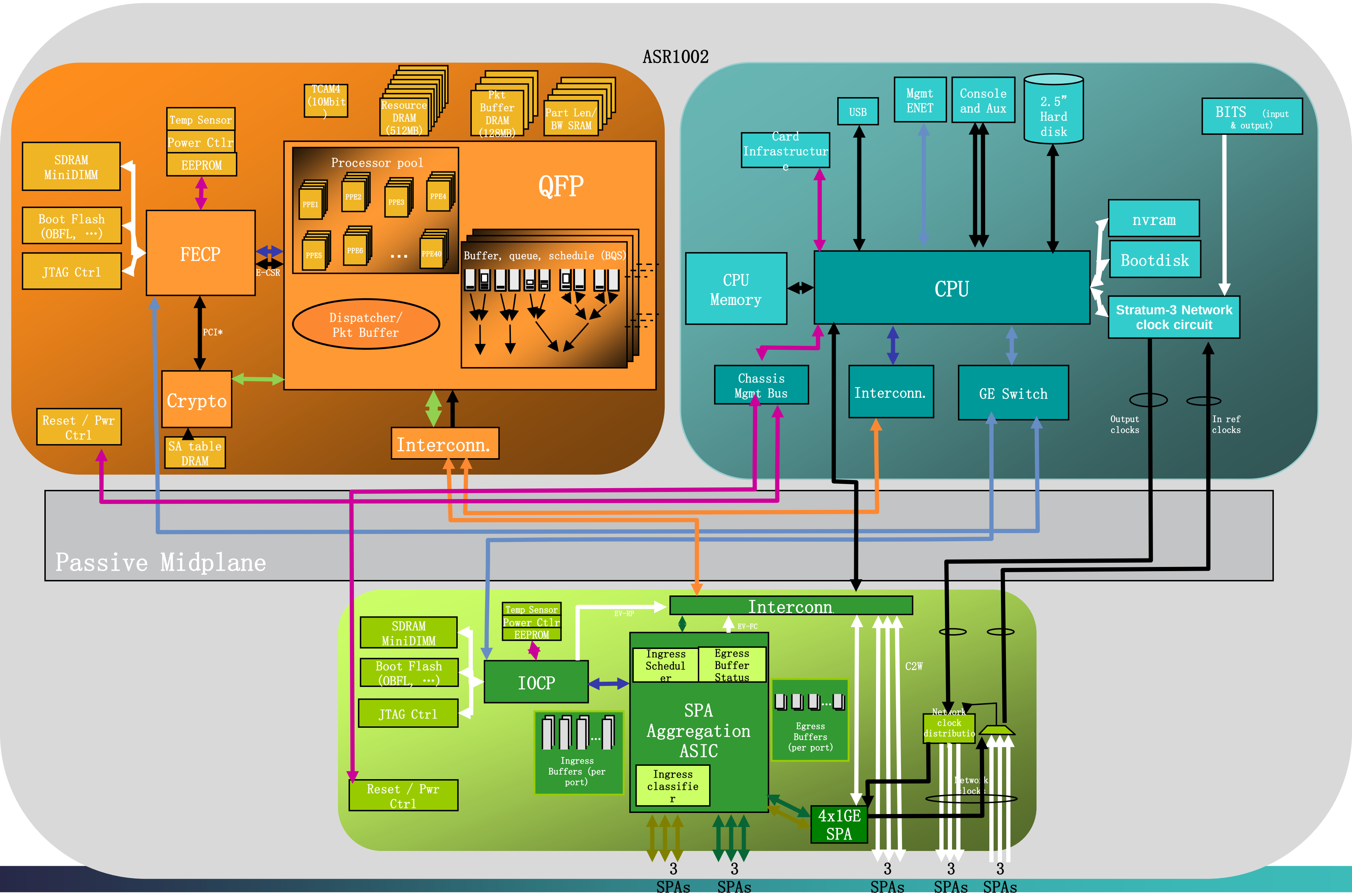


# ESI Capacity by ESP-xxx and SIP-xxx

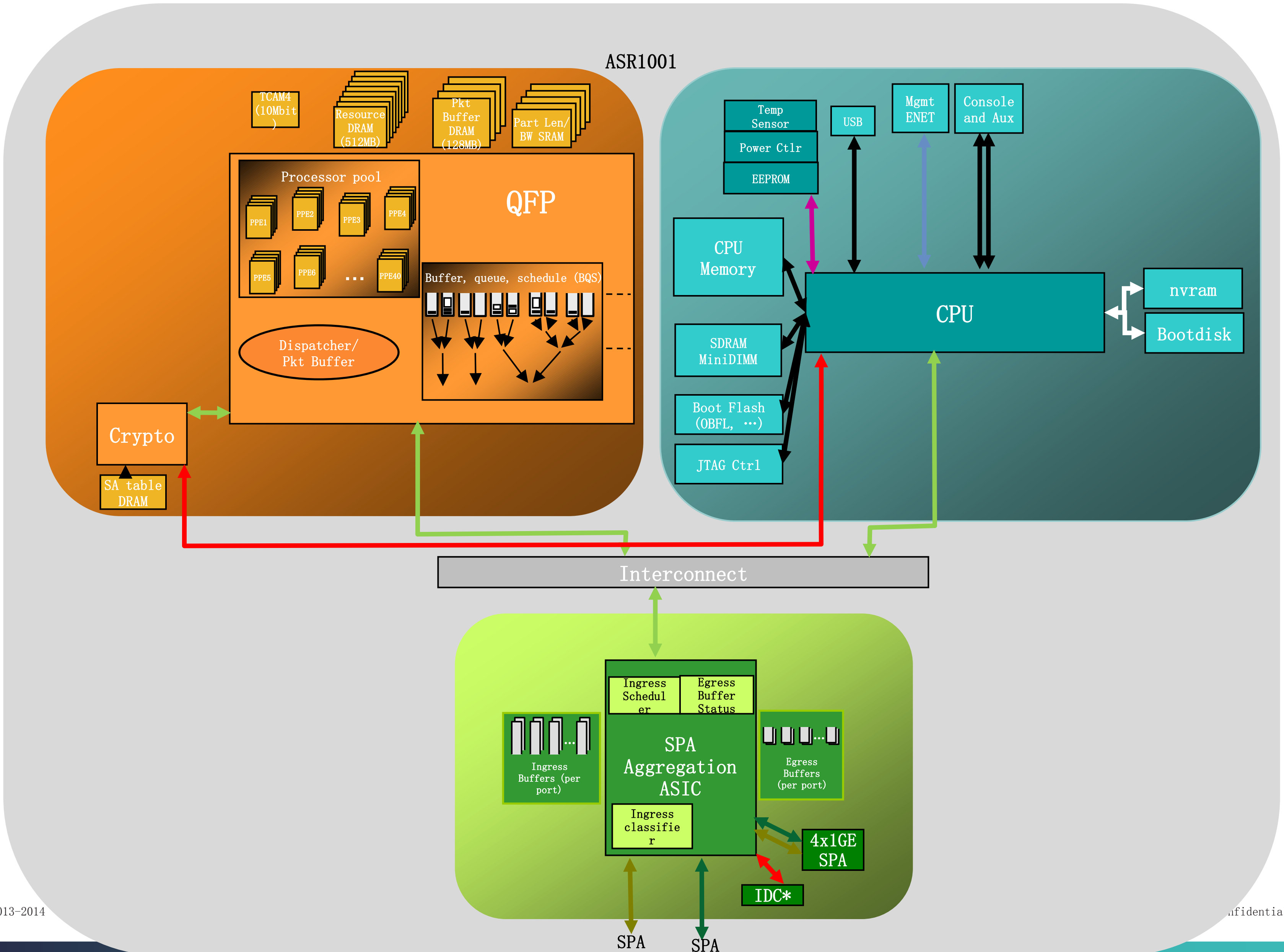



- Enhanced SerDes Interconnect (ESI) links over midplane carry
  - Packets between ESP and the other cards (SIPs, RP and other ESP)
  - Network traffic to/from SPA SIPs
  - Punt/inject traffic to/from RP (e.g. network control pkts)
  - State synchronization to/from standby ESP
- Additional full set of ESI links to/from standby ESP (not shown)
- CRC protection of packet contents
- ESP-10G: 1 x 11.5G ESI to each SIP slot
- ESP-20G: 2 x 11.5G ESI to two SIP slots; 1 x 11.5G to third SIP slot
- ESP-40G
  - 2 x 23G ESI* to all three SIP slots
  - Also 23G between two ESP-40G's
- SIP-10G: supports 1 x 11.5G mode only
- SIP-40G: supports 1 x 11.5G & 2 x 23G

# ASR 1002 Block Diagram



# ASR 1001 Block Diagram

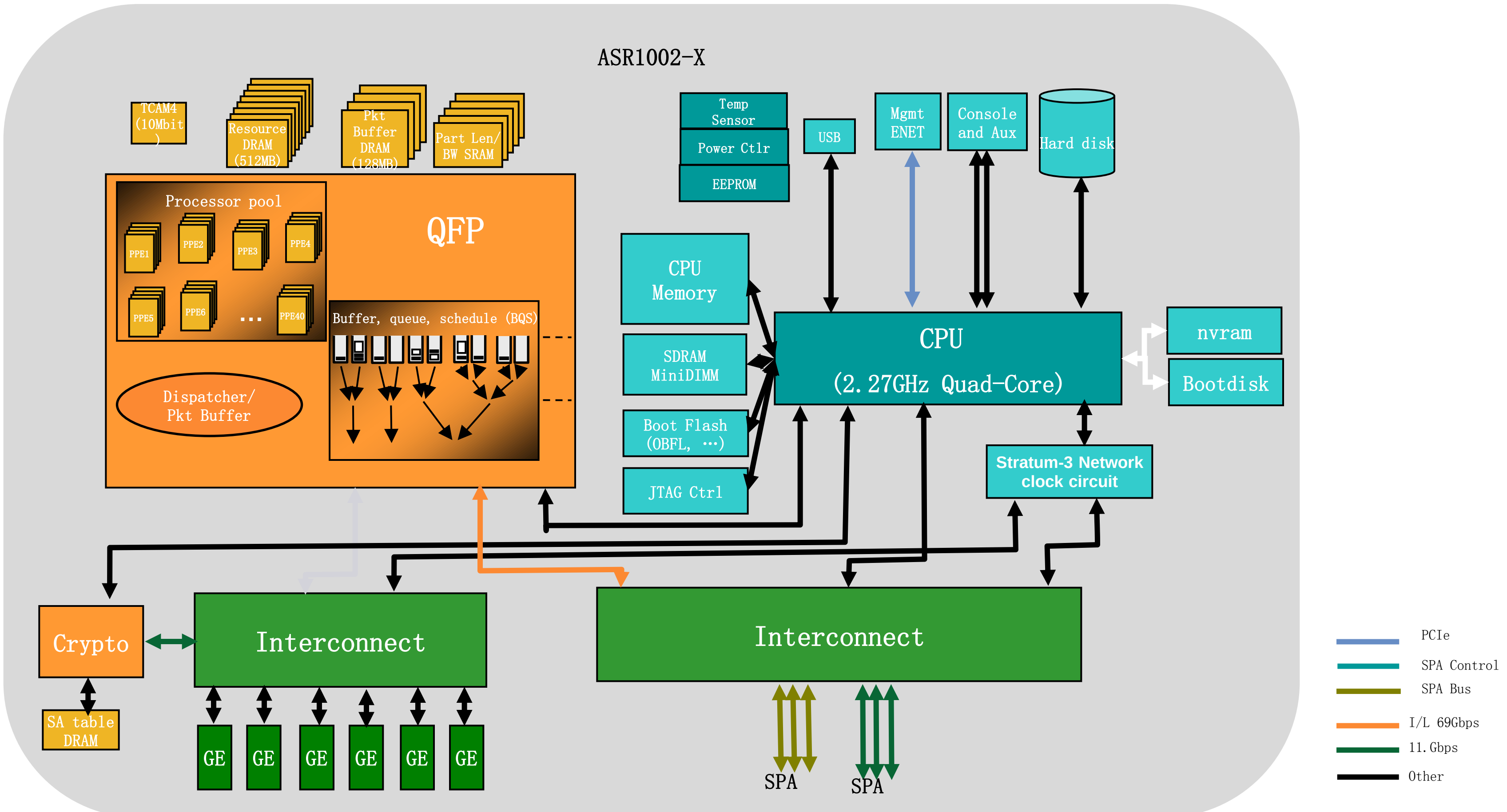






* In case of HDD, SATA connected

# ASR 1002-X Block Diagram




- Increased throughput and Crypto Bandwidth
- ESP / SIP-40 integrated into chassis
- SW licensing to address performance and scaling range

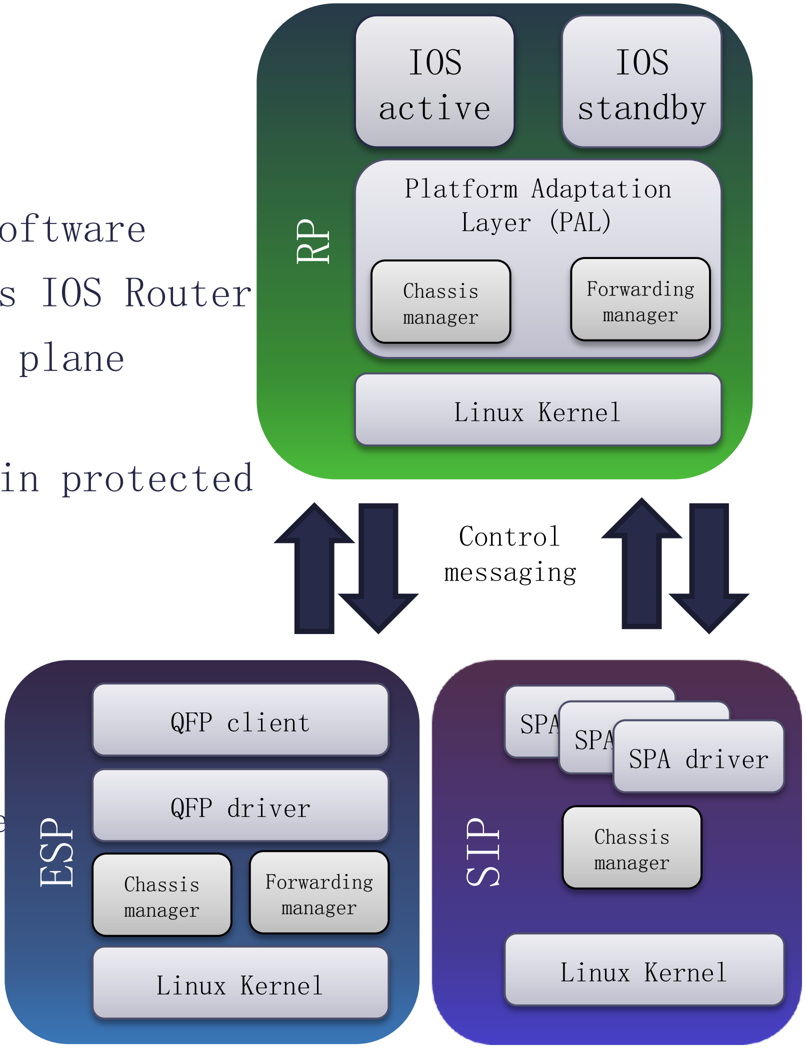



* In case of HDD, SATA connected

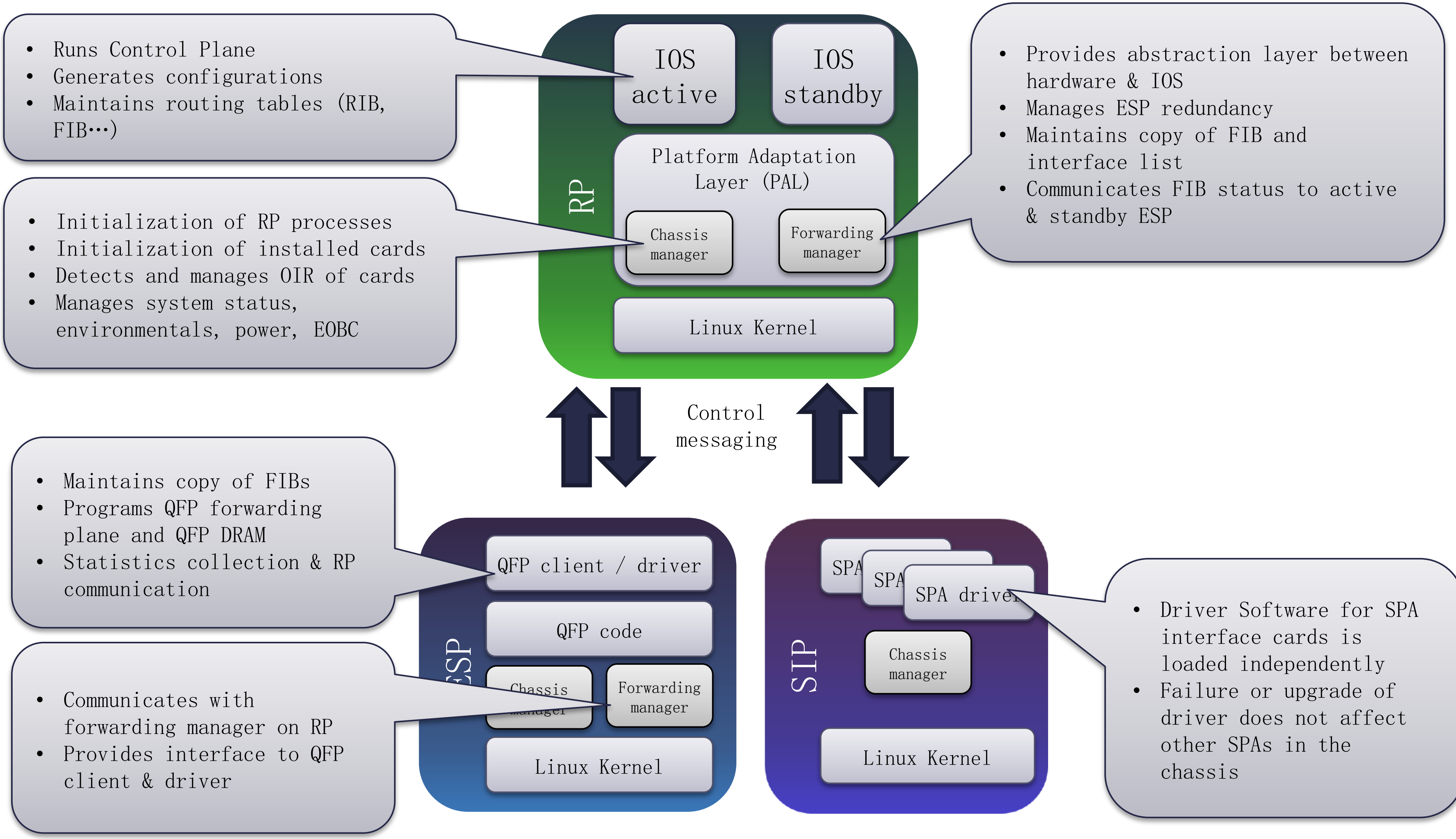
# ASR1000 Software Architecture

# IOS XE Software architecture

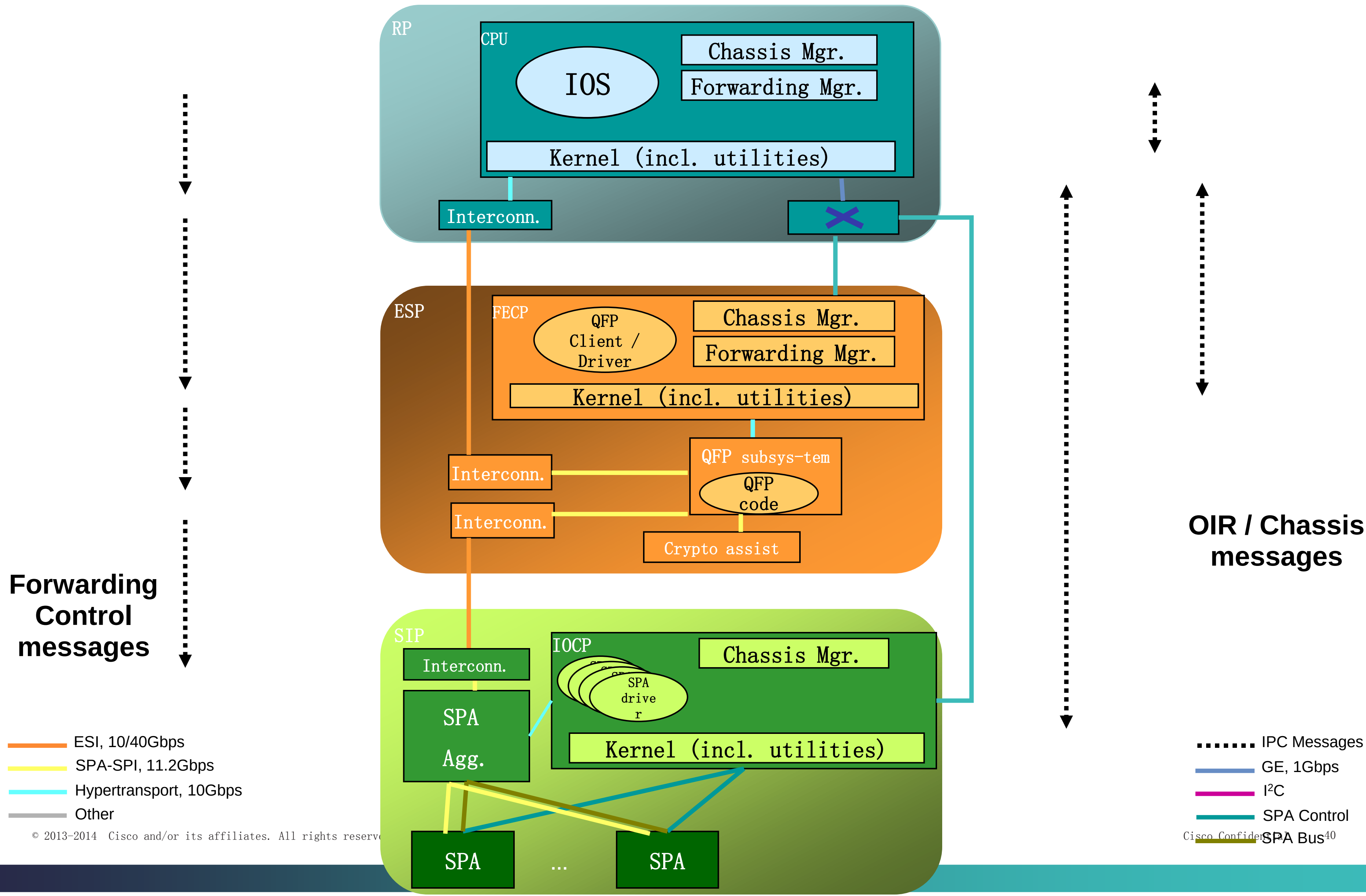
- IOS XE = IOS + IOS XE Middleware + Platform Software
- Operational Consistency—same look and feel as IOS Router
- IOS runs as its own Linux process for control plane
  64 bit capable
- Linux kernel with multiple processes running in protected memory
  Fault containment
  Re-startability
  ISSU of individual SW packages
- ASR 1000 HA Innovations
  Zero packet loss with RP Failover, <50ms ESP Failover
  Software redundancy



