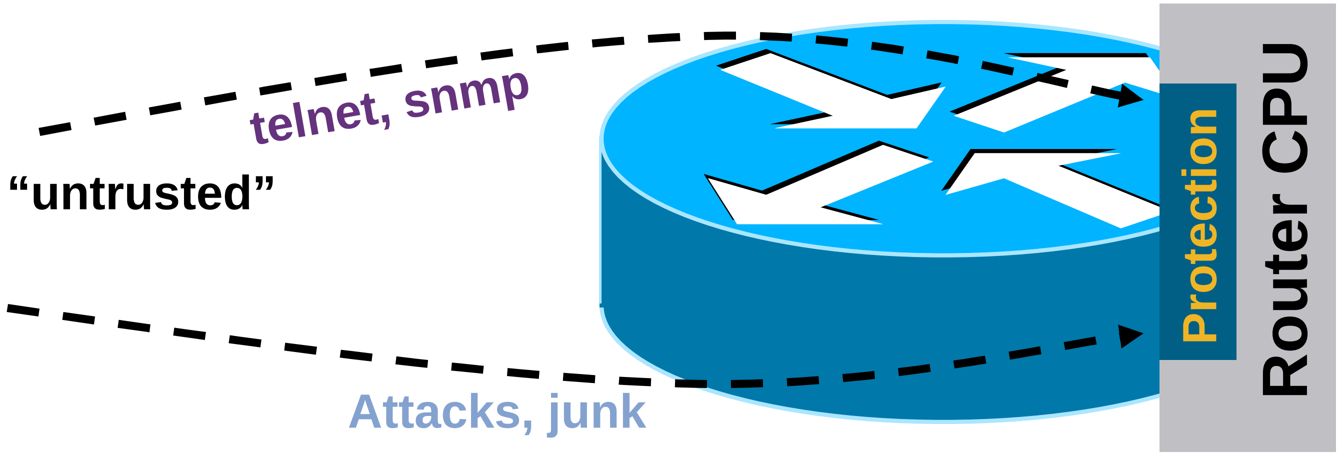
Network Core Infrastructure Best Practices

Yusuf Bhaiji

Agenda

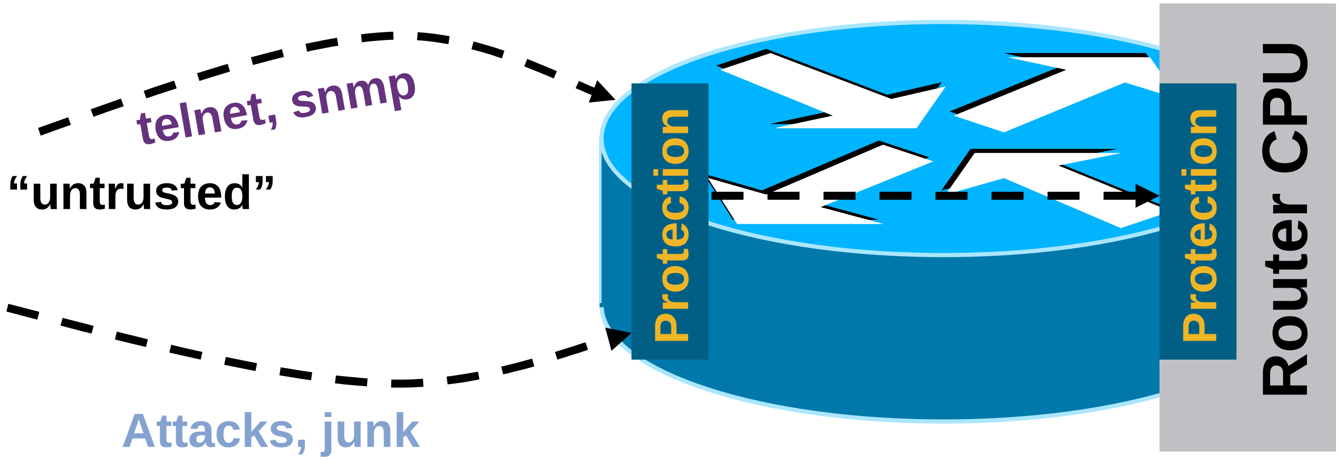
- Infrastructure Protection Overview
- Understanding Routers and Planes
- Infrastructure Protection from the Inside Out
    - Router Hardening: Traditional Methods
    - Router Hardening: Protecting the CPU
    - Network Hardening