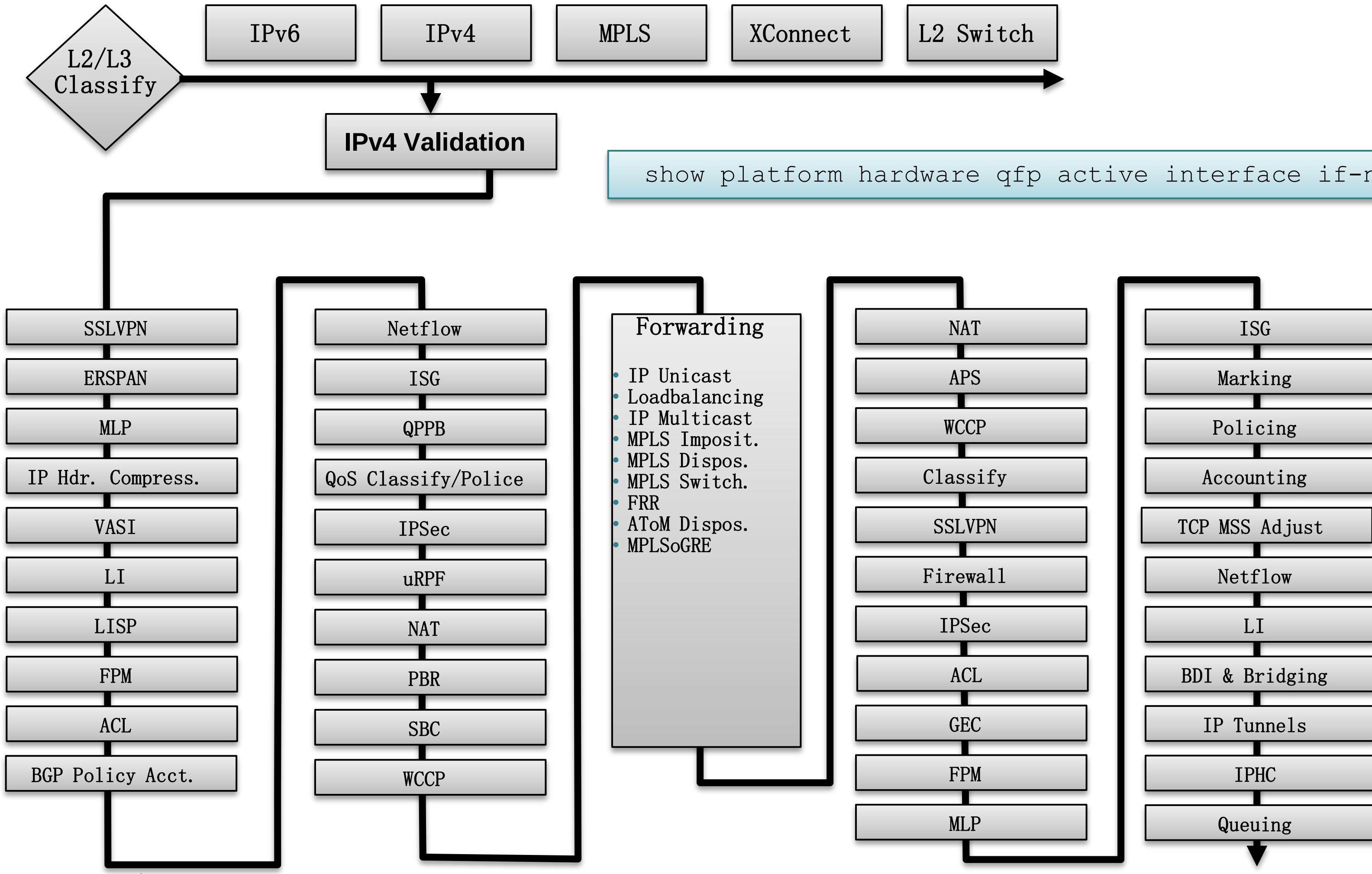
# IOS XE Software architecture



# Control Plane Process Communication



# Feature Invocation Array (FIA) for efficient packet forwarding




```
show platform hardware qfp active interface if-name <name>
```

# ASR 1000 Initialization Sequence

**SIP**

**POST**
**HW Initialization**
**Initialize EOBC**
**Wait for RP Master**

**Detect $RP_{act}$ via ROMMON**
**Upload inventory via CPLD**
**ROMMON download software package**
**Boot Kernel**
**$CM_{SIP}$ registers with $CM_{RP}$**
**$CM_{SIP}$ starts IOS-XE for SPAs**

**$CM_{SIP}$ sends ESI link status**

**RP**

**POST**
**HW Initialization**
**Initialize EOBC**
**Boot Linux Kernel and Middleware**
**Start IOS**
**$CM_{RP}$ detects cards via CPLD**
**$CM_{RP}$ determines Master RP and ESP**
**$CM_{RP}$ informs SIPs & ESP about Master via $I^2C$**
**$CM_{RP}$ downloads SIP & ESP software packages to SIP / ESP**

**$CM_{RP}$ sends ESI config to $CM_{SIP}$ and $CM_{ESP}$**

**ESP**

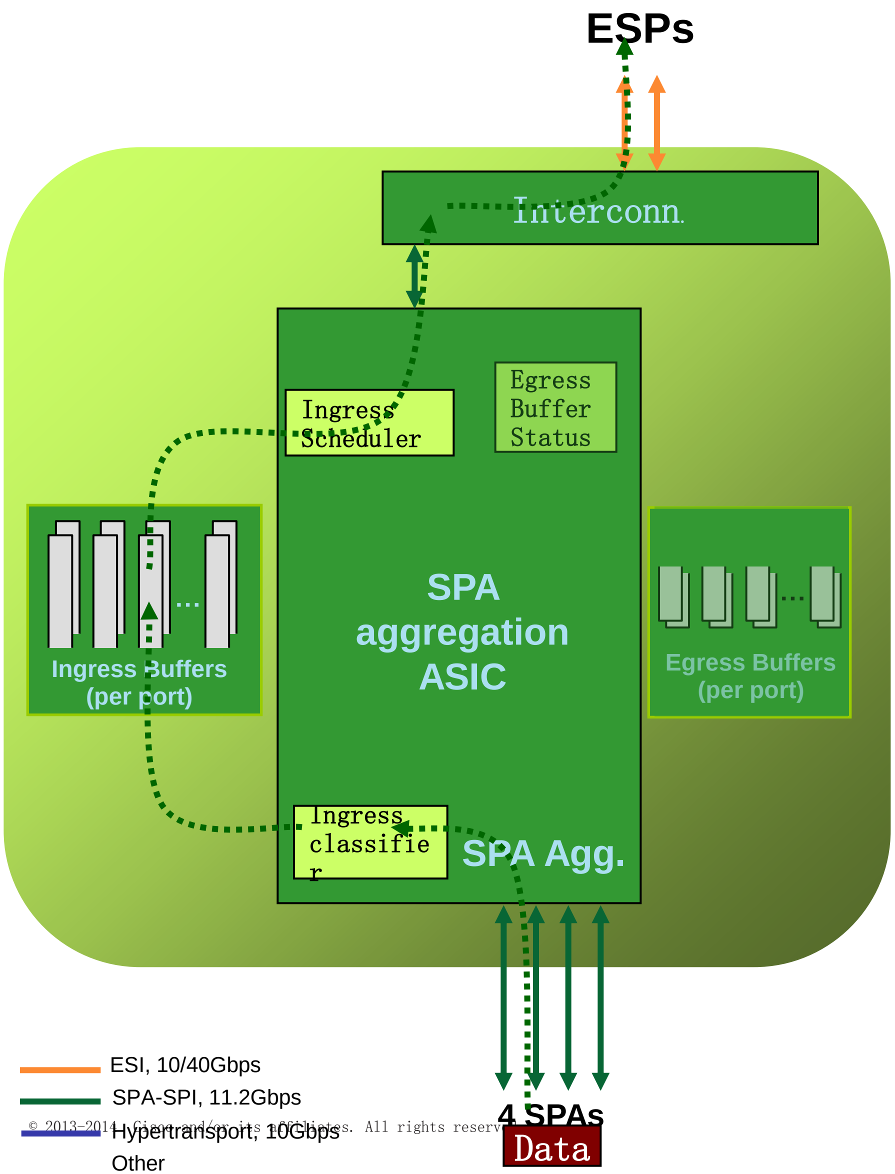
**POST**
**HW Initialization**
**Initialize EOBC**
**Wait for RP Master**

**Detect $RP_{act}$ via ROMMON**
**Upload inventory via CPLD**
**ROMMON download software package**
**Boot Kernel**
**$CM_{ESP}$ registers with $CM_{RP}$**
**$CM_{ESP}$ starts QFP**
**$CM_{ESP}$ signals ready to RP**
**$CM_{ESP}$ sends ESI link status**

- Master RP determines which RP becomes $RP_{act}$ (and which RP becomes $RP_{sby}$)
- Status of ASR 1000 hardware component is kept in the RPs chassis management process $CM_{RP}$
- Failure in the bring-up of any component will make it unavailable
  - Could result in single-RP chassis or single ESP chassis, for example

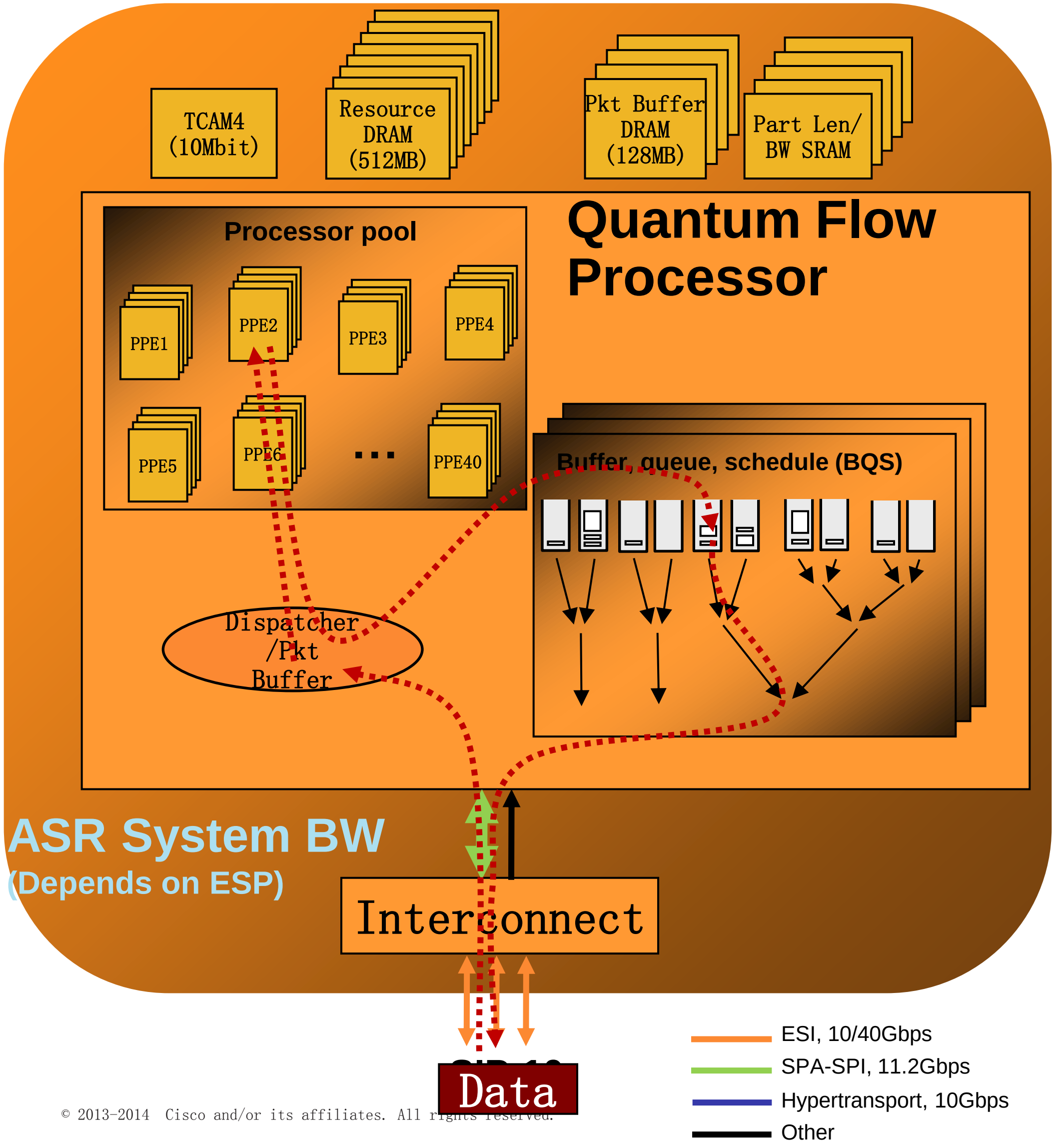
# Packet flows

# Data Packet Flow: From SPA Through SIP




1. SPA receives packet data from its network interfaces and transfers the packet to the SIP
2. SPA Aggregation ASIC classifies the packet into H/L priority
3. SIP writes packet data to external 128B memory (at 40Gbps from 4 full-rate SPAs)
4. Ingress buffer memory is carved into 64 queues. The queues are arranged by SPA-SPI channel and optionally H/L. Channels on "channelized" SPAs share the same queue.
5. SPA ASIC selects among ingress queues for next pkt to send to ESP over ESI. It prepares the packet for internal transmission
6. The interconnect transmits packet data of selected packet over ESI to active ESP at up to 40 Gbps
7. Active ESP can backpressure SIP via ESI ctl message to slow pkt transfer over ESI if overloaded (provides separate backpressure for Hi vs. Low priority pkt data)

# Data Packet Flow: Through ESP40




1. Packet arrives on QFP
2. Packet assigned to a PPE thread.
3. The PPE thread processes the packet in a feature chain similar to 12.2S IOS (very basic view of a v4 use case):

   Input Features applied

   NetFlow, MQC/NBAR Classify, FW, RPF, Mark/Police, NAT, WCCP etc.

   Forwarding Decision is made

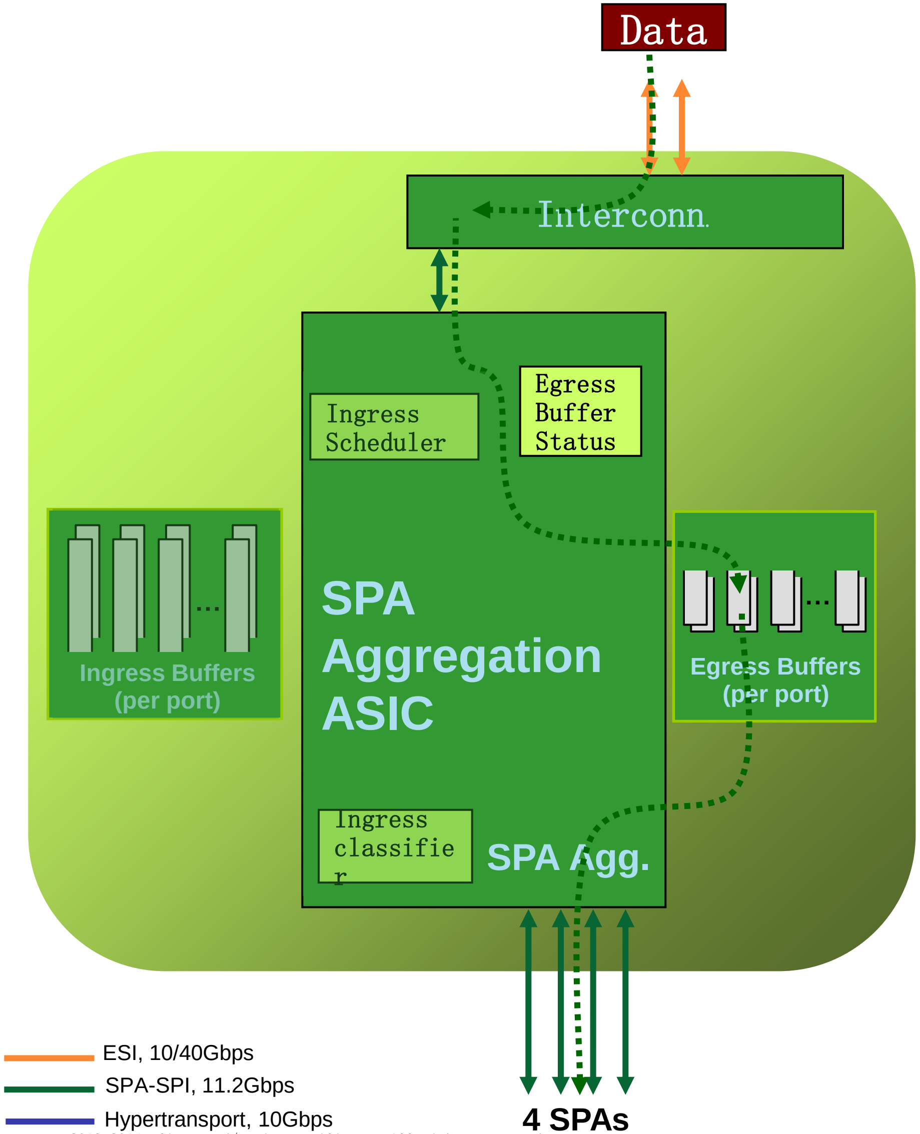
   Ipv4 FIB, Load Balance, MPLS, MPLSoGRE, Multicast etc.

   Output Features applied

   NetFlow, FW, NAT, Crypto, MQC/NBAR Classify, Police/Mark etc.

   Finished
4. Packet released from on-chip memory to Traffic Manager (Queued)
5. The Traffic Manager schedules which traffic to send to which SIP interface (or RP or Crypto Chip) based on priority and what is configured in MQC
6. SIP can independently backpressure ESP via ESI control message to pace the packet transfer if overloaded

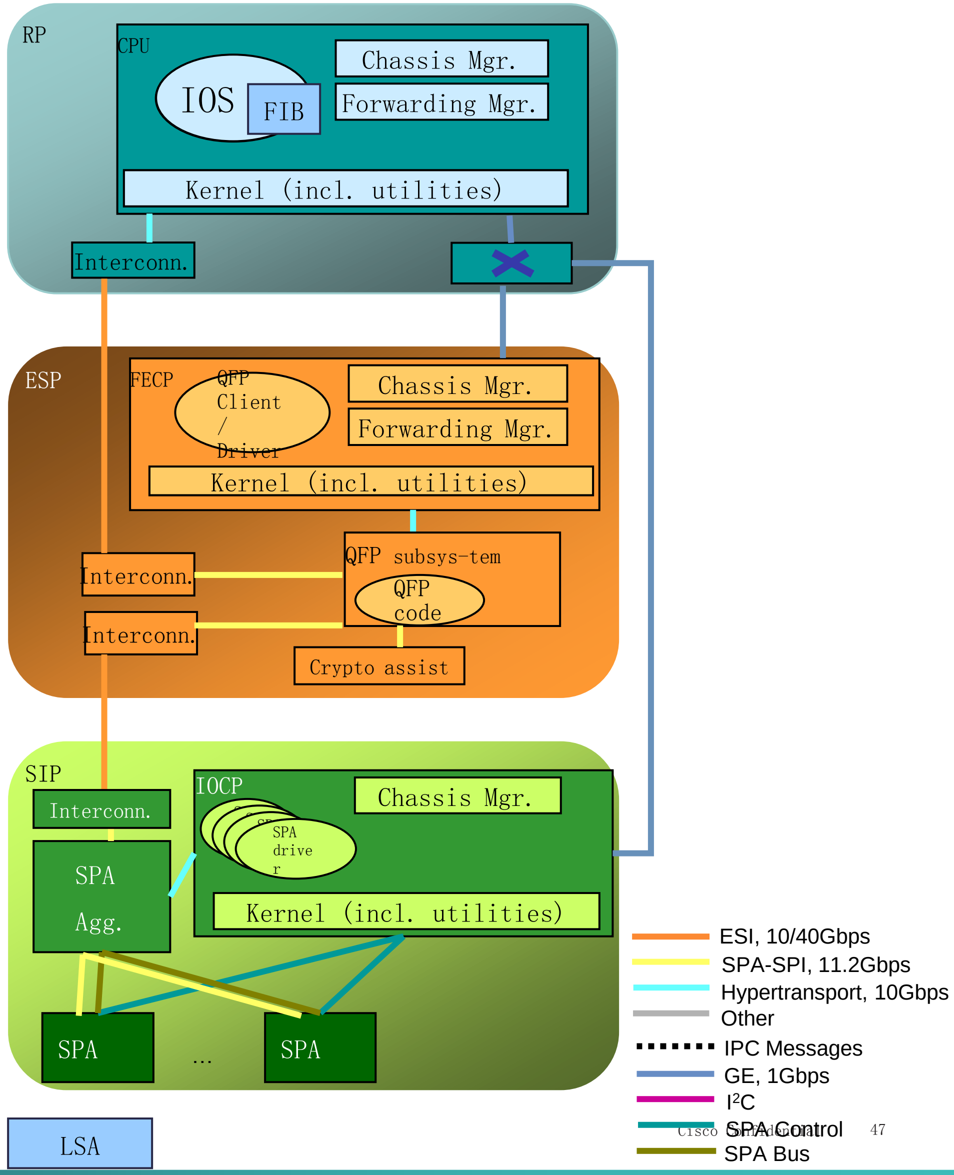
# Data Packet Flow: Through SIP to SPA



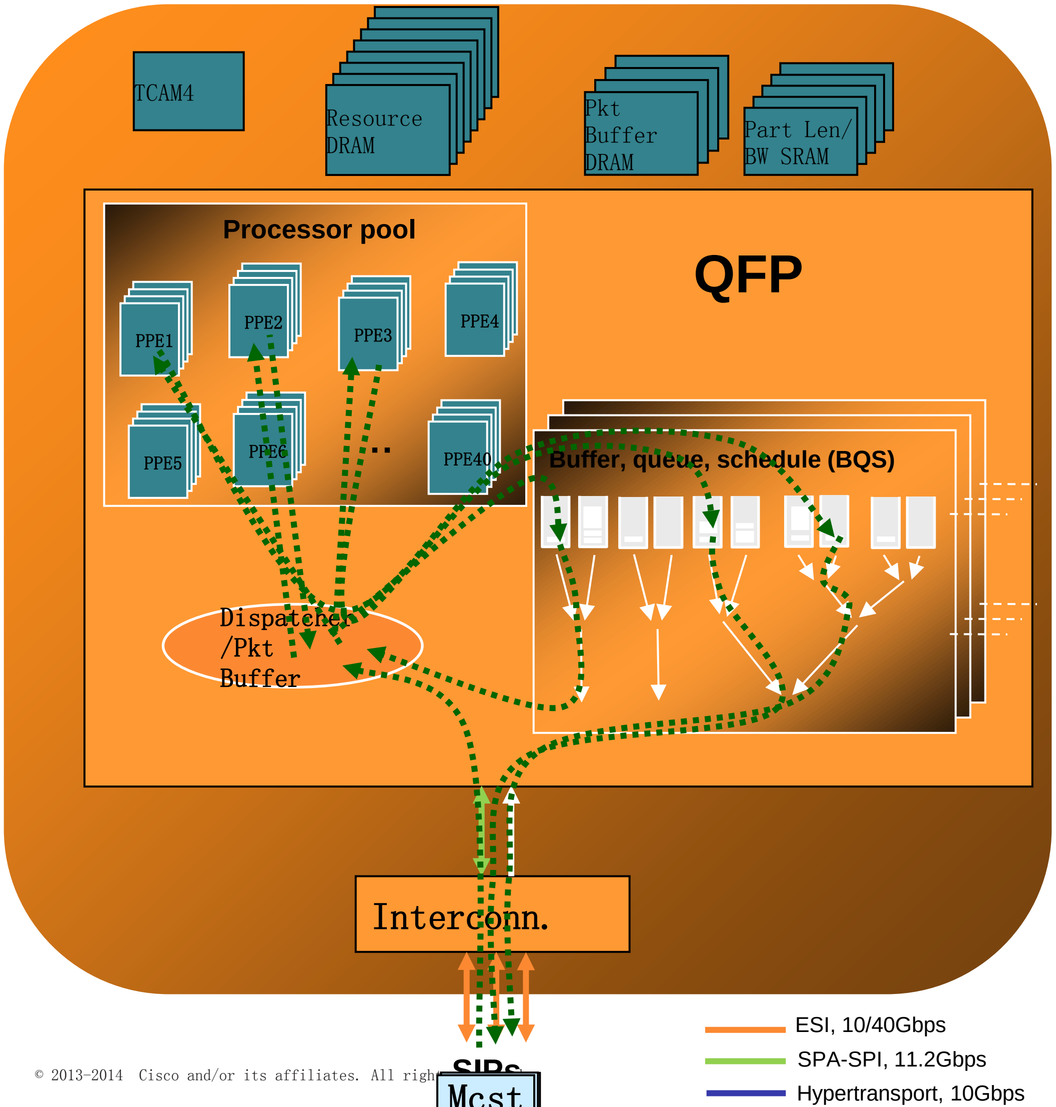

1. Interconnect receives packet data over ESI from the active ESP at up to 40 Gbps
2. SPA Aggregation ASIC receives the packet and writes it to external egress buffer memory
3. Egress buffer memory is carved into 64 queues. The queues are arranged by egress SPA-SPI channel and optionally H/L. Channels on “channelized” SPAs share
the same queue.
4. SPA Aggregation ASIC selects and transfers packet data from eligible queues to SPA-SPI channel (Hi queue are selected before Low)
5. SPA can backpressure transfer of packet data burst independently for each SPA-SPI channel using SPI FIFO status
6. SPA transmits packet data on network interface

# ASR 1000 Control Packet Flow

- By example of OSPF LSA
- OSPF LSA arrives on the SPA and is forwarded to the SIP
- SIP performs ingress H/L classification and sends packet to ESP
- QFP receives OSPF LSA and sends to a PPE for processing
- PPE executes features and realizes this is an OSPF LSA
- PPE marks internal header to forward packet to the RP
- PPE releases OSPF LSA to BQS
- BQS Scheduler sends packet to RP
- RP receives packet over ESI link and sends to IOS
- IOS Processes OSPF LSA and performs SPF
- IOS updates RIB/FIB and sends to $FM_{RP}$
- $FM_{RP}$ keeps copy of FIB and sends also down to $FM_{ESP}$
- $FM_{ESP}$ keeps a copy of the FIB and programs QFP



# Multicast Packet Flow Through ESP




1. Packet arrives on QFP
2. Packet assigned to a PPE thread.

   **Input Features applied**

   Netflow, MQC/NBAR Classify, FW, RPF, Mark/Police, NAT, WCCP etc.

3. Packet replicated
4. The PPE thread processes the packet in a feature chain similar to 12.2S IOS (very basic view of a v4 packet):

   **Forwarding Decision is made**

   IPv4 / IPv6 MFIB

   **Output Features applied**

   Netflow, FW, NAT, Crypto, MQC/NBAR Classify, Police/Mark etc.

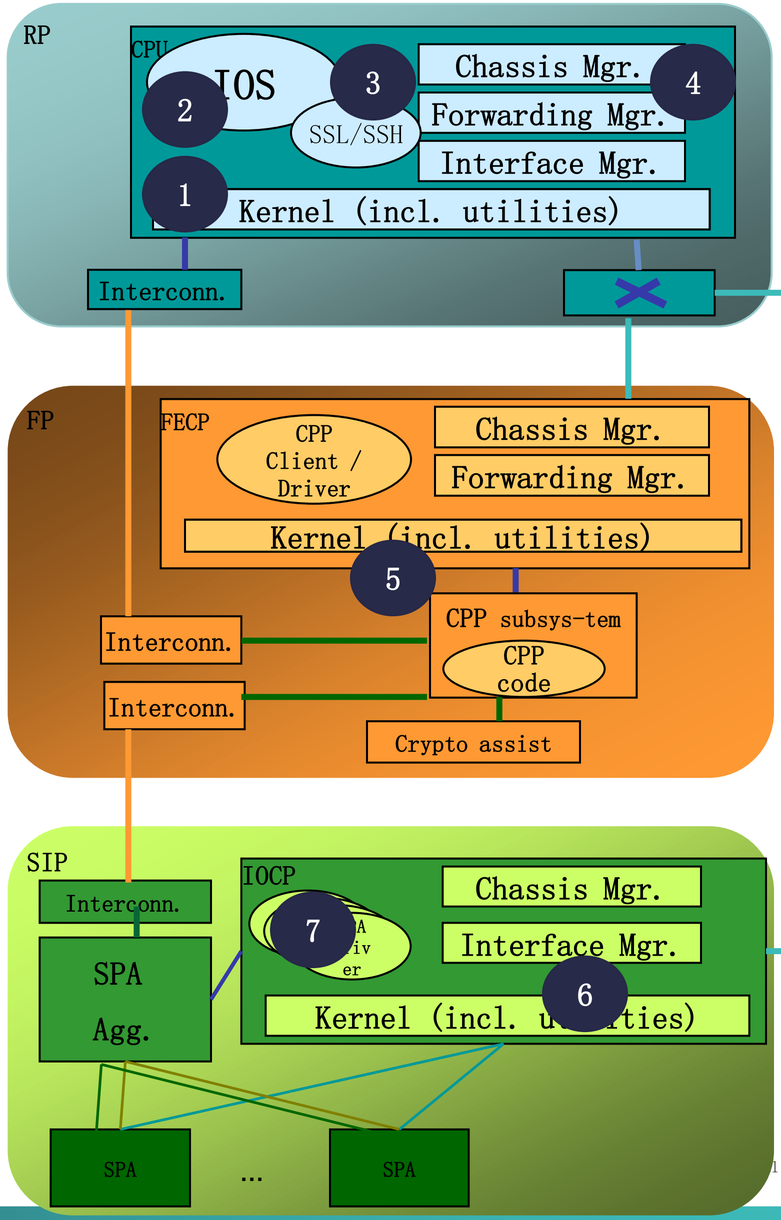
   **Finished**

5. Packet released from on-chip memory to Traffic Manager (Queued)
6. The Traffic Manager schedules which traffic to send to which SIP interface (or RP or Crypto Chip) based on priority and what is configured in MQC
7. SIP can independently backpressure ESP via ESI control message to pace the packet transfer if needed

# IOS XE Releases and Packaging for ASR 1000

# Software Sub-packages

1. **RPBase: RP OS**
   **Why?: Upgrading of the OS will require reload to the RP and expect minimal changes**
2. **RPIOS: IOS**
   **Why?: Facilitates Software Redundancy feature**
3. **RPAccess (K9 & non-K9): Software required for Router access; 2 versions will be available. One that contains open SSH & SSL and one without**
   **Why?: To facilitate software packaging for export-restricted countries**
4. **RPControl : Control Plane processes that interface between IOS and the rest of the platform**
   **Why?: IOS XE Middleware**
5. **ESPBase: ESP OS + Control processes + QFP client/driver/ucode:**
   **Why?: Any software upgrade of the ESP requires reload of the ESP**
6. **SIPBase: SIP OS + Control processes**
   **Why?: OS upgrade requires reload of the SIP**
7. **SIPSPA: SPA drivers and FPD (SPA FPGA image)**
   **Why?: Facilitates SPA driver upgrade of specific SPA slots**



# Cisco IOS XE Images for Enterprise and Managed Services - CPE

## Optional Features

**Cisco ASR 1000 Series Feature Licenses**

- SW Redundancy
- SBC
- IPSec
- Firewall
- Flexible Packet Inspection

**Cisco ASR1000 Series IP Base w/o Crypto (SASR1R1-IPB)**

- BGP, EIGRP, ISIS, OSPF, RIP
- ACL
- HSRP/VRRP
- HA: BFD, ISSU
- NAT
- Netflow
- QoS, WCCPv2
- IPv6 (rls5)

**Cisco ASR1000 Series IP Base (SASR1R1-IPBK9)**

- SSL, SSH

- BGP, EIGRP, ISIS, OSPF, RIP
- ACL
- HSRP/VRRP
- HA: BFD, ISSU
- NAT
- Netflow
- QoS, WCCPv2
- IPv6 (rls5)

**Cisco ASR1000 Series RP1 Advanced Enterprise Services w/o Crypto (SASR1R1-AES)**

- Legacy – IPX, Appletalk, DecNet, etc

- Broadband
- L2 & L3 VPN
- MPLS
- IPv6
- ATOM, VPLS
- PfR
- Multicast
- SBC

- BGP, EIGRP, ISIS, OSPF, RIP
- ACL
- HSRP/VRRP
- NAT
- HA: BFD, ISSU
- Netflow
- QoS, WCCPv2

**Cisco ASR1000 Series RP1 Advanced Enterprise Services (SASR1R1-AESK9)**

- Legacy – IPX, Appletalk, DecNet, etc

- Broadband
- L2 & L3 VPN
- MPLS
- IPv6
- ATOM, VPLS
- PfR
- Security, LI
- Multicast
- SBC

- SSL, SSH

- BGP, EIGRP, ISIS, OSPF, RIP
- ACL
- HSRP/VRRP
- NAT
- HA: BFD, ISSU
- Netflow
- QoS, WCCPv2

# Cisco IOS XE Images for Service Providers

## Optional Features

### Cisco ASR 1000 Series Feature Licenses

- SW Redundancy
- SBC
- Flexible Packet Inspection
- BB subscribers

### Cisco ASR1000 Series IP Base w/o Crypto (SASR1R1-IPB)

- BGP, EIGRP, ISIS, OSPF, RIP
- ACL
- HSRP/VRRP
- HA: BFD, ISSU
- NAT
- Netflow
- QoS, WCCPv2
- IPv6 (rls5)

### Cisco ASR1000 Series IP Base (SASR1R1-IPBK9)

- SSL, SSH

- BGP, EIGRP, ISIS, OSPF, RIP
- ACL
- HSRP/VRRP
- HA: BFD, ISSU
- NAT
- Netflow
- QoS, WCCPv2
- IPv6 (rls5)

### Cisco ASR1000 Series RP1 Advanced IP Services w/o Crypto (SASR1R1-AIS)

- Broadband
- L2 & L3 VPN
- MPLS
- IPv6
- ATOM, VPLS
- PfR
- Multicast
- SBC

- BGP, EIGRP, ISIS, OSPF, RIP
- ACL
- HSRP/VRRP
- HA: BFD, ISSU
- NAT
- Netflow
- QoS, WCCPv2

### Cisco ASR1000 Series RP1 Advanced IP Services (SASR1R1-AISK9)

- Broadband
- L2 & L3 VPN
- MPLS
- IPv6
- ATOM, VPLS
- PfR
- Security, LI
- Multicast
- SBC

- SSL, SSH

- BGP, EIGRP, ISIS, OSPF, RIP
- ACL
- HSRP/VRRP
- HA: BFD, ISSU
- NAT
- Netflow
- QoS, WCCPv2

# ASR1000 feature sets

| IP Base | | |
|---|---|---|
| ACL | BGP | EIGRP |
| ISIS | OSPF | RIP |
| EEM | ERSPAN | ISSU |
| HSRP | VRRP | GLBP |
| Multicast | NAT | NBAR |
| Netflow | PPPoE client | SNMP |
| TACACS | All intf | IPSLA |
| IPv6 parity to IPv4 features | | LI |

| Advanced IP services | |
|---|---|
| All IP Base features | |
| BFD | |
| Broadband (BNG / ISG) | |
| CUBE (SP) | CUBE (Ent) |
| Firewall | L2 & L3 VPN |
| MPLS | OTV |
| PfR | LISP |

| Advanced enterprise services | |
|---|---|
| All IP services features | |
| DECNet V | |
| IPX | |

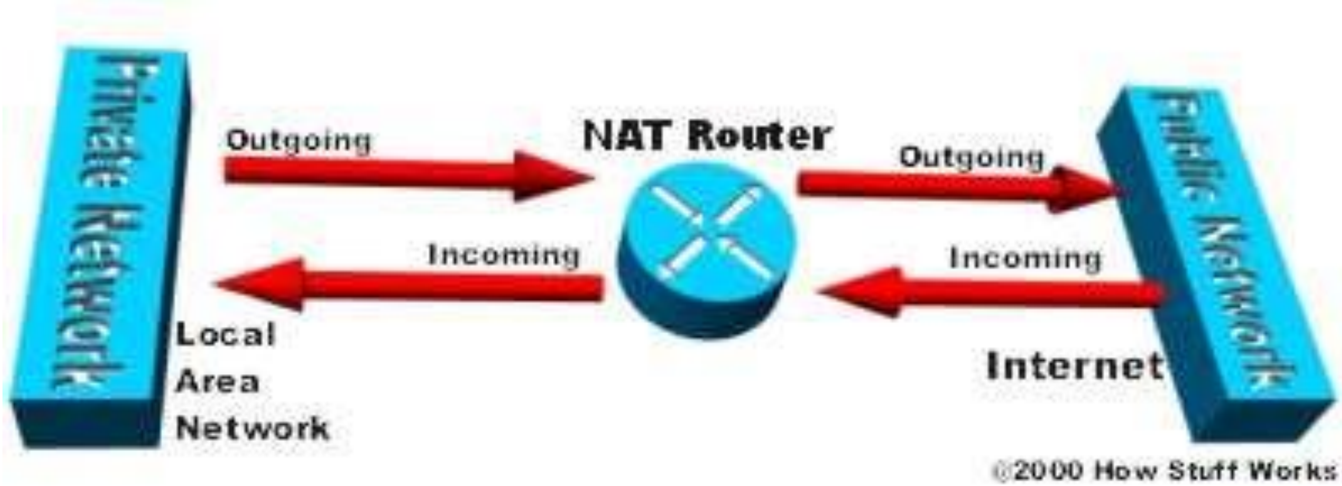
Some of the features require Feature Licenses in addition to the software image

Data current to IOS XE3.7. Always check Cisco Feature Navigator for the most up to date information regarding features included in releases and feature sets.

# NAT/ZBFW and HA

# ASR1K NAT

At the most basic level NAT is intended to connect a private network using private IP addresses to a public network using public IP addresses. NAT will map the private IP addresses into public IP address(es) based on configuration. Below is a depiction of this most common topology.




The below picture is intended to help explain some commonly used NAT terminology.

Inside Local Address
Inside Global Address
Outside Global Address
Outside Local Address

# NAT Mapping Configuration

**Inside Static Mapping**

ip nat inside source static 13.1.1.2 12.1.1.2

**Static Inside Network Mapping**

ip nat inside source static network 13.1.1.0 12.1.1.0 /24

**Outside Static Mapping**

ip nat outside source static 12.1.1.1 16.1.1.1 add

**Inside Dynamic Mapping Example**

access-list 1 permit 13.1.1.0 0.0.0.255
ip nat pool pool1 12.1.1.10 12.1.1.150 prefix-length 24
ip nat inside source list 1 pool pool1

**PAT - Inside Dynamic Overload Mapping Example**

ip nat pool pool1 18.1.1.1 18.1.1.254 pre 24
ip nat inside source list 1 pool pool1 overload
access-list 1 permit 13.1.0.0 0.0.255.255
ip route 18.1.1.0 255.255.255.0 gi0/0/1

# NAT TCP Session timeout-issue

By default TCP time out is 24 hours if router do nit received TCP-FIN packet

tcp 120.28.1.2:53762 172.16.1.1:53762 202.97.32.1:23 202.97.32.1:23
create: 09/03/14 20:26:14, use: 09/03/14 20:26:17, **timeout: 23:59:42**
Map-Id(In): 1
Appl type: none
Mac-Address: 0000.0000.0000 Input-IDB: GigabitEthernet0/0/1
entry-id: 0x90c9d2b0, use_count:1

tcp 120.28.1.2:53762 172.16.1.1:53762 202.97.32.1:23 202.97.32.1:23
create: 09/03/14 20:26:14, use: 09/03/14 20:27:02**, timeout: 00:00:57 <<= if TCP FIN received**
Map-Id(In): 1
Flags: timing-out
Appl type: none
Mac-Address: 0000.0000.0000 Input-IDB: GigabitEthernet0/0/1
entry-id: 0x90c9d2b0, use_count:1

# NAT Miscellaneous Configuration

**Changing Default Timeout Value Example**

ip nat translation tcp-timeout 1200

The following is the complete list of supported timeout:

```
ASR1002-3(config)#ip nat translation ?
  dns-timeout             Specify timeout for NAT DNS flows
  finrst-timeout          Specify timeout for NAT TCP flows after a FIN or RST
  icmp-timeout            Specify timeout for NAT ICMP flows
  max-entries             Specify maximum number of NAT entries
  port-timeout            Specify timeout for NAT TCP/UDP port specific flows
  pptp-timeout            Specify timeout for NAT PPTP flows
  routemap-entry-timeout  Specify timeout for routemap created half entry
  syn-timeout             Specify timeout for NAT TCP flows after a SYN and no further data
  tcp-timeout             Specify timeout for NAT TCP flows
  timeout                 Specify timeout for dynamic NAT translations
  udp-timeout             Specify timeout for NAT UDP flows
```

# EPC on ASR1K

**If customer use Main interface…….**

```
Extended IP access list tcp-capture
   10 permit tcp host 172.16.1.1 host 202.97.32.1
--
monitor capture tcptest interface gigabitEthernet 0/0/1 both access-list tcp-capture
monitor capture tcptest start
monitor capture tcptest stop
monitor capture tcptest export harddisk:tcptest.cap

ASR1004-1#show monitor capture tcptest buffer brief
 -------------------------------------------------------------
 #   size   timestamp     source             destination   protocol
 -------------------------------------------------------------
   0   56    0.000000   172.16.1.1       ->  202.97.32.1     TCP
   1   54    0.000992   172.16.1.1       ->  202.97.32.1     TCP
   2   56    0.160032   172.16.1.1       ->  202.97.32.1     TCP
   3   54    0.161039   172.16.1.1       ->  202.97.32.1     TCP
```

# Captured packets

```
26 6.153044   172.16.1.1          202.97.32.1          TELNET      56 Telnet Data ...
27 6.254037   172.16.1.1          202.97.32.1          TCP         54 14338→23 [ACK] Seq=
28 6.254037   172.16.1.1          202.97.32.1          TCP         54 14338→23 [ACK] Seq=
29 6.254037   172.16.1.1          202.97.32.1          TCP         54 14338→23 [FIN, PSH,

    Source Port: 14338 (14338)
    Destination Port: 23 (23)
    [Stream index: 0]
    [TCP Segment Len: 0]
    Sequence number: 27    (relative sequence number)
    Acknowledgment number: 134    (relative ack number)
    Header Length: 20 bytes
  ⊞ .... 0000 0001 1001 = Flags: 0x019 (FIN, PSH, ACK)
    Window size value: 3884
    [Calculated window size: 3884]
    [Window size scaling factor: -1 (unknown)]
```

# If it's sub-interface

ERSPAN could consider as a workaround

http://www.cisco.com/c/en/us/td/docs/ios-xml/ios/lanswitch/configuration/xe-3s/lanswitch-xe-3s-book/lnsw-conf-erspan.html

Key points:
1# need configure additional loopback interface
2# need connected PC to ASR1K to locally capture packets.
3# RESPAN is not supported on Layer 2 switching interface.