Router Hardening: Traditional Methods



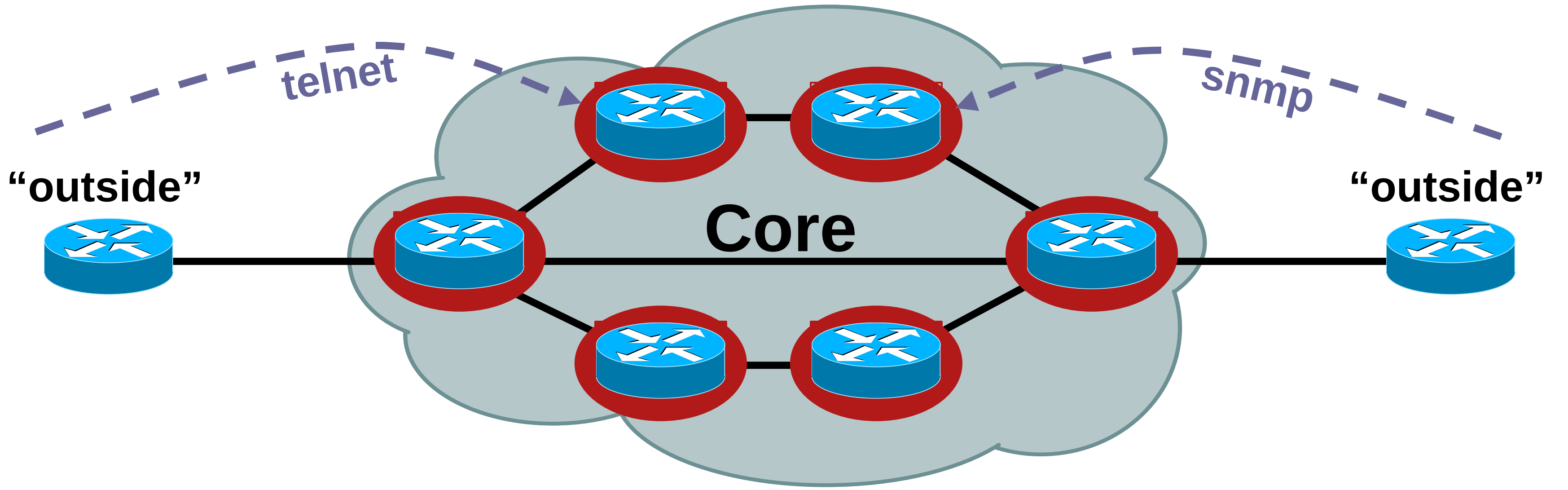
- We will look at best practices on securing the CPU

Router Hardening: Protecting the CPU



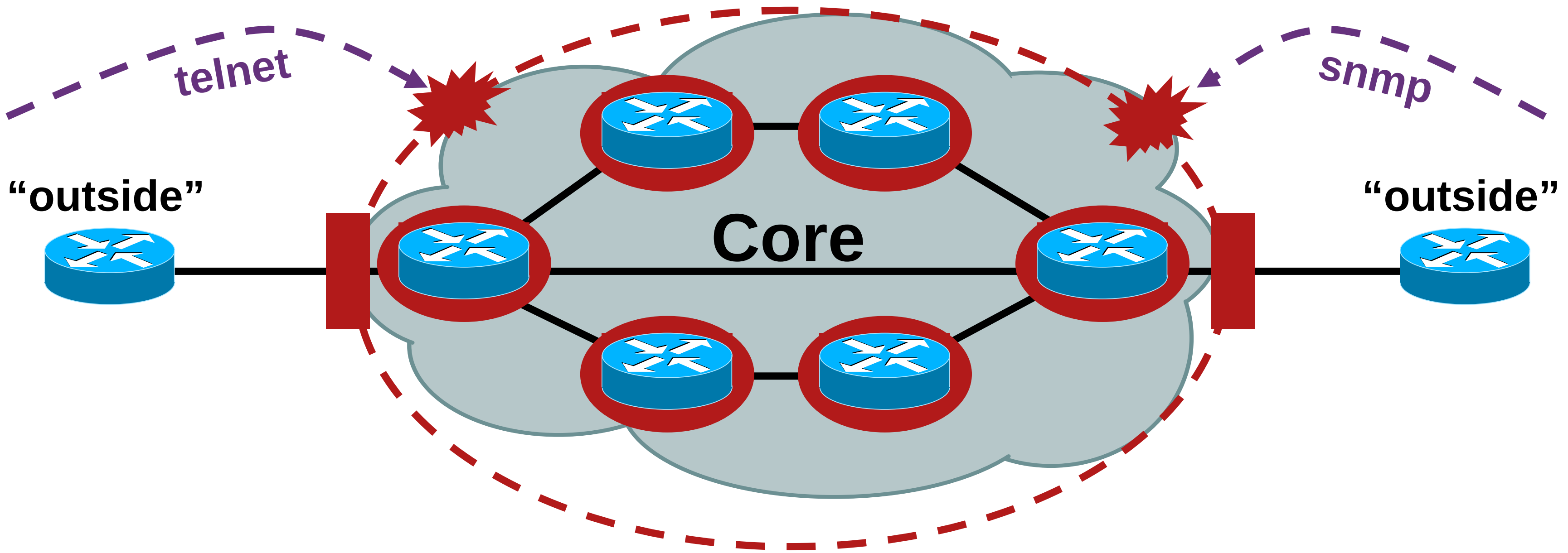
- We will look at best practices on preventing unwanted traffic from reaching the CPU