**DDTS CSCug04984  is working on EPC for sub-interface**

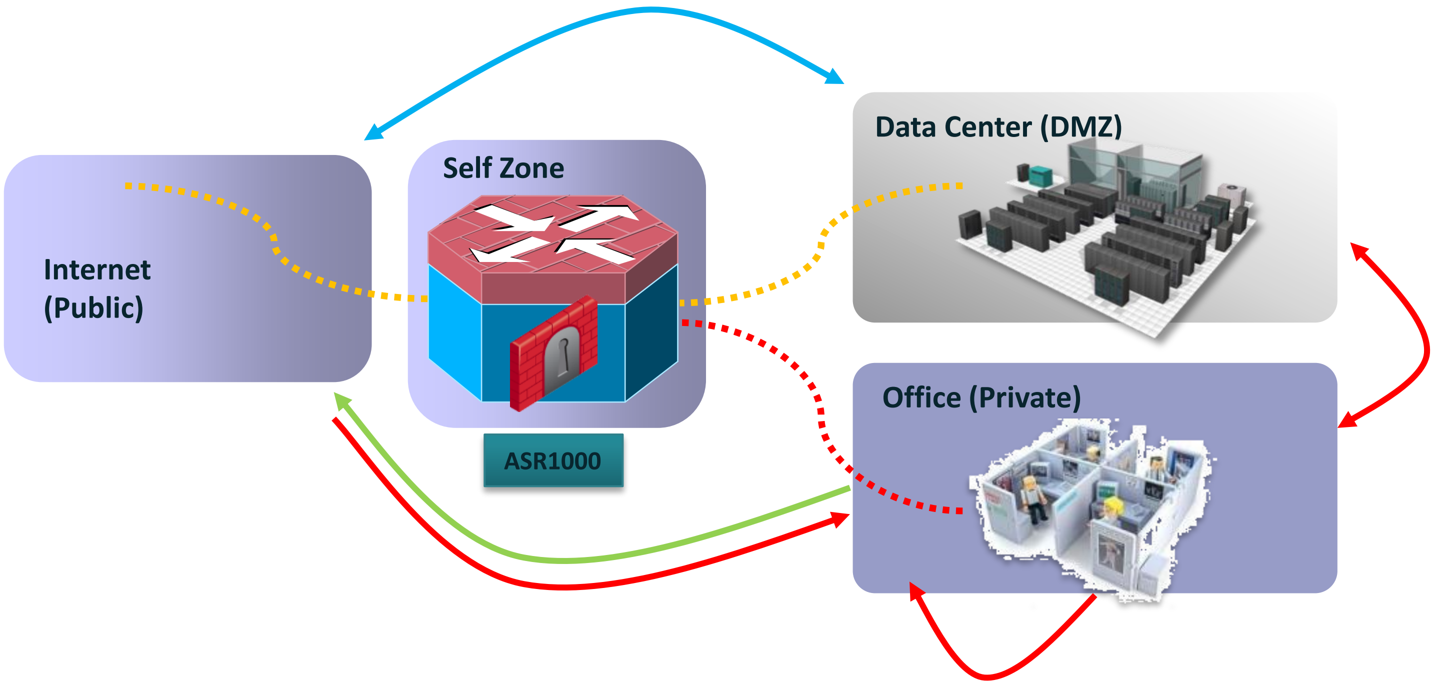
# How do NAT Limit Interact which each other

There is a hierarchy for NAT limits which goes like this from high to low:

```
ASR1004-1(config)#ip nat translation max-entries ?
  <1-2147483647>  Number of entries
  all-host        Specify maximum number of NAT entries for each host
  all-vrf         Specify maximum number of NAT entries for each vrf
  host            Specify per-host NAT entry limit
  list            Specify access list based NAT entry limit
  redundancy      Specify maximum number of NAT entries for RG
  vrf             Specify per-VRF NAT entry limit
```

Max-entries is special and always is applied to every creation. After we check max-entries, we proceed to see if any other limit applies to the packet. If it does, we stop there and do not proceed to any other limit. Thus if a user has a configuration like this:

# Zone Based Firewall



# Zone Based Firewall(Config Example)

### Step.1 Classify traffic

```
class-map type inspect match-any match-protocol
  match protocol ftp
  match protocol tcp
  match protocol udp
!
class-map type inspect match-any match-acl
 match access-group 181
```

### Step.2 Define actions in Policy map

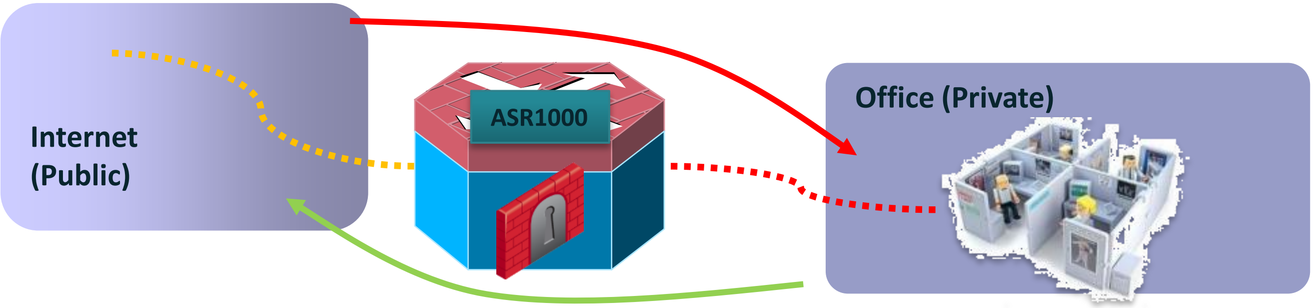
```
policy-map type inspect private-to-public
  class type inspect match-acl
     inspect
  !
  class type inspect match-protocol
    pass log
  !
  class type inspect class-default
    drop log
```

### Step.3 Define Security Zones

```
zone security public-zone
zone security private-zone
!
Interface GigabitEthernet 1/0/0
   Description ***connect-to-Internet***
   Zone-member security public-zone
!
Interface GigabitEthernet 1/1/0
   Description ***connect-to-private***
   Zone-member security private-zone
```

### Step.4 Define inter-Zone Rules

```
Zone-pair security private-to-public source private-zone destination
public-zone
  service-policy type inspect private-to-public
```



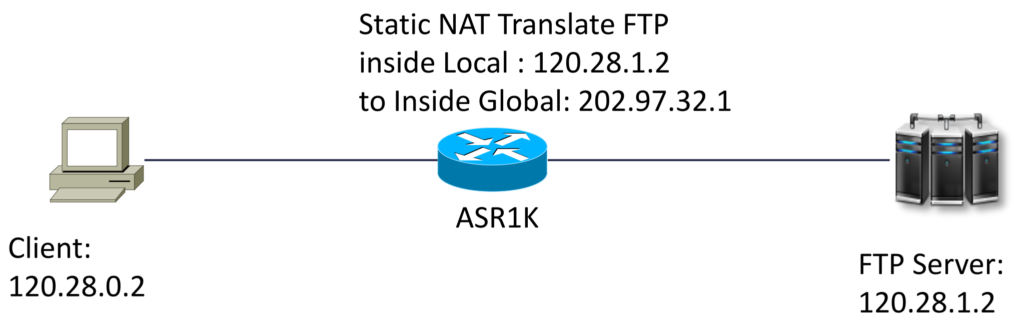
# Check ZBF session

```
ASR1004-1#show policy-map type inspect zone-pair test-in2out sessions
 Zone-pair: test-in2out
 Service-policy inspect : nat-test

  Class-map: nat-test (match-all)
   Match: access-group 2300
   Inspect
    Established Sessions
     Session 2ECA5D8 (172.16.1.1:56322)=>(202.97.32.1:23) telnet SIS_OPEN
      Created 00:55:02, Last heard 00:02:21
      Bytes sent (initiator:responder) [289:1234]


  Class-map: class-default (match-any)
   Match: any
   Drop    <<<<<===
    0 packets, 0 bytes
```

# Typical Scenario : NAT+ALG+ZBFW




Configuration example for static NAT:

```
ip nat inside source static 120.28.1.2 202.97.32.1 extendable
ip nat inside source static tcp 120.28.1.2 20 202.97.32.1 21 extendable
```

1, What is the difference for Active Mode and Passive Mode, and how would it impact Negotiation process between NAT Server and Client?

2, how FTP ALG works in this scenario, if we only configure TCP port 20 as static NAT, do we need also configure TCP port 20 as Data Channel? Why ?

3, if we configure ZBW, which ip address should be used (Inside Local? Inside Global? ) for the security access-list? Why ?

# NAT working Mode

## Active Mode

| | | | | | |
|---|---|---|---|---|---|
| 37 | 8.88884600 | 202.97.32.1 | 120.28.0.2 | FTP | 84 Response: 200 PORT command successfu |
| 38 | 8.88944400 | 120.28.0.2 | 202.97.32.1 | FTP | 74 Request: RETR ACL-ABF-2.tcl |
| 39 | 8.89435700 | 202.97.32.1 | 120.28.0.2 | FTP | 106 Response: 150 File status OK ; about |
| 40 | 8.89550600 | 202.97.32.1 | 120.28.0.2 | TCP | 66 20→64029 [SYN] Seq=0 Win=65535 Len=( |
| 41 | 8.89564600 | 120.28.0.2 | 202.97.32.1 | TCP | 66 64029→20 [SYN, ACK] Seq=0 Ack=1 Win= |
| 42 | 8.89622300 | 202.97.32.1 | 120.28.0.2 | TCP | 60 20→64029 [ACK] Seq=1 Ack=1 Win=25696 |

## Passive Mode

| | | | | | |
|---|---|---|---|---|---|
| 25 | 5.89703900 | 120.28.0.2 | 119.188.46.33 | TCP | 66 64015→80 [SYN] Seq=0 Win=8192 Len=0 MSS=12 |
| 26 | 6.13159800 | 120.28.0.2 | 119.188.46.33 | TCP | 66 64016→80 [SYN] Seq=0 Win=8192 Len=0 MSS=12 |
| 27 | 8.26609300 | 120.28.0.2 | 202.97.32.1 | FTP | 60 Request: PASV |
| 28 | 8.27759000 | 202.97.32.1 | 120.28.0.2 | FTP | 100 Response: 227 Entering passive mode (120,2 |
| 29 | 8.27854900 | 120.28.0.2 | 202.97.32.1 | FTP | 74 Request: RETR ACL-ABF-2.tcl |
| 30 | 8.27907500 | 120.28.0.2 | 120.28.1.2 | TCP | 66 64017→2296 [SYN] Seq=0 Win=65535 Len=0 MSS |
| 31 | 8.27991500 | 120.28.1.2 | 120.28.0.2 | TCP | 66 2296→64017 [SYN, ACK] Seq=0 Ack=1 Win=6553 |
| 32 | 8.28002600 | 120.28.0.2 | 120.28.1.2 | TCP | 54 64017→2296 [ACK] Seq=1 Ack=1 Win=4194304 L |
| 33 | 8.28443500 | 202.97.32.1 | 120.28.0.2 | FTP | 90 Response: 125 Using existing data connecti |

| | | | | | |
|---|---|---|---|---|---|
| 27 | 8.26609300 | 120.28.0.2 | 202.97.32.1 | FTP | 60 Request: PASV |
| 28 | 8.27759000 | 202.97.32.1 | 120.28.0.2 | FTP | 100 Response: 227 Entering passive mode (120,28,1,2,8,248) |
| 29 | 8.27854900 | 120.28.0.2 | 202.97.32.1 | FTP | 74 Request: RETR ACL-ABF-2.tcl |
| 30 | 8.27907500 | 120.28.0.2 | 120.28.1.2 | TCP | 66 64017 > theta-lm [SYN] Seq=0 Win=65535 Len=0 MSS=1260 WS=128 |
| 31 | 8.27991500 | 120.28.1.2 | 120.28.0.2 | TCP | 66 theta-lm > 64017 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1 |
| 32 | 8.28002600 | 120.28.0.2 | 120.28.1.2 | TCP | 54 64017 > theta-lm [ACK] Seq=1 Ack=1 Win=4194304 Len=0 |
| 33 | 8.28443500 | 202.97.32.1 | 120.28.0.2 | FTP | 90 Response: 125 Using existing data connection |
| 34 | 8.33558500 | 120.28.1.2 | 120.28.0.2 | FTP-DAT | 1078 FTP Data: 1024 bytes |
| 35 | 8.33640900 | 120.28.1.2 | 120.28.0.2 | FTP-DAT | 1314 FTP Data: 1260 bytes |
| 36 | 8.33659600 | 120.28.0.2 | 120.28.1.2 | TCP | 54 64017 > theta-lm [ACK] Seq=1 Ack=2285 Win=4194304 Len=0 |

```
⊞ Ethernet II, Src: Cisco_72:bb:14 (00:22:55:72:bb:14), Dst: Vmware_17:f2:32 (00:0c:29:17:f2:32)
⊞ Internet Protocol Version 4, Src: 202.97.32.1 (202.97.32.1), Dst: 120.28.0.2 (120.28.0.2)
⊞ Transmission Control Protocol, Src Port: ftp (21), Dst Port: 64003 (64003), Seq: 1, Ack: 7, Len: 46
⊟ File Transfer Protocol (FTP)
  ⊟ 227 Entering passive mode (120,28,1,2,8,248)\r\n
      Response code: Entering Passive Mode (227)
      Response arg: Entering passive mode (120,28,1,2,8,248)
      Passive IP address: 120.28.1.2 (120.28.1.2)
      Passive port: 2296
      Passive IP NAT: True
```

# How ZBF co-work with NAT in FTP ALG Mode

Active Mode

```
ASR1002-1#show policy-map type inspect zone-pair sessions
 Zone-pair: out2in
 Service-policy inspect : fw-nat

   Class-map: fw-nat (match-all)
     Match: access-group name fw-nat
     Inspect
       Established Sessions
        Session 11D50E3C (120.28.0.2:49366)=>(120.28.1.2:21) ftp SIS_OPEN
         Created 00:00:14, Last heard 00:00:14
         Bytes sent (initiator:responder) [42:125]
        Session 11D50E88 (120.28.0.2:49367)=>(120.28.1.2:21) ftp SIS_OPEN
         Created 00:00:09, Last heard 00:00:08
         Bytes sent (initiator:responder) [97:225]

       Pre-Generating Sessions
        Session 11D50E3C (202.97.32.1:0)=>(120.28.0.2:49368) ftp-data   SIS_PREGEN   <<<== Is this
Correct ?? Why ?
         Created 00:00:08, Last heard 00:00:08
         Bytes sent (initiator:responder) [0:0]
```

# How to Debug

Enable parameter Map:

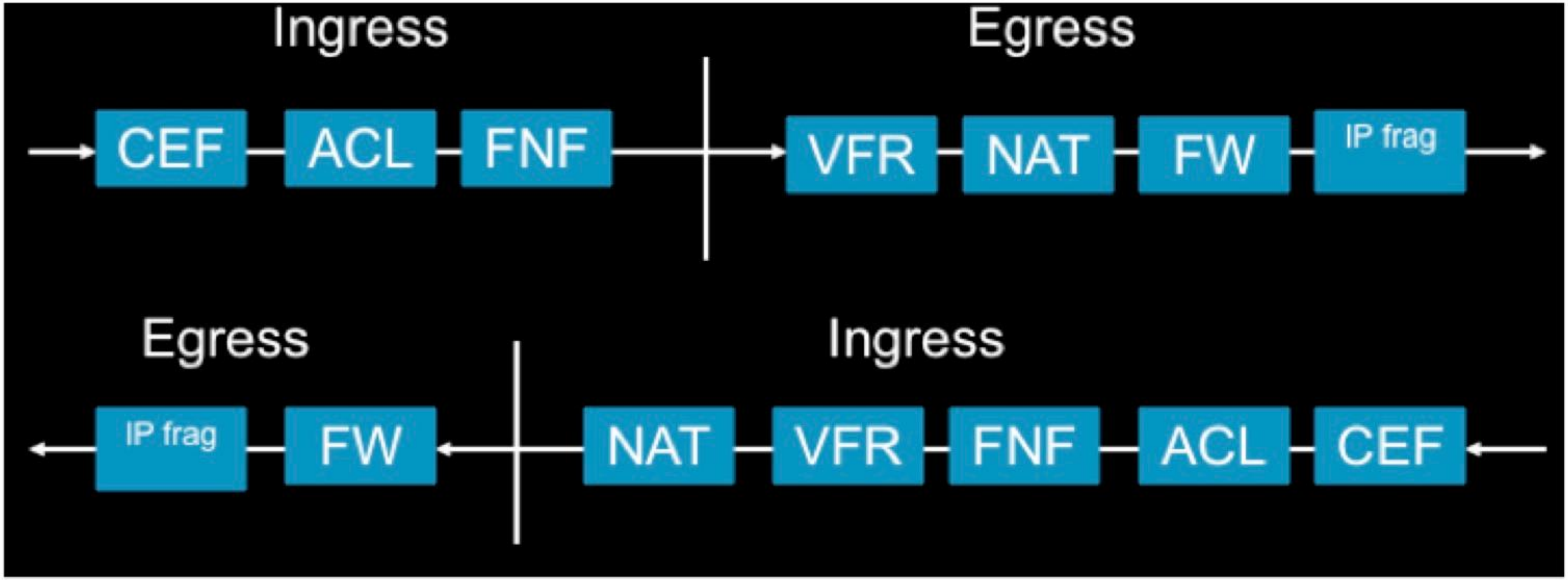
```
ASR1002-1#show parameter-map type inspect-global
parameter-map type inspect-global
 log dropped-packet on  <<<<==
 alert on
 aggressive aging disabled
 syn_flood_limit  unlimited
 tcp window scaling enforcement loose off
 max incomplete unlimited  aggressive aging disabled
 max_incomplete TCP unlimited
 max_incomplete UDP unlimited
 max_incomplete ICMP unlimited
 application-inspect all    <<<<==
```

Debug Output:

```
Sep 24 06:56:45.770: %IOSXE-6-PLATFORM: F0: cpp_cp: QFP:0.0 Thread:052
TS:00000452717812126627 %FW-6-DROP_PKT: Dropping tcp pkt from GigabitEthernet0/3/1 120.28.1.2:20 =>
120.28.0.2:54873(target:class)-(none:none) due to Zone-pair without policy   <<<<===
 with ip ident 1557 tcp flag 0x2, seq 2406560295, ack 0
```

# Packet Flow

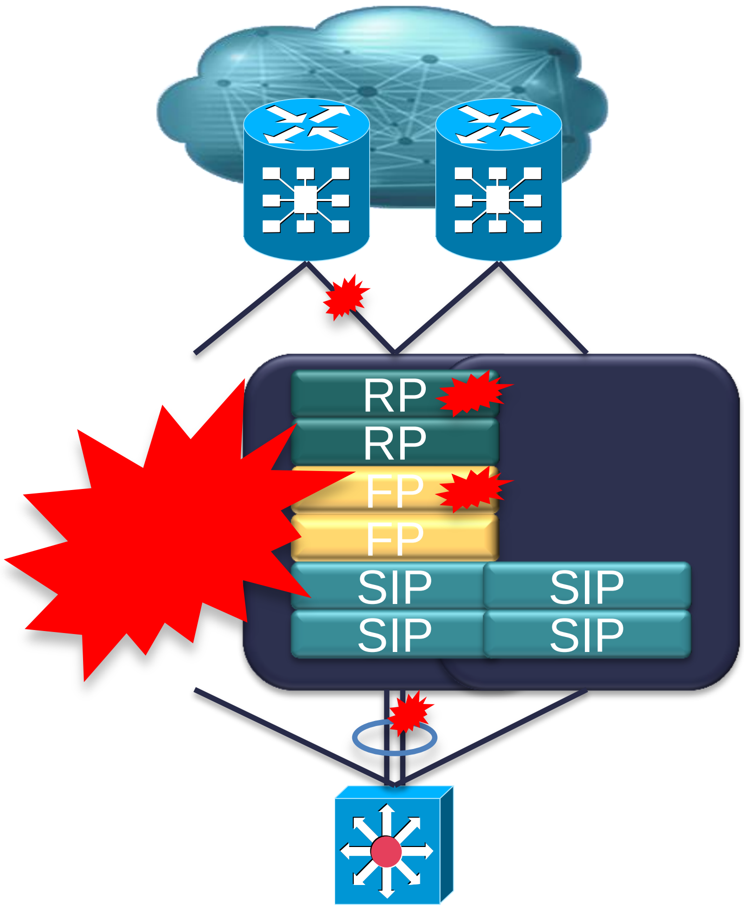



In QFP, feature order and execution are determined by Feature Invocation Arrays (FIAs). Zone based Firewall policies are associated with egress interface by virtue of a security zone pair. So Firewall processing happens on the egress path of packet processing.

# Inter-Chassis Redundancy

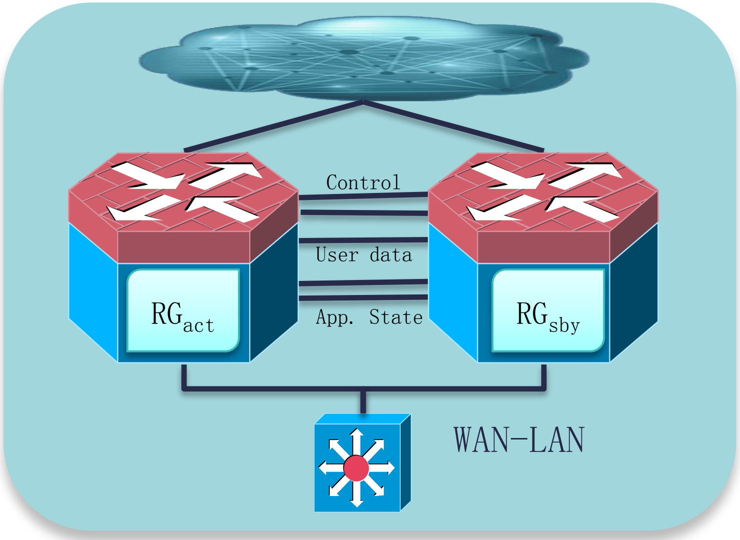
# Motivation for Stateful Application Inter-Chassis Redundancy

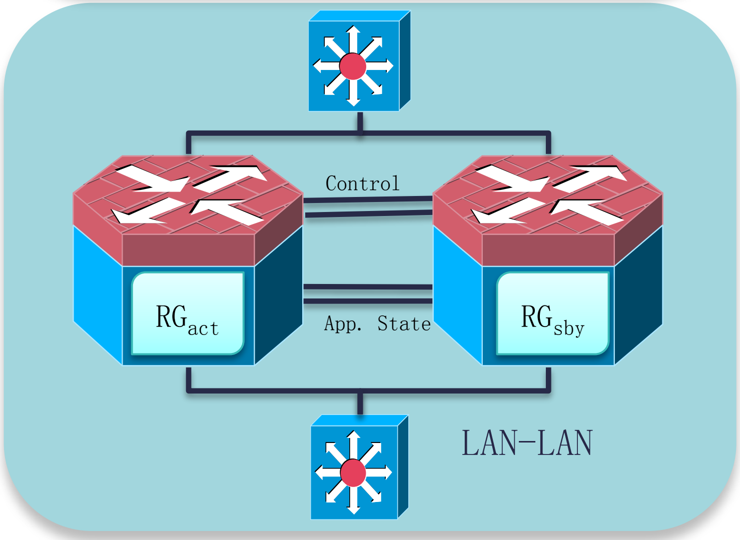
- Current Intra-chassis HA typically protects against
  - Control Plane (RP) Failures
  - Forwarding Plane (ESP) failures
  - Interface failures can be mitigated using link bundling (e.g. GEC)
- Any other failures may result in recovery times O(hours)
- Inter-chassis redundancy provides additional resilience against
  - Interface Failures
  - System failures
  - Site failures (allowing for geographic redundancy)



# Physical Topology Requirements

- 2 ASR 1000 chassis with single RP / single ESP
  - Co-existence of inter-chassis and intra-chassis redundancy currently NOT supported
  - Clusters with more than 2 members currently NOT supported
- Physical connectivity to both member systems from adjacent routers / switches
  - Need a mechanism to direct traffic to either member system in case of failover
- L1/L2 Connectivity between the two member systems for RG control traffic
  - Used by the 2 RG instances to exchange control traffic (RG hellos, RG state, fail-over signaling etc.)
  - Need guaranteed communication between the two member systems to avoid split-brain condition
  - L3 connectivity on roadmap
- L1/L2 Connectivity between the two member systems for Application state data
  - Synchronization of NAT/Firewall/SBC state tables
  - NOTE: FIBs are NOT synchronized by RG Infra
- Possible a user data cross-connect for asymmetric routing cases



# Configuring an RG Instance

- Multiple RGs can be instantiated on a single system
  - Currently allowing up to 2 instances
- Option 'Name' is description
- Option 'Preempt' allows for the standby to attract control and become 'active'
  - Otherwise the standby will not transition to active even if Priority($RG_{sby}$)<Priority($RG_{act}$)
- Option 'Priority' determines RG state
  - Higher priority 'wins'
  - Priorities can be decremented by failures
  - Priority change may or may not trigger failover (depending on Threshold)
  - 'Threshold' allows for event dampening. Threshold must be exceeded before a failover is initiated
- For each RG instance, need to specify which interface is used to exchange RG control and Application state data
  - Currently MUST be a physical interface
  - Cannot be a user data interface

```
redundancy
 mode none
 application redundancy
  group 1
   name GENERIC-NAT-FW
   preempt
   priority 50
   control Gig0/3/0 protocol 1
   data Gig0/3/1
group 2
   name VRFawareNAT-FW
   priority 150 failover threshold 100
   control Gig0/3/0 protocol 2
   data Gig0/3/1
```

# Notes on RG Configuration

- Two RG instances are implicitly linked by the same RG Number
- The RG starts to announce its present and RG state information out of the RG control interface
- If a peer-RG is found (same instance number), then RG control information is exchanged
  - RG State determination based on priorities
  - Periodic 'Hello' s' to monitor the health of the peer
    - Hello Timer defaults: 3 sec Hello, 10 sec holddown
    - BFD on control interface recommended for fast peer failure detection
- Initial Priority can range between 1-255
- RG Control and RG Data interfaces can be the same
  - Looking to support VLANs in future
- Option 'Protocol' allows for the specification of RG control Protocol timers
  - E.g. protocol 1
  - hellotime 5 holdtime 15
  - authentication md5 key-string d00b4r987654321a

# What failures are tracked?

- An RG Instance monitors the hardware of the member system
  - ESP / RP / RG Control link / RG Data link
- A single RG instance can track multiple objects!
- Interfaces can be tracked explicitly by the command 'redundancy group <#>'
  - Need to configure a VIP
  - Option 'decrement' indicates by how much to change the priority of the local RG instance upon failure
  - If RG Priority becomes lower than that of peer, a failover is initiated
- Interfaces on member systems can be associated with each other using a 'remote interface identifier' (RII)
  - No need to match Gig0/0/1 on member A with Gig0/0/1 on member B
  - RII can range from 1 to 65535
  - RIIs must match to pair-up interfaces on different member systems

```
Interface Ten/0/0.4
 encapsulation dot1Q 4
 ip address 75.3.1.1 255.255.0.0
 ip nat outside
 ip nbar protocol-discovery
 ip flow egress
 ip virtual-reassembly
 zone-member security public3
 redundancy rii 1003
 redundancy group 1 ip 75.3.1.255 exclusive
 decrement 10
 service-policy output parent3
!
ip nat pool VPN152 160.1.0.0 160.1.255.255
 netmask 255.255.0.0
ip nat inside source list vpn152 pool VPN152
 redundancy 1 mapping-id 152
```

# A Note on Object Tracking

Object tracking can also be associated with an RG Instance

- Object is defined using track object command like any other enhanced object definition.

A user may choose to shutdown a RG if the tracked object goes down instead of just changing priority.

- track <object-number> [decrement <1-255> | shutdown]

```
track 1 interface Gig0/0/0 line-protocol

redundancy
 mode sso
 application redundancy
  group 1
   preempt
   control Gig2/1/0 protocol 1
   data Gig2/1/0
   track 1 decrement 30
```

# State Synchronization

- Configuration indicates which application state has to be synchronized
  - E.G. firewall, configuring redundancy 'group <ID>' under a zone member
  - E.g. NAT, configuring redundancy group under the NAT inside pool with 'mapping ID'
- State is synchronized between RG instances across the RG Data link for the applications that is associated with the RG
  - For TCP sessions, need to have 3-way handshake completed
  - For UDP sessions, need to receive at least 2 packets in the same flow
- Following state is NOT synchronized
  - FIB
  - HTTP NAT Sessions
    - Configure 'ip nat switchover replication http' if sync is required
  - Half-Open FW sessions
  - Configuration file

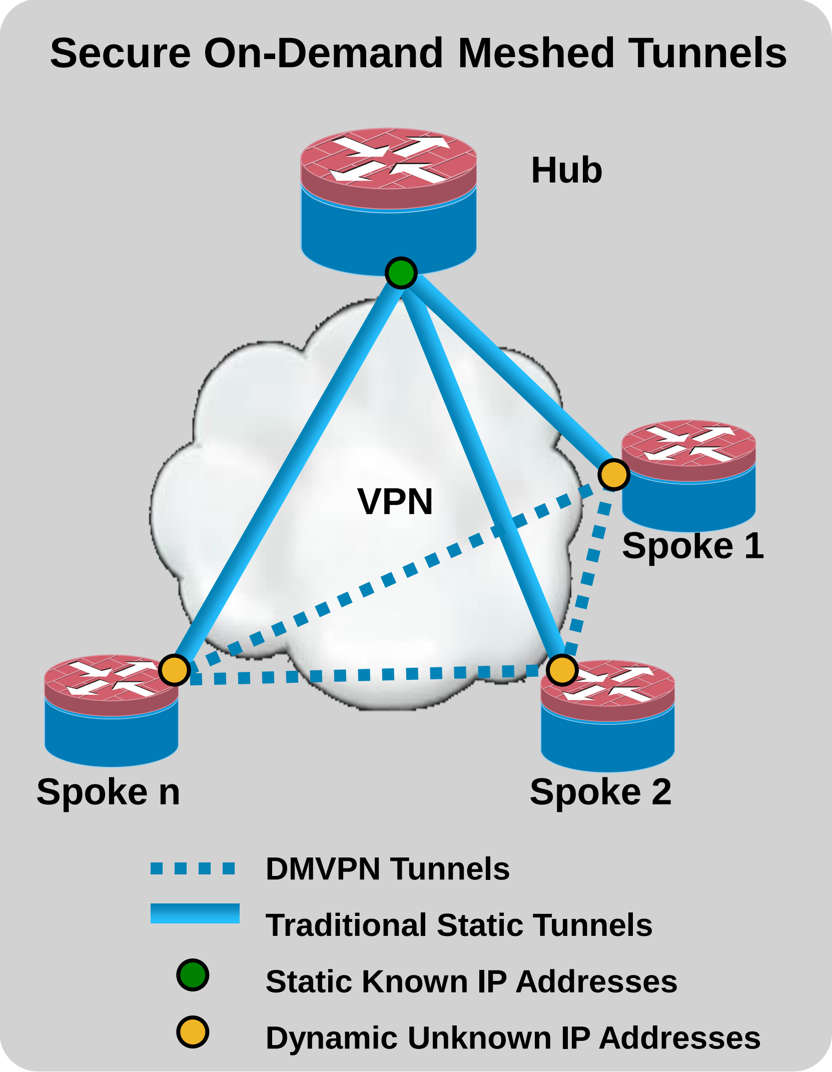
# Triggers for Failover

- Hardware failures
  - System / ESP / RP / Power
- Manual shut down of System Modules
- Manual shut-down or reload of an RG instance
  - Reload: ‘redundancy application reload group <rg-number> self’
    - Shutdown: 4RU(config-red-app)#group 1
    - 4RU(config-red-app-grp)#shutdown
- Control interface for RG is shutdown/link down
- Data Interface for RG is shutdown/link down
- Application Failure
- Tracked Object Failure
- Keepalive Failure (holddown time, BFD)
- Priority($RG_{act}$)<Priority($RG_{sby}$)
  - Run-time priority, not configured priority!

# DMVPN

# Dynamic Multipoint VPN

- Provides full meshed connectivity with simple configuration of hub and spoke
- Supports dynamically addressed spokes
- Facilitates zero-touch configuration for addition of new spokes
- Features automatic IPsec triggering for building an IPsec tunnel



# What Is Dynamic Multipoint VPN?

- DMVPN is a Cisco IOS Software solution for building IPsec+GRE VPNs in an easy, dynamic and scalable manner
- DMVPN relies on two proven technologies
  - Next Hop Resolution Protocol (NHRP)
    - Creates a distributed (NHRP) mapping database of all the spoke's tunnel to real (public interface) addresses
  - Multipoint GRE Tunnel Interface
    - Single GRE interface to support multiple GRE/IPsec tunnels
    - Simplifies size and complexity of configuration

# DMVPN—How It Works

- Spokes have a dynamic permanent GRE/IPsec tunnel to the hub, but not to other spokes; they register as clients of the NHRP server
- When a spoke needs to send a packet to a destination (private) subnet behind another spoke, it queries the NHRP server for the real (outside) address of the destination spoke
- Now the originating spoke can initiate a dynamic GRE/IPsec tunnel to the target spoke (because it knows the peer address
- The spoke-to-spoke tunnel is built over the mGRE interface

# Dynamic Multipoint VPN (DMVPN) Major Features

- Configuration reduction and no-touch deployment
- IP unicast, IP multicast and dynamic routing protocols
- Spokes with dynamically assigned addresses
- NAT—spoke routers behind dynamic NAT and hub routers behind static NAT
- Dynamic spoke-spoke tunnels for scaling partial/full mesh VPNs
- Can be used without IPsec encryption
- VRFs—GRE tunnels and/or data packets in VRFs
- 2547oDMVPN—MPLS switching over tunnels
- QoS—aggregate; static/manual per-tunnel
- Transparent to most data packet level features
- Wide variety of network designs and options

# DMVPN Components

- **Next Hop Resolution Protocol (NHRP)**

  Creates a distributed (NHRP) mapping database of all the spoke's tunnel to real (public interface) addresses

- **Multipoint GRE Tunnel Interface (MGRE)**

  Single GRE interface to support multiple GRE/IPsec tunnels

  Simplifies size and complexity of configuration

- **IPsec tunnel protection**

  Dynamically creates and applies encryption policies

- **Routing**

  Dynamic advertisement of branch networks; almost all routing protocols (EIGRP, RIP, OSPF, BGP, ODR) are supported

# “Static” Spoke-Hub, Hub-Hub Tunnels

- **GRE, NHRP and IPsec configuration**
  - p-pGRE or mGRE on spokes; mGRE on hubs
- **NHRP registration**
  - Dynamically addressed spokes (DHCP, NAT,…)
- **Routing protocol, NHRP, and IP multicast**
  - On spoke-hub and hub-hub tunnels
- **Data traffic on spoke-hub tunnels**
  - All traffic for hub-and-spoke only networks
  - Spoke-spoke traffic while building spoke-spoke tunnels

# Dynamic Spoke-Spoke Tunnels

- **GRE, NHRP and IPsec configuration**
  - mGRE on both hub and spokes
- **Spoke-spoke unicast data traffic**
  - Reduced load on hubs
  - Reduced latency
    - Single IPsec encrypt/decrypt
- **On demand tunnel creates when need it**
- **NHRP resolutions and redirects**
  - Find NHRP mappings for spoke-spoke tunnels

# DMVPN Phases

| Phase 1 | Phase 2 | Phase 3 |
|---|---|---|
| • Hub and spoke functionality 12.2(13)T<br>• Simplified and smaller config for hub & spoke<br>• Support dynamically address CPE<br>• Support for multicast traffic from hub to spoke<br>• Summarize routing at hub | • Spoke to spoke functionality 12.3(4)T<br>• Single mGRE interface in spokes<br>• Direct spoke to spoke data traffic reduced load on hub<br>• Cannot summarize spoke routes on hub<br>• Route on spoke must have IP next hop of remote spoke | • Architecture and scaling 12.4(6)T<br>• Increase number of hub with same hub and spoke ratio<br>• No hub daisy-chain<br>• Spokes don't need full routing table<br>• OSPF routing protocol not limited to 2 hubs<br>• Cannot mix phase 2 and phase 3 in same DMVPN cloud |

# Four Layer Troubleshooting Methodology

# Before You Begin

- Sync up the timestamps between the hub and spoke
- Enable msec debug and log timestamps

    service timestamps debug date time msec

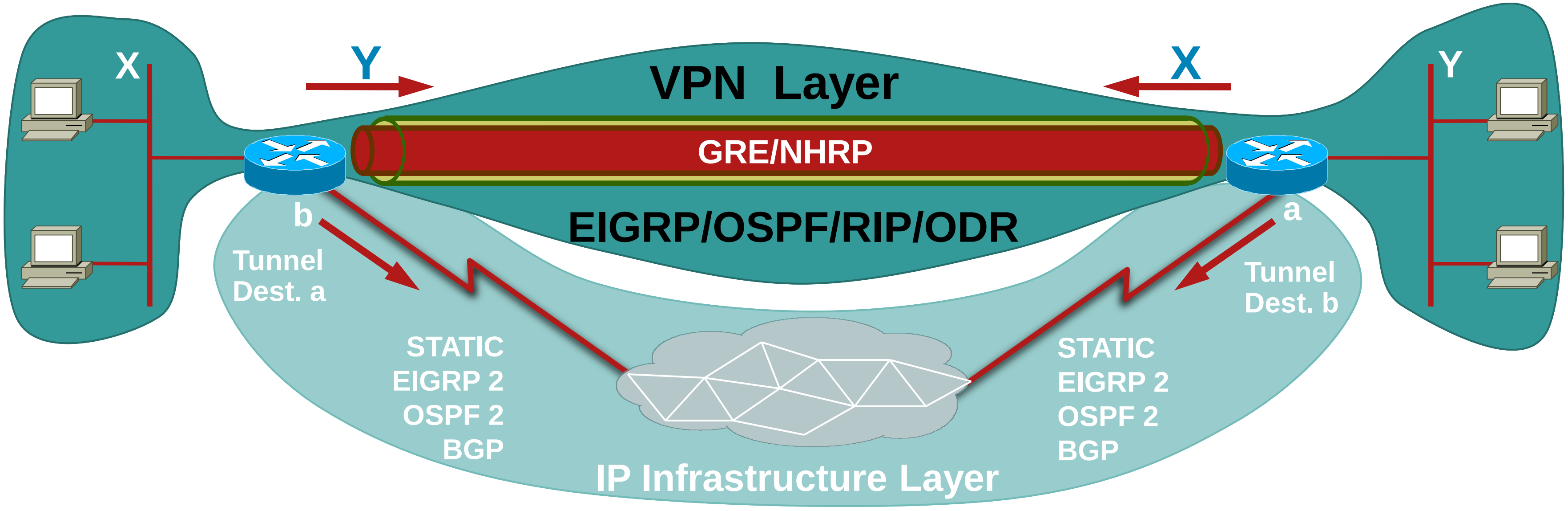
    service timestamps log date time msec

- Enable "terminal exec prompt timestamp" for the debugging sessions.

    This way you can easily correlate the debug output with the show command output

# Four Layer Troubleshooting Methodology

- Four layers for troubleshooting
  - Physical and routing layer
  - IPsec encryption layer—IPsec/ISAKMP
  - GRE encapsulation layer—NHRP
  - VPN routing layer—routing and IP data

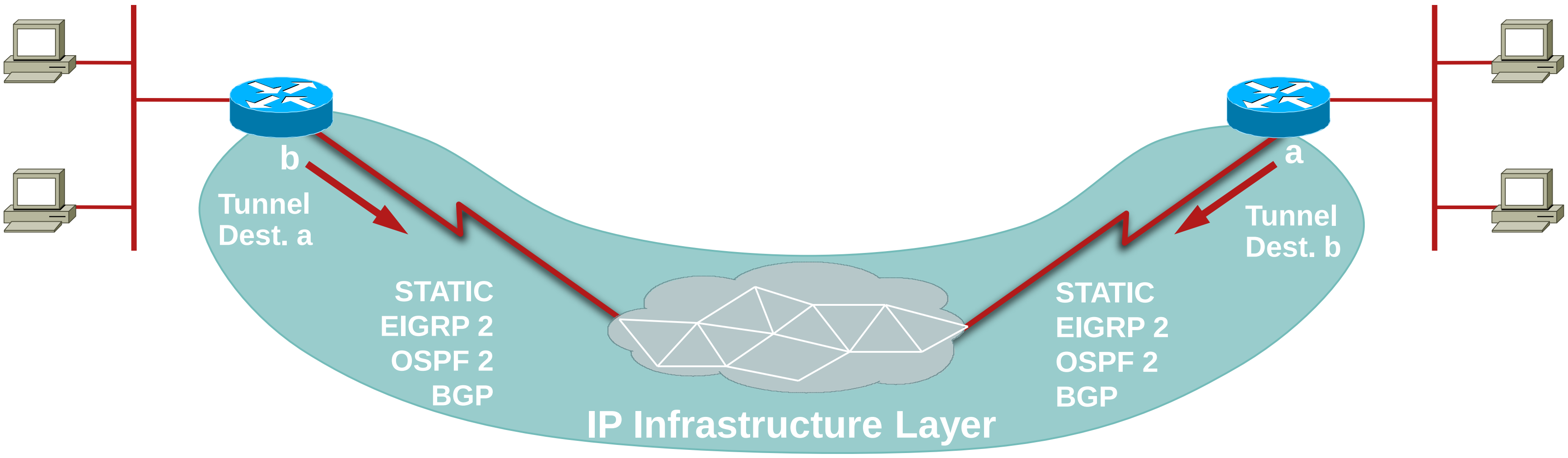


# Four Layers for Troubleshooting: Physical and Routing Layer

- Physical (NBMA or tunnel endpoint) routing layer

  This is getting the encrypted tunnel packets between the tunnel endpoints (DMVPN hub and spoke or between spoke and spoke routers)



# Four Layers for Troubleshooting: Physical and Routing Layer

- Ping from the hub to the spoke's using NBMA addresses (and reverse):
  - These pings should go directly out the physical interface, not through the DMVPN tunnel
  - Hopefully there isn't a firewall that blocks ping packets
  - If this doesn't work, check the routing and any firewalls between the hub and spoke routers
- Also use traceroute to check the path that the encrypted tunnel packets are taking
- Check for “administratively prohibited” (ACL) messages

# Four Layers for Troubleshooting: Physical and Routing Layer (Cont.)

- **Debugs and show commands use if no connectivity**

  **debug ip icmp**

  Valuable tool used to troubleshoot connectivity issues

  Helps you determine whether the router is sending or receiving ICMP messages

```
ICMP: rcvd type 3, code 1, from 172.17.0.1
ICMP: src 172.17.0.1, dst 172.16.1.1, echo reply
ICMP: dst (10.120.1.0) port unreachable rcv from 10.120.1.15
ICMP: src 172.17.0.5, dst 172.16.1.1, echo reply
```

Debug icmp field descriptions: http://www.cisco.com/en/US/docs/ios/12_0/debug/command/reference/dipdrp.html#wp3795

# Four Layers for Troubleshooting: Physical and Routing Layer (Cont.)

- **Debugs and show commands use if no connectivity (cont.)**

  **debug ip packet** [*access-list-number*] [detail] [dump]

  Useful tool use for troubleshooting end to end communication

  IP packet debugging captures the packets that are process switched including received, generated and forwarded packets

```
IP: s=172.16.1.1 (local), d=172.17.0.1 (FastEthernet0/1), len 100, sending
      ICMP type=8, code=0
IP: table id=0, s=172.17.0.1 (FastEthernet0/1), d=172.16.1.1 (FastEthernet0/1), routed via RIB
IP: s=172.17.0.1 (FastEthernet0/1), d=172.16.1.1 (FastEthernet0/1), len 100, rcvd 3
      ICMP type=0, code=0
```

Caution: Debug IP packet command can generate a substantial amount of output and uses a substantial amount of system resources. This command should be used with caution in production networks. Always use with an ACL.

# Four Layers for Troubleshooting: Physical and Routing Layer

**Common Issues:**

- ACL in firewall/ISP side block ISAKMP traffic
- Traffic filtering resulting traffic flows one direction

# Common Issues: ACL in Firewall/ISP Side Block ISAKMP Traffic

**Problem:**

- Network connectivity between hub and spoke is fine
- IPsec tunnel is not coming up
- How to detect?

**show crypto isa sa**

IPv4 Crypto ISAKMP SA

| Dst | src | state | conn-id | slot | status |
|---|---|---|---|---|---|
| 172.17.0.1 | 172.16.1.1 | MM_NO_STATE | 0 | 0 | ACTIVE |
| 172.17.0.1 | 172.16.1.1 | MM_NO_STATE | 0 | 0 | ACTIVE (deleted) |
| 172.17.0.5 | 172.16.1.1 | MM_NO_STATE | 0 | 0 | ACTIVE |
| 172.17.0.5 | 172.16.1.1 | MM_NO_STATE | 0 | 0 | ACTIVE (deleted) |

**VPN tunnel flapping**

# Common Issues:
# ACL in Firewall/ISP Side Block ISAKMP Traffic

- Further check debug crypto isakmp to verify spoke router is sending udp 500 packet

**debug crypto isakmp**

```
04:14:44.450: ISAKMP:(0):Old State = IKE_READY  New State = IKE_I_MM1
04:14:44.450: ISAKMP:(0): beginning Main Mode exchange
04:14:44.450: ISAKMP:(0): sending packet to 172.17.0.1 my_port 500 peer_port 500 (I) MM_NO_STATE
04:14:44.450: ISAKMP:(0):Sending an IKE IPv4 Packet.
04:14:54.450: ISAKMP:(0): retransmitting phase 1 MM_NO_STATE...
04:14:54.450: ISAKMP (0:0): incrementing error counter on sa, attempt 1 of 5: retransmit phase 1
04:14:54.450: ISAKMP:(0): retransmitting phase 1 MM_NO_STATE
04:14:54.450: ISAKMP:(0): sending packet to 172.17.0.1 my_port 500 peer_port 500 (I) MM_NO_STATE
04:14:54.450: ISAKMP:(0):Sending an IKE IPv4 Packet.
04:15:04.450: ISAKMP:(0): retransmitting phase 1 MM_NO_STATE...
04:15:04.450: ISAKMP (0:0): incrementing error counter on sa, attempt 2 of 5: retransmit phase 1
04:15:04.450: ISAKMP:(0): retransmitting phase 1 MM_NO_STATE
04:15:04.450: ISAKMP:(0): sending packet to 172.17.0.1 my_port 500 peer_port 500 (I) MM_NO_STATE
04:15:04.450: ISAKMP:(0):Sending an IKE IPv4 Packet.
```

**Above debug output shows spoke router is sending udp 500 packet every 10 secs**

# Common Issues:
# ACL in Firewall/ISP Side Block ISAKMP Traffic

- How to fix?

  Check with either firewall admin OR ISP admin if spoke router is directly connected to ISP router to make sure they are allowing udp 500 traffic

  After ISP or Firewall admin allowed udp 500 add inbound ACL in egress interface which is tunnel source to allow udp 500 to make sure UDP 500 traffic coming into the router show access-list to verify hit counts are incrementing

**show access-lists 101**

```
Extended IP access list 101
   10 permit udp host 172.17.0.1 host 172.16.1.1 eq isakmp log (4 matches)
   20 permit udp host 172.17.0.5 host 172.16.1.1 eq isakmp log (4 matches)
   30 permit ip any any (295 matches)
```

Caution: Make sure you have 'ip any any' allowed in your access-list otherwise all other traffic will be blocked by this acl applied inbound on egress interface.

# Common Issues:
# ACL in Firewall/ISP Side Block ISAKMP Traffic

- How to verify?

**show crypto isa sa**

```
IPv4 Crypto ISAKMP SA
dst          src          state       conn-id  slot  status
172.17.0.1   172.16.1.1   QM_IDLE        1009    0   ACTIVE
172.17.0.5   172.16.1.1   QM_IDLE        1008    0   ACTIVE
```

**debug crypto isa**

```
ISAKMP:(0):Old State = IKE_READY  New State =IKE_I_MM1
ISAKMP:(0): beginning Main Mode exchange
ISAKMP:(0): sending packet to 172.17.0.1 my_port 500 peer_port 500 (I) MM_NO_STATE
ISAKMP (0:0): received packet from 172.17.0.1 dport 500 sport 500 Global (I) MM_NO_STATE
ISAKMP:(0):Sending an IKE IPv4 Packet Old State = IKE_R_MM1  New State = IKE_R_MM2
ISAKMP:(0):atts are acceptable
…
ISAKMP:(1009):Old State = IKE_R_MM3  New State IKE_R_MM3
…
ISAKMP:(1009):Old State = IKE_P1_COMPLETE  New State = IKE_P1_COMPLETE
```

**Phase 1 is UP, UDP 500 packet received**

# Common Issues:
# Traffic Filtering, Traffic Flows One Direction

## Problem

- VPN tunnel between spoke to spoke router is UP
- Unable to pass data traffic
- How to detect?

```
spoke1# show crypto ipsec sa peer 172.16.2.11
  local  ident (addr/mask/prot/port):    (172.16.1.1/255.255.255.255/47/0)
  remote ident (addr/mask/prot/port): (172.16.2.11/255.255.255.255/47/0)
   #pkts encaps: 110, #pkts encrypt: 110,  #pkts decaps: 0, #pkts decrypt: 0,
local crypto endpt.: 172.16.1.1,  remote crypto endpt.: 172.16.2.11
     inbound esp sas:  spi: 0x4C36F4AF(1278669999)
     outbound esp sas:  spi: 0x6AC801F4(1791492596)
```

```
spoke2#show crypto ipsec sa peer 172.16.1.1
  local  ident (addr/mask/prot/port): (172.16.2.11/255.255.255.255/47/0)
  remote ident (addr/mask/prot/port): (172.16.1.1/255.255.255.255/47/0)
  #pkts encaps: 116, #pkts encrypt: 116,  #pkts decaps: 110, #pkts decrypt: 110,
local crypto endpt.: 172.16.2.11,  remote crypto endpt.: 172.16.1.1
    inbound esp sas: spi: 0x6AC801F4(1791492596)
    outbound esp sas: spi: 0x4C36F4AF(1278669999)
```

There is no decap packets in Spoke 1, which means ESP packets are dropped some where in the path return from Spoke 2 towards Spoke1

# Common Issues:
# Traffic Filtering, Traffic Flows One Direction

- **How to fix?**

  Spoke 2 router shows both encap and decap which means either firewall in spoke 2 customer side ahead of router or ISP device in spoke 2 or any where in path between spoke 2 router and spoke 1 router filter ESP traffic

- **How to verify?**

```
spoke1# show crypto ipsec sa peer 172.16.2.11
  local  ident (addr/mask/prot/port):    (172.16.1.1/255.255.255.255/47/0)
  remote ident (addr/mask/prot/port): (172.16.2.11/255.255.255.255/47/0)
   #pkts encaps: 300, #pkts encrypt: 300
   #pkts decaps: 200, #pkts decrypt: 200,
```

```
spoke2#sh cry ipsec sa peer 172.16.1.1
  local  ident (addr/mask/prot/port): (172.16.2.11/255.255.255.255/47/0)
  remote ident (addr/mask/prot/port): (172.16.1.1/255.255.255.255/47/0)
  #pkts encaps: 316, #pkts encrypt: 316,
  #pkts decaps: 300, #pkts decrypt: 310,
```

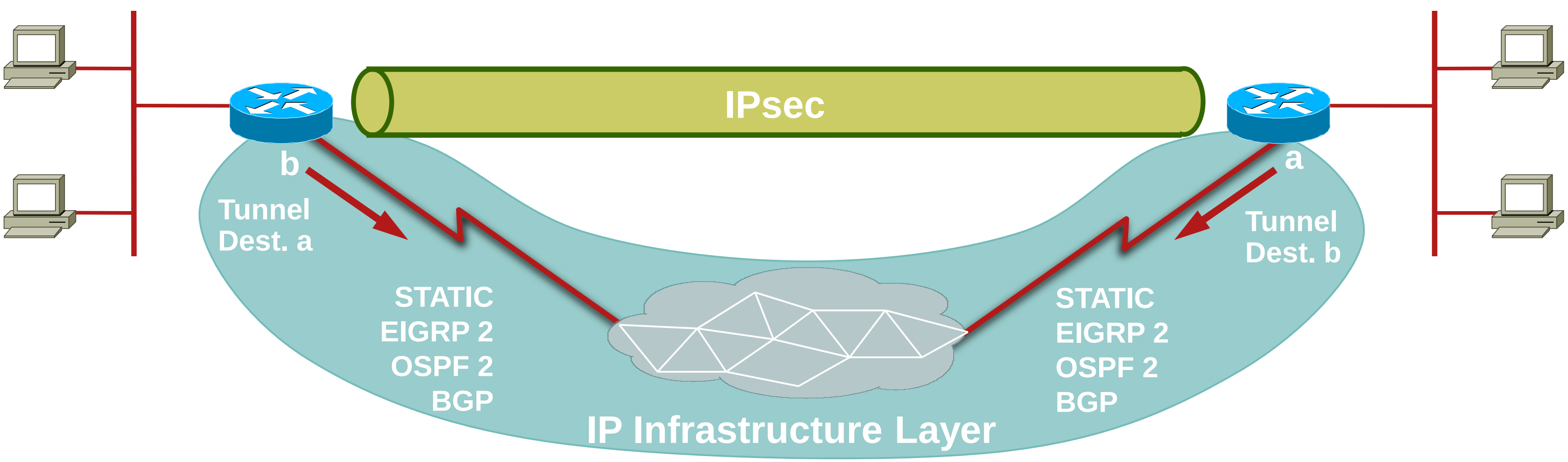
After allowed ESP (IP protocol 50) Spoke 1 and Spoke 2 both shows encaps and decaps, counters are incrementing.

# Four Layers for Troubleshooting: IPsec Encryption Layer

- The IPsec encryption layer—

  This is encrypting the GRE tunnel packet going out and decrypting the IPsec packet coming in to reveal the GRE encapsulated packet



# Four Layers for Troubleshooting: IPsec Encryption Layer—IPsec Component

**DMVPN Component-IPsec**

- DMVPN introduced tunnel protection
- The profile must be applied on the tunnel interface

  tunnel protection ipsec profile prof

- Internally Cisco IOS Software will treat this as a dynamic crypto map and it derives the local-address, set peer and match address parameters from the tunnel parameters and the NHRP cache
- This must be configured on the hub and spoke tunnels

# Four Layers for Troubleshooting: IPsec Encryption Layer—IPsec Component

## DMVPN Component-IPsec (Cont.)

- A transform set must be defined:

```
crypto ipsec transform-set ts esp-3des esp-sha-hmac
 mode transport
```

- An IPsec profile replaces the crypto map

```
crypto ipsec profile prof
 set transform-set ts
```

- The IPsec profile is like a crypto map without "set peer" and "match address"

```
Interface Tunnel0
 Ip address 10.0.0.1 255.255.255.0
 :
 tunnel source fast ethernet0/0

 tunnel protection ipsec profile prof
```

**Note: GRE Tunnel Keepalives are not supported in combination with Tunnel Protection**

# Four Layers for Troubleshooting: IPsec Encryption Layer

## IPsec Layer Verification-show commands

- Verify that ISAKMP SAs and IPsec SAs between the NBMA addresses of the hub and spoke have being created

  show crypto isakmp sa detail

  show crypto IPsec sa peer <NBMA-address-peer

- Notice SA lifetime values

  If they are close to the configured lifetimes (default --24 hrs for ISAKMP and 1 hour for IPsec) then that means these SAs have been recently negotiated

  If you look a little while later and they have been re-negotiated again, then the ISAKMP and/or IPsec may be bouncing up and down

# Four Layers for Troubleshooting: IPsec Encryption Layer

## IPsec Layer Verification-show commands (Cont.)

- New show commands for dmvpn introduced in 12.4(9)T that has brief and detail output

**show dmvpn detail**

Covers both Isakmp phase 1 and IPsec phase 2 status

Does not show remaining life time for both Isakmp phase1 and IPsec phase 2 ,to check life time still use old commands

```
Show dmvpn [ {interface <i/f>} |
             {vrf <vrf-name>} |
             {peer  {{nbma | tunnel } <ip-addr> } |
                    {network <ip-addr> <mask>}} ]
             [detail]
```

# Four Layers for Troubleshooting: IPsec Encryption Layer

## IPsec Layer Verification-debug commands

- Check the debug output on both the spoke and the hub at the same time

    debug crypto isakmp

    debug crypto ipsec

    debug crypto engine

    New command → debug dmvpn detail crypto (Introduced in 12.4(9)T)

- Use conditional debugging on the hub router to restrict the crypto debugs to only show debugs for the particular spoke in question:

    debug crypto condition peer ipv4 <nbma address>

    debug dmvpn condition peer <nbma|tunnel>

- Verify the communication between NHRP and IPsec by showing the crypto  map and socket tables

    show crypto map

    show crypto socket

# Four Layers for Troubleshooting: IPsec Encryption Layer—Show Commands

## show crypto isakmp sa

```
Router# show crypto isakmp sa
dst            src            state              connid   slot
172.17.0.1   172.16.1.1     QM_IDLE                1        0
```

IKE Phase 1 status UP

## show crypto isakmp sa detail

```
Router# show crypto isakmp sa detail
Codes: C - IKE configuration mode,
       D - Dead Peer Detection
       K - Keepalives, N - NAT-traversal
       X - IKE Extended Authentication
        psk - Preshared key, rsig - RSA signature,    renc - RSA encryption

C-id  Local          Remote         I-VRF Encr Hash Auth DH Lifetime Cap.
1     172.16.1.1     172.17.0.1           3des sha  psk  1  23:59:40
       Connection-id:Engine-id =  1:1(hardware)
```

Encryption:3des
Authenticatio :Pre-shared key
Remaining lifetime before phase 1 re-key

# Four Layers for Troubleshooting: IPsec Encryption Layer—Show Commands

**show crypto ipsec sa**

```
Router# show crypto ipsec sa
interface: Ethernet0/3
   Crypto map tag: vpn, local addr. 172.17.0.1
  local  ident (addr/mask/prot/port): (172.16.1.1/255.255.255.255/47/0)
  remote ident (addr/mask/prot/port): (172.17.0.1/255.255.255.255/47/0)
  current_peer: 172.17.0.1:500
    PERMIT, flags={origin_is_acl,}
  #pkts encaps: 19, #pkts encrypt: 19, #pkts digest 19
  #pkts decaps: 19, #pkts decrypt: 19, #pkts verify 19
  #pkts compressed: 0, #pkts decompressed: 0
  #pkts not compr'ed: 0, #pkts compr. failed: 0, #pkts decompr. failed: 0
  #send errors 1, #recv errors 0
   local crypto endpt.: 172.16.1.1, remote crypto endpt.: 172.17.0.1
   path mtu 1500, media mtu 1500
   current outbound spi: 8E1CB77A
```

# Four Layers for Troubleshooting: IPsec Encryption Layer—Show Commands

## show crypto ipsec sa (cont.)

```
 inbound esp sas:
      spi: 0x4579753B(1165587771)
        transform: esp-3des esp-md5-hmac ,
        in use settings ={Tunnel, }
        slot: 0, conn id: 2000, flow_id: 1, crypto map: vpn
        sa timing: remaining key lifetime (k/sec): (4456885/3531)
        IV size: 8 bytes
        replay detection support: Y
outbound esp sas:
      spi: 0x8E1CB77A(2384246650)
        transform: esp-3des esp-md5-hmac ,
        in use settings ={Tunnel, }
        slot: 0, conn id: 2001, flow_id: 2, crypto map: vpn
        sa timing: remaining key lifetime (k/sec): (4456885/3531)
        IV size: 8 bytes
        replay detection support: Y
```

Remaining life time before re-key

# Four Layers for Troubleshooting: IPsec Encryption Layer—Show Commands

**show dmvpn**

```
HUB-1#show dmvpn

Legend: Attrb --> S - Static, D - Dynamic, I - Incomplete
        N - NATed, L - Local, X - No Socket
        # Ent --> Number of NHRP entries with same NBMA peer

Tunnel1, Type:Hub, NHRP Peers:2,
 # Ent  Peer NBMA Addr Peer Tunnel Add State  UpDn Tm Attrb
 ----- --------------- --------------- ----- -------- -----
     1         1.1.1.1      172.20.1.1    UP 00:04:32 D
     1         2.2.2.2      172.20.1.2    UP 00:01:25 D

SPOKE-1#show dmvpn

Legend: Attrb --> S - Static, D - Dynamic, I - Incompletea
        N - NATed, L - Local, X - No Socket
        # Ent --> Number of NHRP entries with same NBMA peer

Tunnel1, Type:Spoke, NHRP Peers:1,
 # Ent  Peer NBMA Addr Peer Tunnel Add State  UpDn Tm Attrb
 ----- --------------- --------------- ----- -------- -----
     1         3.3.3.3    172.20.1.100    UP 00:21:56 S
```

**Learn Dynamically, Entry shows either in hub or in spoke for spoke to spoke tunnels**

**Static NHRP mapping**

# Four Layers for Troubleshooting: IPsec Encryption Layer—Show Commands

## show dmvpn detail

```
HUB-1#show dmvpn detail

Legend: Attrb --> S - Static, D - Dynamic, I - Incompletea
        N - NATed, L - Local, X - No Socket
        # Ent --> Number of NHRP entries with same NBMA peer

 -------------- Interface Tunnel1 info: --------------
Intf. is up, Line Protocol is up, Addr. is 172.20.1.100
   Source addr: 3.3.3.3, Dest addr: MGRE
  Protocol/Transport: "multi-GRE/IP", Protect "gre_prof",
Tunnel VRF "", ip vrf forwarding ""

NHRP Details:
Type:Hub, NBMA Peers:2
# Ent  Peer NBMA Addr Peer Tunnel Add State  UpDn Tm Attrb    Target Network
----- --------------- --------------- ----- -------- ----- -----------------
    1         1.1.1.1      172.20.1.1    UP 00:26:38 D         172.20.1.1/32

  IKE SA: local 3.3.3.3/500 remote 1.1.1.1/500 Active
  Crypto Session Status: UP-ACTIVE
  fvrf: (none)
  IPSEC FLOW: permit 47 host 3.3.3.3 host 1.1.1.1
        Active SAs: 2, origin: crypto map
   Outbound SPI : 0xB28957C6, transform : esp-3des esp-sha-hmac
    Socket State: Open
```

Only One Peer Shown

# Four Layers for Troubleshooting: IPsec Encryption Layer—debug crypto condition

- The crypto conditional debug CLIs **(debug crypto condition, debug crypto condition unmatched, and show crypto debug-condition)** allow you to specify conditions (filter values) in which to generate and display debug messages related only to the specified conditions
- The router will perform conditional debugging only after at least one of the global crypto debug commands **(debug crypto isakmp, debug crypto ipsec,** or **debug crypto engine)** has been enabled. This requirement helps to ensure that the performance of the router will not be impacted when conditional debugging is not being used.

# Four Layers for Troubleshooting: IPsec Encryption Layer—debug crypto Condition

- To enable crypto conditional debugging:

  **debug crypto condition <cond-type> <cond-value>**
  **debug crypto { isakmp | ipsec | engine }**

- To view crypto condition debugs that have been enabled:

  **show crypto debug-condition [ all | peer | fvrf | ivrf | isakmp | username | connid | spi ]**

- To disable crypto condition debugs:

  **debug crypto condition reset**

# Four Layers for Troubleshooting: IPsec Encryption Layer—debug crypto Condition

| | |
|---|---|
| Fvrf | The name string of a virtual private network (VPN) routing and forwarding (VRF) instance. Relevant debug messages will be shown if the current IPsec operation uses this VRF instance as its front-door VRF (FVRF). |
| ivrf | The name string of a VRF instance. Relevant debug messages will be shown if the current IPsec operation uses this VRF instance as its inside VRF (IVRF). |
| isakmp profile | The name string of the isakmp profile to be matched against for debugging. |
| Local ipv4 | The ip address string of the local IKE endpoint. |
| Peer group | A ezvpn group name string. Relevant debug messages will be shown if the peer is using this group name as its identity. |
| Peer ipv4 | A single IP address. Relevant debug messages will be shown if the current IPsec operation is related to the IP address of this peer. |
| Peer subnet | A subnet and a subnet mask that specify a range of peer IP addresses. Relevant debug messages will be shown if the IP address of the current IPsec peer falls into the specified subnet range. |
| Peer hostname | A fully qualified domain name (FQDN) string. Relevant debug messages will be shown if the peer is using this string as its identity. |
| username | The username string (XAuth username or PKI-aaa username obtained from a certificate). |

# Four Layers for Troubleshooting: IPsec Encryption Layer—debug dmvpn detail all

**debug tunnel protection** → **debug crypto socket** → **debug crypto isakmp** → **debug crypto IPsec** → **debug tunnel protection** → **debug nhrp packet**

- debug dmvpn introduced in 12.4(9)T

  **debug dmvpn** {[{**condition** [**unmatched**] |
  [**peer** [**nbma** | **tunnel** {*ip-address*}]] |
  [**vrf** {*vrf-name*}] |
  [**interface** {**tunnel** *number*}]}] |
  [{**error** | **detail** | **packet** | **all**}
  {**nhrp** | **crypto** | **tunnel** | **socket** | **all**}]}

- One complete debug to help troubleshoot dmvpn issues

# Four Layers for Troubleshooting: IPsec Encryption Layer—debug dmvpn detail all (Cont.)

**debug tunnel protection** → **debug crypto socket** → **debug crypto isakmp** → **debug crypto IPsec** → **debug tunnel protection** → **debug nhrp packet**

- Tunnel protection configured on tunnel interface open crypto socket as soon as either router or tunnel came up

IPSEC-IFC MGRE/Tu0: **Checking tunnel status**

IPSEC-IFC MGRE/Tu0(172.16.2.11/172.17.0.1): Opening a socket with **profile dmvpn**

IPSEC-IFC MGRE/Tu0(172.16.2.11/172.17.0.1): connection lookup returned 0

IPSEC-IFC MGRE/Tu0(172.16.2.11/172.17.0.1): Triggering tunnel immediately.

IPSEC-IFC MGRE/Tu0: **tunnel coming up**

IPSEC-IFC MGRE/Tu0(172.16.2.11/172.17.0.1): Opening a socket with profile dmvpn

IPSEC-IFC MGRE/Tu0(172.16.2.11/172.17.0.1): connection lookup returned **83884274**

IPSEC-IFC MGRE/Tu0(172.16.2.11/172.17.0.1): Socket is already being opened. Ignoring.

# Four Layers for Troubleshooting: IPsec Encryption Layer—debug dmvpn detail all (Cont.)

**debug tunnel protection** → **debug crypto socket** → **debug crypto isakmp** → **debug crypto IPsec** → **debug tunnel protection** → **debug nhrp packet**

- Shows socket state
- Crypto socket debug shows creation of local and remote proxy id

**CRYPTO_SS** (TUNNEL SEC): **Application started listening**

insert of map into mapdb AVL failed, map + ace pair already exists on the mapdb

CRYPTO-6-ISAKMP_ON_OFF: ISAKMP is ON

CRYPTO_SS(TUNNEL SEC): **Active open**, socket info:

local 172.16.2.11 172.16.2.11/255.255.255.255/0,

remote 172.17.0.1 172.17.0.1/255.255.255.255/0, prot 47, ifc Tu0

# Four Layers for Troubleshooting: IPsec Encryption Layer—debug dmvpn detail all (Cont.)

debug tunnel protection → debug crypto socket → **debug crypto isakmp** → debug crypto IPsec → debug tunnel protection → debug nhrp packet

- IKE negotiation
- Shows six packet exchange(MM1-MM6) in main mode

```
ISAKMP:(0):Old State = IKE_READY  New State = IKE_I_MM1
ISAKMP:(0): beginning Main Mode exchange
ISAKMP:(0): sending packet to 172.17.0.1 my_port 500 peer_port 500 (I) MM_NO_STATE
ISAKMP:(0):Sending an IKE IPv4 Packet
ISAKMP:(0):Old State = IKE_I_MM1  New State = IKE_I_MM2
ISAKMP:(0):Checking ISAKMP transform 1 against priority 10 policy
ISAKMP:(0):atts are acceptable. Next payload is 0
ISAKMP:(0):Old State = IKE_I_MM2  New State = IKE_I_MM3
ISAKMP:(0):Old State = IKE_I_MM3  New State = IKE_I_MM4
ISAKMP:(1051):Old State = IKE_I_MM4  New State = IKE_I_MM5
ISAKMP:(1051):Old State = IKE_I_MM5  New State = IKE_I_MM6
ISAKMP:(1051):Old State = IKE_I_MM6  New State = IKE_P1_COMPLETE
```

IKE has found matching policy

IKE complete authentication

# Four Layers for Troubleshooting: IPsec Encryption Layer—debug dmvpn detail all (Cont.)

debug tunnel protection → debug crypto socket → debug crypto isakmp → **debug crypto IPsec** → debug tunnel protection → debug nhrp packet

- IKE negotiates to set up the IP Security (IPsec) SA by searching for a matching transform set
- Creation of inbound and outbound security association database (SADB)

```
ISAKMP:(1051):beginning Quick Mode exchange, M-ID of 1538742728
ISAKMP:(1051):Old State = IKE_QM_READY  New State = IKE_QM_I_QM1
ISAKMP:(1051):atts are acceptable.
INBOUND local= 172.16.2.11, remote= 172.17.0.5,
local_proxy= 172.16.2.11/255.255.255.255/47/0 (type=1),
remote_proxy= 172.17.0.5/255.255.255.255/47/0 (type=1),
protocol= ESP, transform= esp-3des esp-sha-hmac  (Transport),
ISAKMP:(1051): Creating IPsec SAs
inbound SA from 172.17.0.5 to 172.16.2.11 (f/i)  0/ 0
 (proxy 172.17.0.5 to 172.16.2.11)
has spi 0xE563BB42 and conn_id 0
outbound SA from 172.16.2.11 to 172.17.0.5 (f/i) 0/0
(proxy 172.16.2.11 to 172.17.0.5)
has spi  0xFE745CBD and conn_id 0
ISAKMP:(1051):Old State = IKE_QM_I_QM1  New State = IKE_QM_PHASE2_COMPLETE
```

**Phase 2 Complete**

# Four Layers for Troubleshooting: IPsec Encryption Layer

**Common Issues:**

- Incompatible ISAKMP Policy
- Incorrect Pre-shared key secret
- Incompatible IPsec transform set

# Common Issues: Incompatible ISAKMP Policy

- If the configured ISAKMP policies don't match the proposed policy by the remote peer, the router tries the default policy of 65535, and if that does not match either, it fails ISAKMP negotiation

```
Default protection suite
encryption algorithm:   DES—Data Encryption Standard (56 bit keys).
hash algorithm:         Secure Hash Standard
authentication method:  Rivest-Shamir-Adleman Signature
Diffie-Hellman group:   #1 (768 bit)
lifetime:               86400 seconds, no volume limit
```

- A show crypto isakmp sa shows the ISAKMP SA to be in MM_NO_STATE, meaning that main-mode failed

# Common Issues: Incompatible ISAKMP Policy (Cont.)

Msg 1 and 2 of ISAKMP MM

```
ISAKMP (0:1): processing SA payload.
message ID = 0

ISAKMP (0:1): found peer pre-shared
key matching 209.165.200.227

ISAKMP (0:1): Checking ISAKMP
transform 1 against priority 1 policy

ISAKMP:       encryption 3DES-CBC

ISAKMP:       hash MD5

ISAKMP:       default group 1

ISAKMP:       auth pre-share

ISAKMP:       life type in seconds

ISAKMP:       life duration (VPI) of
0x0 0x1 0x51 0x80

ISAKMP (0:1): Hash algorithm offered
does not match policy!

ISAKMP (0:1): atts are not acceptable.
Next payload is 0
```

```
ISAKMP (0:1): Checking ISAKMP
transform 1 against priority 65535
policy

ISAKMP:       encryption 3DES-CBC

ISAKMP:       hash MD5

ISAKMP:       default group 1

ISAKMP:       auth pre-share

ISAKMP:       life type in seconds

ISAKMP:       life duration (VPI) of
0x0 0x1 0x51 0x80

ISAKMP (0:1): Encryption algorithm
offered does not match policy!

ISAKMP (0:1): atts are not acceptable.
Next payload is 0

ISAKMP (0:1): no offers accepted!

ISAKMP (0:1): phase 1 SA not
acceptable!
```

# Common Issues: Incorrect Pre-Shared Secrets

- If the pre-shared secrets are not the same on both sides, the negotiation will fail again, with the router complaining about **sanity check failed**
- A show crypto isakmp sa shows the ISAKMP SA to be in MM_NO_STATE, meaning the main mode failed

```
ISAKMP (62): processing SA payload. message ID = 0
ISAKMP (62): Checking ISAKMP transform 1 against priority 10 policy
                    encryption DES-CBC
                    hash SHA
                    default group 1
                    auth pre-share
ISAKMP (62): atts are acceptable. Next payload is 0
ISAKMP (62): SA is doing pre-shared key authentication
ISAKMP (62): processing KE payload. message ID = 0
ISAKMP (62): processing NONCE payload. message ID = 0
ISAKMP (62): SKEYID state generated
ISAKMP (62); processing vendor id payload
ISAKMP (62): speaking to another Cisco IOS box!
ISAKMP: reserved not zero on ID payload!
%CRYPTO-4-IKMP_BAD_MESSAGE: IKE message from 209.165.200.227
 failed its sanity check or is malformed
```

Msg 5 and 6 of ISAKMP MM

# Common Issues: Incompatible IPsec Transform Set

- If the ipsec transform-set is not compatible or mismatched on the two IPsec devices, the IPsec negotiation will fail, with the router complaining about **"atts not acceptable"** for the IPsec proposal

**Phase II Parameters**

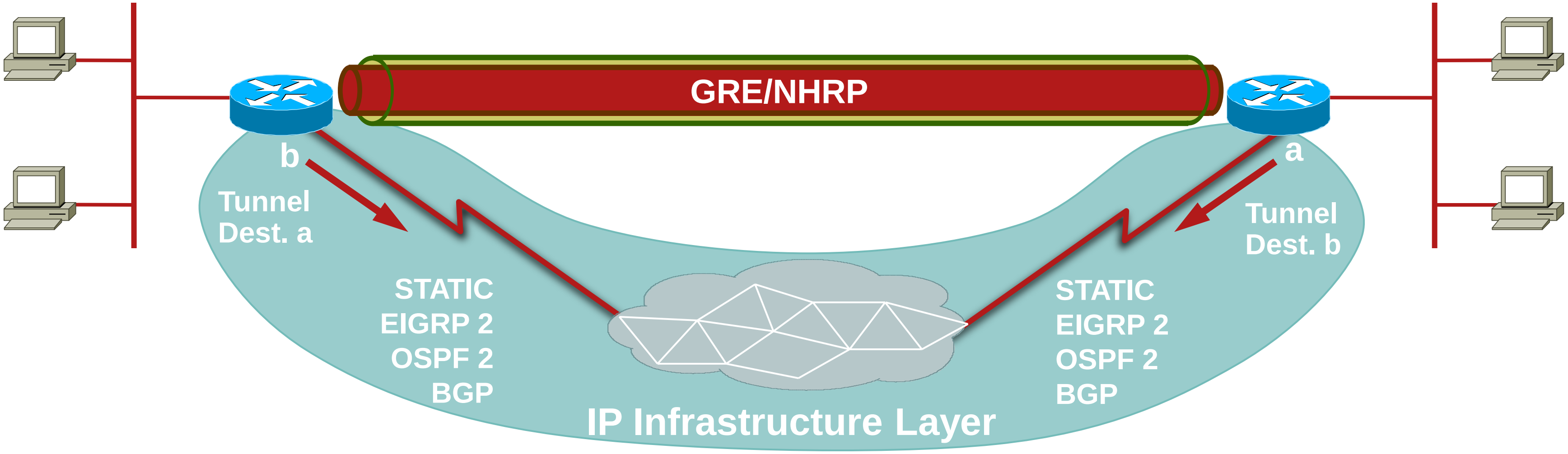
```
IPsec mode (tunnel or transport)
Encryption algorithm
Authentication algorithm
PFS group
IPsec SA Lifetime
Proxy identities
```

```
ISAKMP (0:2): Checking IPsec proposal 1
ISAKMP: transform 1, ESP_3DES
ISAKMP:   attributes in transform:
ISAKMP:      encaps is 1
ISAKMP:      SA life type in seconds
ISAKMP:      SA life duration (basic) of 3600
ISAKMP:      SA life type in kilobytes
ISAKMP:      SA life duration (VPI) of  0x0 0x46 0x50 0x0
IPSEC(validate_proposal): transform proposal (prot 3, trans 3, hmac_alg 0) not supported
ISAKMP (0:2): atts not acceptable. Next payload is 0
ISAKMP (0:2): SA not acceptable!
```

# Four Layers for Troubleshooting: GRE Encapsulation Layer

- The GRE Encapsulation layer—NHRP

  This is GRE encapsulating the data IP packet going out and GRE decapsulating the GRE packet (after IPsec encryption) coming in to get the data IP packet



# Four Layers for Troubleshooting: GRE Encapsulation Layer

**DMVPN Component-GRE/NHRP**

- Multipoint GRE Tunnel Interface

    Single GRE interface to support multiple GRE/IPsec tunnels

    Simplifies size and complexity of configuration

- Next Hop Resolution Protocol (NHRP)

    Creates a distributed (NHRP) mapping database of all the spoke's tunnel to real (public interface) addresses

# Four Layers for Troubleshooting: GRE Encapsulation Layer

## DMVPN Component-mGRE

- A p-pGRE interface definition includes
  - An IP address
  - A tunnel source
  - A tunnel destination
  - An optional tunnel key

```
interface Tunnel
   ip address 10.0.0.1 255.0.0.0
   tunnel source Dialer1
   tunnel destination 172.16.0.2
   tunnel key 1
```

- An mGRE interface definition includes
  - An IP address
  - A tunnel source
  - An option tunnel key

```
interface Tunnel
   ip address 10.0.0.1 255.0.0.0
   tunnel source Dialer1
   tunnel mode gre multipoint
   tunnel key 1
```

# Four Layers for Troubleshooting: GRE Encapsulation Layer

## DMVPN Component-mGRE (Cont.)

- Single tunnel interface (multipoint)
  - Non-Broadcast Multi-Access (NBMA) Network
  - Smaller hub configuration
  - Multicast/broadcast support
- Dynamic tunnel destination
  - Next Hop Resolution Protocol (NHRP)
  - VPN IP to NBMA IP address mapping
  - Short-cut forwarding
  - Direct support for dynamic addresses and NAT

# Four Layers for Troubleshooting: GRE Encapsulation Layer—What Is NHRP

**DMVPN Component-NHRP**

- NHRP is a layer two resolution protocol and cache like ARP or Reverse ARP (Frame Relay)
- It is used in DMVPN to map a tunnel IP address to an NBMA address
- Like ARP, NHRP can have static and dynamic entries
- NHRP has worked fully dynamically since Release 12.2(13)T

# Four Layers for Troubleshooting:GRE Encapsulation Layer—Basic NHRP Configuration

**DMVPN Component-NHRP (Cont.)**

- In order to configure an mGRE interface to use NHRP, the following command is necessary:

  **ip nhrp network-id <id>**

- Where <id> is a unique number (recommend same on hub and all spokes)
- <id> has nothing to do with tunnel key
- The network ID defines an NHRP domain
- Several domains can co-exist on the same router
- Without having this command, tunnel interface won’t come UP

# Four Layers for Troubleshooting: GRE Encapsulation Layer—Adding NHRP Cache

## DMVPN Component-NHRP (Cont.)

- Three ways to populate the NHRP cache:
  - Manually add static entries
  - Hub learns via registration requests
  - Spokes learn via resolution requests
- "Resolution" is for spoke to spoke

# Four Layers for Troubleshooting: GRE Encapsulation Layer—Initial NHRP Caches

**DMVPN Component-NHRP (Cont.)**

- Initially, the hub has an empty cache
- The spoke has one static entry mapping the hub's tunnel address to the hub's NBMA address:

  **ip nhrp map 10.0.0.1 172.17.0.1**

- Multicast traffic must be sent to the hub

  **ip nhrp map multicast 172.17.0.1**

# Four Layers for Troubleshooting: GRE Encapsulation Layer—Spoke Must Register with Hub

**DMVPN Component-NHRP (Cont.)**

- In order for the spokes to register themselves to the hub, the hub must be declared as a Next Hop Server (NHS):

  **ip nhrp nhs 10.0.0.1**

  **ip nhrp holdtime 300 (recommended; default =7200)**

  **ip nhrp registration no-unique (recommended*)**

- Spokes control the cache on the hub

# Four Layers for Troubleshooting: GRE Encapsulation Layer—NHRP Registration

**DMVPN Component-NHRP (Cont.)**

- NHRP Registration
  - Spoke dynamically registers its mapping with NHS
  - Supports spokes with dynamic NBMA addresses or NAT
- NHRP Resolutions and Redirects
  - Supports building dynamic spoke-spoke tunnels
  - Control and Multicast traffic still via hub
  - Unicast data traffic direct, reduced load on hub routers

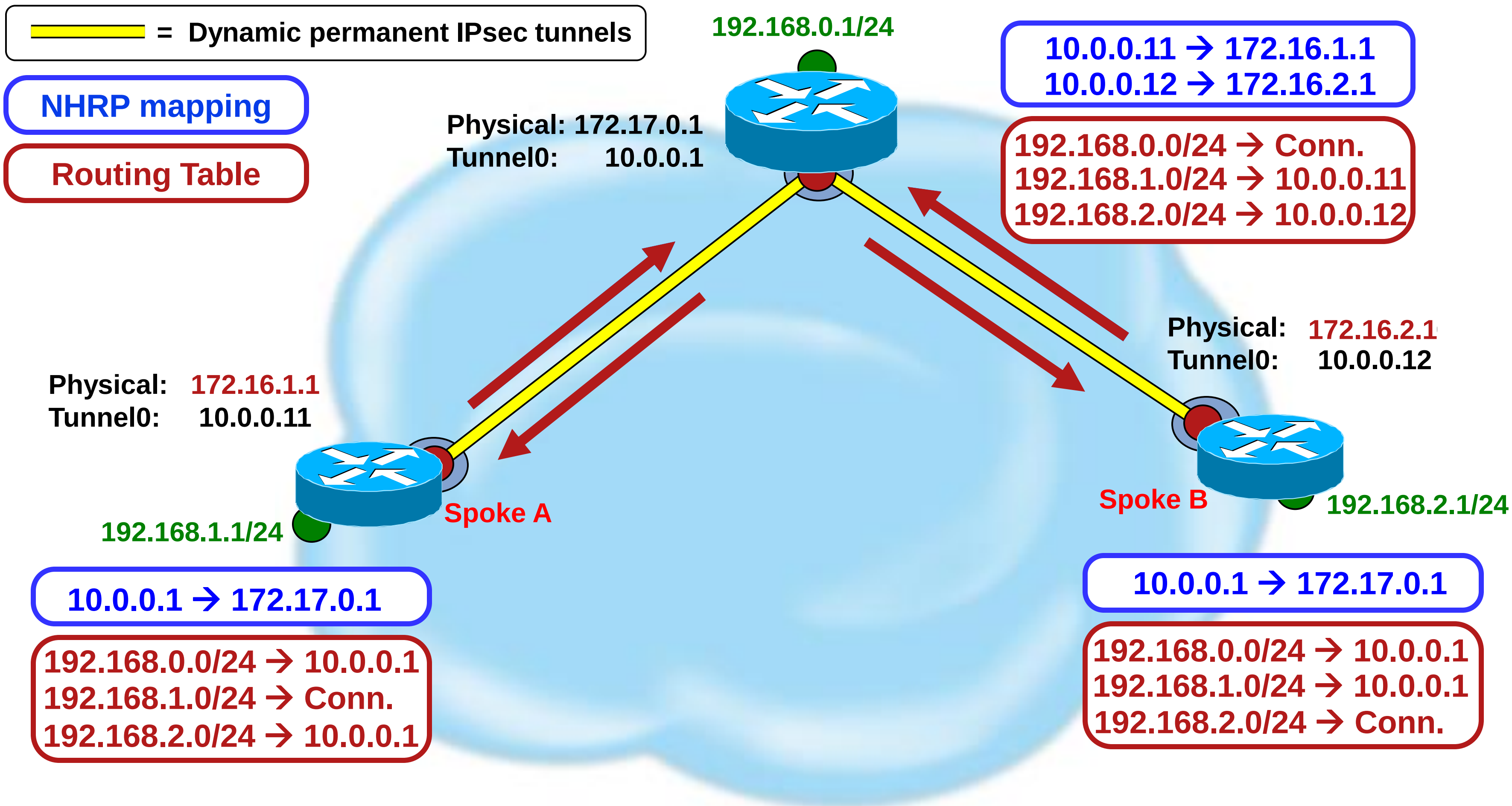
# Four Layers for Troubleshooting: GRE Encapsulation Layer—NHRP Registration (Cont.)

**DMVPN Component-NHRP (Cont.)**

- Builds base hub-and-spoke network
  - Hub-and-spoke data traffic
  - Control traffic; NHRP, Routing protocol, IP multicast
- Next Hop Client (NHC) has static mapping for Next Hop Servers (NHSs)
- Registration time is configurable
  - ip nhrp registration timer <value> (default = 1/3 nhrp hold time)
- NHS registration reply gives liveliness of NHS
  - Important for Phase 2 networks

# NHRP Registration Example
# Dynamically Addressed Spokes



# Four Layers for Troubleshooting: GRE Encapsulation Layer

- Look at NHRP. The spoke should be sending an NHRP registration packet on a regular basis, every 1/3 NHRP hold time (on spoke) or 'ip nhrp registration timeout <seconds>' value.

  On the Spoke: **show ip nhrp nhs detail**

  On the hub: **show ip nhrp <spoke-tunnel-ip-address>**

- Check the 'created' and 'expire' timer :

  **'created' timer:** how long this NHRP mapping entry has continuously been in the NHRP mapping table.

  **'expire' timer:** how long before this NHRP mapping entry would be deleted, if the hub were not to receive another NHRP registration from the spoke.

  If the 'created' timer is low and gets reset a lot then that means that the NHRP mapping entry is getting reset

# Four Layers for Troubleshooting: GRE Encapsulation Layer

- Verify pings from the hub to the spoke's tunnel ip address and the reverse.
- Use the following debugs on the hub router.

  debug nhrp condition peer <nbma|tunnel>

  debug nhrp

  debug tunnel protection

  debug crypto socket

    (these last two show communication between NHRP and IPsec)

# Four Layers for Troubleshooting: GRE Encapsulation Layer—Show Commands

## show ip nhrp detail

```
10.0.0.5/32 via 10.0.0.5, Tunnel0 created 03:36:47, never expire
 Type: static, Flags: used
 NBMA address: 172.17.0.5
10.0.0.9/32 via 10.0.0.9, Tunnel0 created 03:26:26, expire 00:04:04
  Type: dynamic, Flags: unique nat registered
  NBMA address: 110.110.110.2
10.0.0.11/32 via 10.0.0.11, Tunnel0 created 01:55:43, expire 00:04:15
  Type: dynamic, Flags: unique nat registered
  NBMA address: 120.120.120.2
```

## show ip nhrp nhs detail

```
Legend: E=Expecting replies, R=Responding

Tunnel0: 10.0.0.1 RE  req-sent 654 req-failed 0 repl-recv 590 (00:00:09 ago)
         10.0.0.5 RE  req-sent 632 req-failed 0 repl-recv 604 (00:00:09 ago)
```

**NHRP Flag Information**:
**http://www.cisco.com/en/US/docs/ios/12_4/ip_addr/configuration/guide/hadnhrp_ps6350_TSD_Products_Configuration_Guide_Chapter.html#wp1067931**

# Four Layers for Troubleshooting:GRE Encapsulation Layer—debug dmvpn detail all

debug tunnel protection → debug crypto socket → debug crypto isakmp → debug crypto IPsec → **debug tunnel protection** → debug nhrp packet

- Tunnel protection start again after Phase 2 came UP
- Connection lookup id should be same used when tunnel start
- Syslog message shows socket came UP
- Signal NHRP after socket UP

**ID value has to be same when socket open in the beginning**

```
IPSEC-IFC MGRE/Tu0(172.16.2.11/172.17.0.1): connection lookup returned 83884274
IPSEC-IFC MGRE/Tu0(172.16.2.11/172.17.0.5): tunnel_protection_socket_up
IPSEC-IFC MGRE/Tu0(172.16.2.11/172.17.0.5): Signalling NHRP
IPSEC-IFC MGRE/Tu0(172.16.2.11/172.17.0.5): connection lookup returned 83DD7B30
IPSEC-IFC MGRE/Tu0(172.16.2.11/172.17.0.1): connection lookup returned 83884274
IPSEC-IFC MGRE/Tu0(172.16.2.11/172.17.0.1): tunnel_protection_socket_up
IPSEC-IFC MGRE/Tu0(172.16.2.11/172.17.0.1): Signalling NHRP
```

**Syslog message:**

**%DMVPN-7-CRYPTO_SS: Tunnel0-172.16.2.11 socket is UP**

# Four Layers for Troubleshooting: GRE Encapsulation Layer-debug dmvpn detail all (Cont.)

**debug tunnel protection** → **debug crypto socket** → **debug crypto isakmp** → **debug crypto IPsec** → **debug tunnel protection** → **debug nhrp packet**

- Spoke send NHRP registration request.
- Req id has to be same in both registration request and response.

| NHRP: Send Registration Request | NHRP: Receive Registration Reply |
|---|---|
| **NHRP: Send Registration Request** via Tunnel0 vrf 0, packet size: 104 | **NHRP: Receive Registration Reply** via Tunnel0 vrf 0, packet size: 124 |
| src: 10.0.0.9, dst: 10.0.0.1 | (F) afn: IPv4(1), type: IP(800), hop: 255, ver: 1 |
| (F) afn: IPv4(1), type: IP(800), hop: 255, ver: 1 | shtl: 4(NSAP), sstl: 0(NSAP) |
| shtl: 4(NSAP), sstl: 0(NSAP) | (M) flags: "unique nat ", reqid: **1279** |
| (M) flags: "unique nat ", reqid: **1279** | src NBMA: 172.16.1.1. |
| src NBMA: 172.16.1.1 | src protocol: 10.0.0.9, dst protocol: 10.0.0.1 |
| src protocol: 10.0.0.9, dst protocol: 10.0.0.1 | (C-1) code: no error(0) |
| (C-1) code: no error(0) | prefix: 255, mtu: 1514, hd_time: 300 |
| prefix: 255, mtu: 1514, hd_time: 300 | addr_len: 0(NSAP), subaddr_len: 0(NSAP), proto_len: 0, pref: 0 |
| addr_len: 0(NSAP), subaddr_len: 0(NSAP), proto_len: 0, pref: 0 | |

**Syslog message**:

%**DMVPN-5-NHRP_NHS:** Tunnel0 10.0.0.1 is UP

# DMVPN Data Structures Interaction

**Hub1#show ip nhrp**

10.0.0.2/32 via 10.0.0.2, …
  NBMA address: 172.17.0.5
10.0.0.11/32 via 10.0.0.11, …
  NBMA address: 172.16.1.1
10.0.0.12/32 via 10.0.0.12, …
  NBMA address: 172.16.2.1
   (no-socket)

**Hub1# show crypto socket**

Tu0 Peers (local/remote): 172.17.0.1/172.17.0.5
  Local Ident (ad/ma/po/pr): (172.17.0.1/255.255.255.255/0/47)
  Remote Ident (ad/ma/po/pr): (172.17.0.5/255.255.255.255/0/47)
  Socket State: Open
Tu0 Peers (local/remote): 172.17.0.1/172.16.1.1
  Local Ident (ad/ma/po/pr): (172.17.0.1/255.255.255.255/0/47)
  Remote Ident (ad/ma/po/pr): (172.16.1.1/255.255.255.255/0/47)
  Socket State: Open

**Hub1#show crypto ipsec sa**

interface: Tunnel0
Crypto map tag: Tunnel0-head-0,
  local crypto endpt.: 172.17.0.1,
  remote crypto endpt.: 172.17.0.5
    inbound sas: spi: 0x3C32F075
    outbound sas: spi: 0x149FA5E7
  local crypto endpt.: 172.17.0.1,
  remote crypto endpt.: 172.16.1.1
    inbound sas: spi: 0x8FE87A1B
    outbound sas: spi: 0xD111D4E0

**Hub1#show crypto map**

Crypto Map "Tunnel0-head-0" 65537 …
    Map is a PROFILE INSTANCE
    Peer = 172.17.0.5,
    access-list permit gre host 172.17.0.1
      host 172.16.0.5
Crypto Map "Tunnel0-head-0" 65538 …
    Map is a PROFILE INSTANCE
    Peer = 172.16.1.1,
    access-list permit gre host 172.17.0.1
      host 172.16.1.1

# Four Layers for Troubleshooting: GRE Encapsulation Layer

**Common Issues**

- NHRP Registration fails
- Dynamic NBMA address change in spoke resulting inconsistent NHRP mapping in hub

# Common Issues: NHRP Registration Fails

## How to Detect?

- VPN tunnel between hub and spoke is up but unable to pass data traffic.

**Show crypto isa sa**

| dst | src | state | conn-id | slot | status |
|---|---|---|---|---|---|
| 172.17.0.1 | 172.16.1.1 | QM_IDLE | 1082 | 0 | ACTIVE |

**Show crypto IPsec sa**

local ident (addr/mask/prot/port): (172.16.1.1/255.255.255.255/47/0)
remote ident (addr/mask/prot/port): (172.17.0.1/255.255.255.255/47/0)
#pkts encaps: 154, #pkts encrypt: 154, #pkts digest: 154
#pkts decaps: 0, #pkts decrypt: 0, #pkts verify: 0
inbound esp sas:
spi: 0xF830FC95(4163959957)
outbound esp sas:
spi: 0xD65A7865(3596253285)

**Return traffic not coming back from other end of tunnel**

# Common Issues: NHRP Registration Fails (Cont.)

- Check NHS entry in spoke router.

NHS Request fail

```
Show  ip nhrp nhs detail
Legend: E=Expecting replies, R=Responding
Tunnel0:     172.17.0.1  E  req-sent 0  req-failed 30  repl-recv 0
Pending Registration Requests:
Registration Request: Reqid 4371, Ret 64  NHS 172.17.0.1
```

## How to Fix?

- Check spoke router tunnel interface configuration to make sure correct  ip address of NHS server is configured

Wrong NHS server address

```
interface Tunnel0
 ip address 10.0.0.9 255.255.255.0
 ip nhrp map 10.0.0.1 172.17.0.1
 ip nhrp map multicast 172.17.0.1
 ip nhrp nhs 172.17.0.1
```

```
interface Tunnel0
 ip address 10.0.0.9 255.255.255.0
 ip nhrp map 10.0.0.1 172.17.0.1
 ip nhrp map multicast 172.17.0.1
 ip nhrp nhs 10.0.0.1
```

Correct  NHS configuration is IP address of Hub tunnel interface

# Common Issues: NHRP Registration Fails (Cont.)

## How to verify?

- Verify NHS entry and ipsec encrypt/decrypt counters

```
sh ip nhrp nhs detail
Legend: E=Expecting replies, R=Responding
Tunnel0:      10.0.0.1 RE  req-sent 4  req-failed 0  repl-recv 3 (00:01:04 ago)
```

No request fail

```
Show crypto ipsec sa
local  ident (addr/mask/prot/port): (172.16.1.1/255.255.255.255/47/0)
remote ident (addr/mask/prot/port): (172.17.0.1/255.255.255.255/47/0)
#pkts encaps: 121, #pkts encrypt: 121, #pkts digest: 121
#pkts decaps: 118, #pkts decrypt: 118, #pkts verify: 118
inbound esp sas:
  spi: 0x1B7670FC(460747004)
outbound esp sas:
  spi: 0x3B31AA86(993110662)
```

- Verify routing protocol neighbor

```
sh ip eigrp neighbors
IP-EIGRP neighbors for process 10
H   Address          Interface    Hold   Uptime    SRTT   RTO   Q     Seq
                                  (sec)            (ms)         Cnt   Num
1   10.0.0.1          Tu0          11    00:21:20   18    200    0    497
```

# Common Issues: Dynamic NBMA Address Change in Spoke Resulting Inconsistent NHRP Mapping in Hub

- **Problem Description:**

  "Dynamic NBMA address change in spoke resulting inconsistent NHRP mapping in hub until NHRP registration with previous NBMA address expired"

- Show commands in hub before NBMA address change

```
Hub# show ip nhrp
10.0.0.11/32 via 10.0.0.11,Tunnel0 created 16:18:11,expire 00:28:47
Type: dynamic, Flags: unique nat registered,
NBMA address: 172.16.2.2
```

```
Hub # Show crypto socket
Tu0 Peers (local/remote): 172.17.0.1/172.16.2.2
      Local Ident  (addr/mask/port/prot): (172.17.0.1/255.255.255.255/0/47)
      Remote Ident (addr/mask/port/prot): (172.16.2.2/255.255.255.255/0/47)
      IPsec Profile: "dmvpn"
      Socket State: Open)
```

# Common Issues: Dynamic NBMA Address Change in Spoke Resulting Inconsistent NHRP Mapping in Hub

```
Hub# Show crypto ipsec sa
interface: Tunnel0
Crypto map tag: Tunnel0-head-0,
local crypto endpoint:172.17.0.1
Remote crypto endpoint:172.16.2.2
#pkts encaps: 13329,
#pkts decaps: 13326,
inbound esp sas:
  spi: 0xFEAB438C(4272636812)
outbound esp sas:
  spi: 0xDD07C33A(3708273466)
```

```
Hub# Show crypto map
Crypto Map "Tunnel0-head-0" 65540
Map is a PROFILE INSTANCE.
Peer = 172.16.2.2
        Extended IP access list
        access-list  permit gre host 172.17.0.1  host 172.16.2.2
        Current peer: 172.16.2.2
```

## How to Detect?

- Inconsistency after NBMA address change in spoke

```
Hub# show ip nhrp
10.0.0.11/32 via 10.0.0.11, Tunnel0 created 17:37:25, expire 00:09:34
Type: dynamic, Flags: unique nat registered used
NBMA address: 172.16.2.2
```

NHRP shows no entry for 172.16.2.3 still holding entry for previous NBMA address 172.16.2.2

# Common Issues: Dynamic NBMA Address Change in Spoke Resulting Inconsistent NHRP Mapping in Hub

## How to Detect? (Cont.)

```
Hub# show crypto map
Crypto Map "Tunnel0-head-0" 65540 ipsec-isakmp
        Map is a PROFILE INSTANCE.
        Peer = 172.16.2.2
        Extended IP access list
        access-list  permit gre host 172.17.0.1 host 172.16.2.2
        Current peer: 172.16.2.2
Crypto Map "Tunnel0-head-0" 65541 ipsec-isakmp
        Map is a PROFILE INSTANCE.
        Peer = 172.16.2.3
        Extended IP access list
        access-list  permit gre host 172.17.0.1  host 172.16.2.3
        Current peer: 172.16.2.3
```

Crypto map entry for both previous and new NBMA address of spoke

```
Hub# Show crypto socket
Tu0 Peers (local/remote): 172.17.0.1/172.16.2.2
      Local Ident  (addr/mask/port/prot): (172.17.0.1/255.255.255.255/0/47)
      Remote Ident (addr/mask/port/prot): (172.16.2.2/255.255.255.255/0/47)
      IPsec Profile: "dmvpn"
      Socket State: Open
Tu0 Peers (local/remote): 172.17.0.1/172.16.2.3
      Local Ident  (addr/mask/port/prot): (172.17.0.1/255.255.255.255/0/47)
      Remote Ident (addr/mask/port/prot): (172.16.2.3/255.255.255.255/0/47)
      IPsec Profile: "dmvpn"
      Socket State: Open
```

Old NBMA address

New NBMA address

# Common Issues: Dynamic NBMA Address Change in Spoke Resulting Inconsistent NHRP Mapping in Hub

## How to Detect? (Cont.)

- debug nhrp packet in hub router to check NHRP registration request /reply.

```
Hub# debug nhrp packet
NHRP: Receive Registration Request via Tunnel0 vrf 0, packet size: 104
 (F) afn: IPv4(1), type: IP(800), hop: 255, ver: 1
  shtl: 4(NSAP), sstl: 0(NSAP)
 (M) flags: "unique nat ", reqid: 9480
  src NBMA: 172.16.2.3
  src protocol: 10.0.0.11, dst protocol: 10.0.0.1
 (C-1) code: no error(0)
  prefix: 255, mtu: 1514, hd_time: 600
NHRP: Attempting to send packet via DEST 10.0.0.11
NHRP: Encapsulation succeeded.  Tunnel IP addr 172.16.2.3
NHRP: Send Registration Reply via Tunnel0 vrf 0, packet size: 124,  src: 10.0.0.1, dst: 10.0.0.11
 (F) afn: IPv4(1), type: IP(800), hop: 255, ver: 1
  shtl: 4(NSAP), sstl: 0(NSAP)
 (M) flags: " unique nat ", reqid: 9480
  src NBMA: 172.16.2.3
  src protocol: 10.0.0.11, dst protocol: 10.0.0.1
 (C-1) code: unique address registered already(14)
  prefix: 255, mtu: 1514, hd_time: 600
```

**C-1 code shows NBMA address is already registered , that is why it is not updating nhrp mapping table with new NBMA address**

# Common Issues: Dynamic NBMA Address Change in Spoke Resulting Inconsistent NHRP Mapping in Hub

- Spoke router shows the error message indicating  about NBMA address already registered

%NHRP-3-PAKREPLY: Receive Registration Reply packet with error - **unique address registered already(14)**

**How to Fix?**

- "ip nhrp registration no-unique" command in tunnel interface of dynamic
- NBMA address spoke router

```
Spoke# show run interface tunnel0
interface Tunnel0
 ip address 10.0.0.11 255.255.255.0
 ip nhrp map 10.0.0.1 172.17.0.1
 ip nhrp map multicast 172.17.0.1
 ip nhrp holdtime 600
 ip nhrp nhs 10.0.0.1
 ip nhrp registration no-unique
 :
tunnel protection ipsec profile dmvpn
```

**To enable the client to not set the unique flag in the Next Hop Resolution Protocol (NHRP) request and reply packets**

# Common Issues: Dynamic NBMA Address Change in Spoke Resulting Inconsistent NHRP Mapping in Hub

## How to Verify?

```
Hub# debug nhrp packet
NHRP: Receive Registration Request via Tunnel0 vrf 0, packet size: 104
 (F) afn: IPv4(1), type: IP(800), hop: 255, ver: 1
   shtl: 4(NSAP), sstl: 0(NSAP)
 (M) flags: "nat ", reqid: 9462
   src NBMA: 172.16.2.4
   src protocol: 10.0.0.11, dst protocol: 10.0.0.1
 (C-1) code: no error(0)
   prefix: 255, mtu: 1514, hd_time: 600
NHRP: Tu0: Creating dynamic multicast mapping  NBMA: 172.16.2.4
NHRP: Attempting to send packet via DEST 10.0.0.11
NHRP: Encapsulation succeeded.  Tunnel IP addr 172.16.2.4
NHRP: Send Registration Reply via Tunnel0 vrf 0, packet size: 124
   src: 10.0.0.1, dst: 10.0.0.11
 (F) afn: IPv4(1), type: IP(800), hop: 255, ver: 1
   shtl: 4(NSAP), sstl: 0(NSAP)
 (M) flags: "nat ", reqid: 9462
   src NBMA: 172.16.2.4
   src protocol: 10.0.0.11, dst protocol: 10.0.0.1
 (C-1) code: no error(0)
   prefix: 255, mtu: 1514, hd_time: 600
```

**Unique address command result no unique flag**
**C-1 code shows no error**

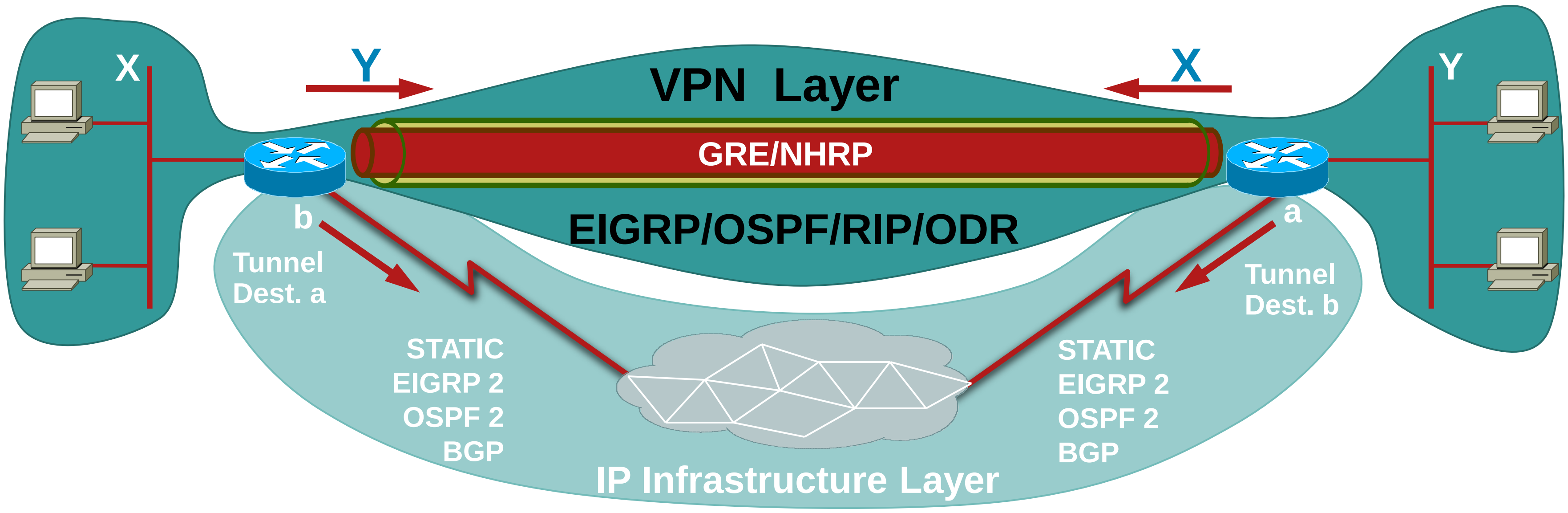
```
Hub#sh ip nhrp
10.0.0.11/32 via 10.0.0.11, Tunnel0 created 01:04:32, expire 00:07:06
  Type: dynamic, Flags: nat registered
  NBMA address: 172.16.2.4
```

**Unique flag not set**

# Four Layers for Troubleshooting: VPN Routing Layer

- The VPN routing layer—this is routing packets in/out of the p-pGRE and/or mGRE interfaces on the tunnel endpoint routers. This is done by running a dynamic routing protocol over the DMVPN tunnels



# Four Layers for Troubleshooting: VPN Routing Layer

- DMVPN Component-routing
- Regular IP networks

    IP routing updates and data packets traverse same physical/logical links

    Routing Protocol monitors state of all links that data packets can use

- DMVPN IP networks

    IP routing updates and IP multicast data packets only traverse hub-and-spoke tunnels

    Unicast IP data packets traverse both hub-and-spoke and direct dynamic spoke-spoke tunnels

    Routing protocol doesn’t monitor state of spoke-spoke tunnels

# Four Layers for Troubleshooting: VPN Routing Layer

- Check for routing neighbor and lifetime
  - show ip route [eigrp | ospf | rip ]
  - show ip protocol
  - show ip [ eigrp | ospf ] neighbor
- Check multicast replication and connectivity
  - show ip nhrp multicast
  - ping [ 224.0.0.10 (eigrp) | 224.0.0.5 (ospf) | 224.0.0.9 (rip) ]
  - ping <tunnel-subnet-broadcast-address>
    - Example:  10.0.0.0/24 → 10.0.0.255
- Debug
  - Various debug commands depending on routing protocol

# Four Layers for Troubleshooting: VPN Routing Layer—Common Issues

**Common Issues:**

- Routing protocol neighbor not established

# Four Layers for Troubleshooting: VPN Routing Layer—Common Issues

## Problem

- Spokes unable to establish routing protocol neighbor relationship
- **How to detect?**

**Hub# show ip eigrp neighbors**
**IP-EIGRP neighbors for process 10**

| H | Address | Interface | Hold (sec) | Uptime | SRTT (ms) | RTO | Q Cnt | Seq Num |
|---|---|---|---|---|---|---|---|---|
| 2 | 10.0.0.9 | Tu0 | 13 | 00:00:37 | 1 | 5000 | 1 | 0 |
| 0 | 10.0.0.5 | Tu0 | 11 | 00:00:47 | 1587 | 5000 | 0 | 1483 |
| 1 | 10.0.0.11 | Tu0 | 13 | 00:00:56 | 1 | 5000 | 1 | 0 |

**Syslog message**

**%DUAL-5-NBRCHANGE: IP-EIGRP(0) 10: Neighbor 10.0.0.9 (Tunnel0) is down: retry limit exceeded**

# Four Layers for Troubleshooting: VPN Routing Layer—Common Issues (Cont.)

```
Hub# show ip route eigrp
                 172.17.0.0/24 is subnetted, 1 subnets
C       172.17.0.0 is directly connected, FastEthernet0/0
    10.0.0.0/24 is subnetted, 1 subnets
C       10.0.0.0 is directly connected, Tunnel0
C    192.168.0.0/24 is directly connected, FastEthernet0/1
S*   0.0.0.0/0 [1/0] via 172.17.0.100
```

- **How to fix?**

  Verify NHRP multicast mapping is configured, in hub it is require to have dynamic nhrp multicast mapping configured in hub tunnel interface

```
interface Tunnel0
 ip address 10.0.0.1 255.255.255.0
 ip mtu 1400
 no ip next-hop-self eigrp 10
 ip nhrp authentication test
 ip nhrp network-id 10
 no ip split-horizon eigrp 10
 tunnel mode gre multipoint
```

```
interface Tunnel0
 ip address 10.0.0.1 255.255.255.0
 ip mtu 1400
 no ip next-hop-self eigrp 10
 ip nhrp authentication test
 ip nhrp map multicast dynamic
 ip nhrp network-id 10
 no ip split-horizon eigrp 10
 tunnel mode gre multipoint
```

Allows NHRP to automatically add spoke routers to the multicast NHRP mappings

# Four Layers for Troubleshooting: VPN Routing Layer—Common Issues (Cont.)

- **How to verify?**

```
Hub # sh ip eigrp neighbors
IP-EIGRP neighbors for process 10
H   Address          Interface     Hold Uptime   SRTT    RTO   Q    Seq
                                        (sec)     (ms)        Cnt  Num
2   10.0.0.9          Tu0            12      00:16:48  13    200    0   334
1   10.0.0.11         Tu0            13      00:17:10  11    200    0   258
0   10.0.0.5          Tu0            12      00:48:44  1017  5000   0   1495

Hub# show ip route

    172.17.0.0/24 is subnetted, 1 subnets
C      172.17.0.0 is directly connected, FastEthernet0/0
D   192.168.11.0/24 [90/2944000] via 10.0.0.11, 00:16:12, Tunnel0
    10.0.0.0/24 is subnetted, 1 subnets
C      10.0.0.0 is directly connected, Tunnel0
C   192.168.0.0/24 is directly connected, FastEthernet0/1
D   192.168.2.0/24 [90/2818560] via 10.0.0.9, 00:15:45, Tunnel0
S*  0.0.0.0/0 [1/0] via 172.17.0.100
```

**Spokes routes learned via EIGRP protocol**

Thank you.