The Old World: Network Edge



- Core routers individually secured
- Every router accessible from outside

Network Hardening



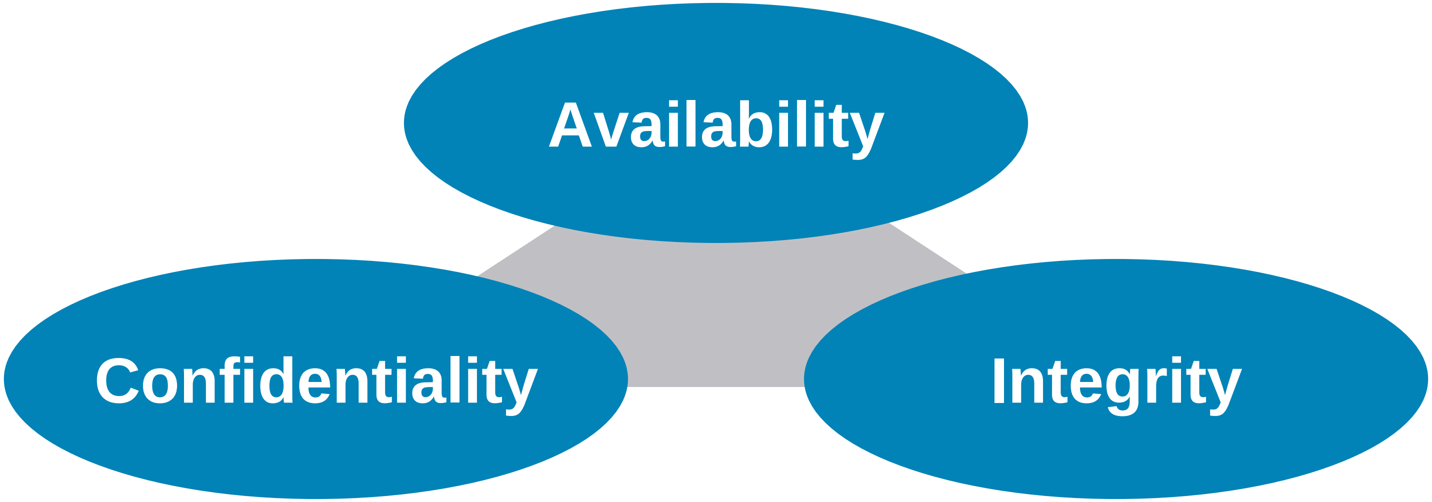
- We will look at best practices on preventing unwanted traffic from reaching the core routers

Agenda

- Infrastructure Protection Overview
- Understanding Routers and Planes
- Infrastructure Protection from the Inside Out
  - Router Hardening: Traditional Methods
  - Router Hardening: Protecting the CPU
  - Network Hardening

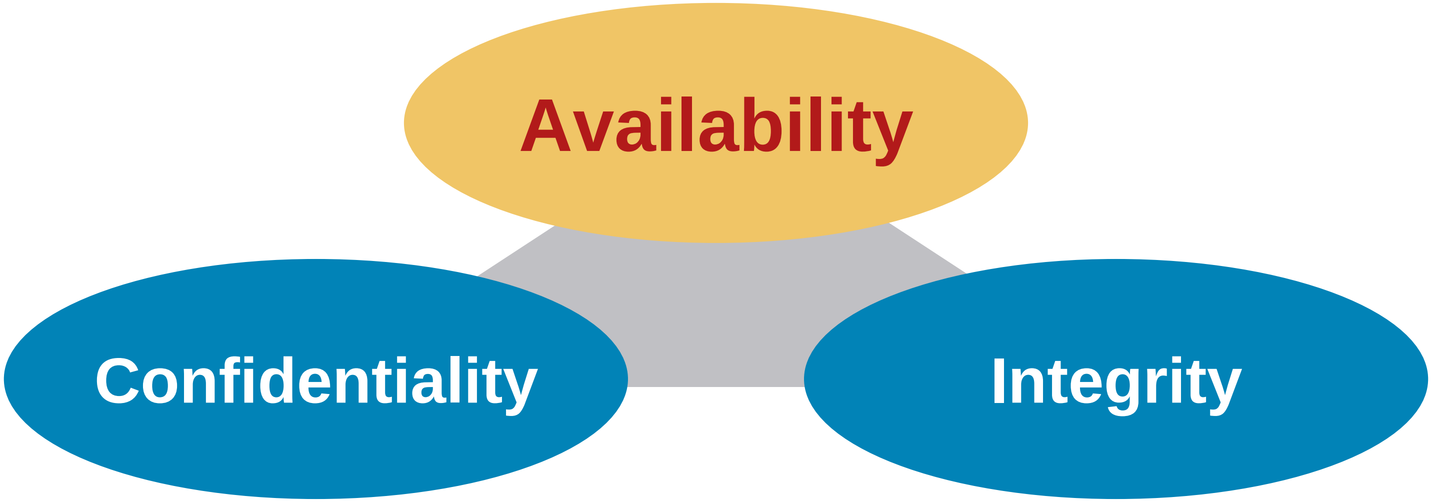
Infrastructure Protection Overview

Three Security Characteristics



- The goal of security is to maintain these three characteristics

Three Security Characteristics



- Primary goal of infrastructure security and this session is maintaining availability

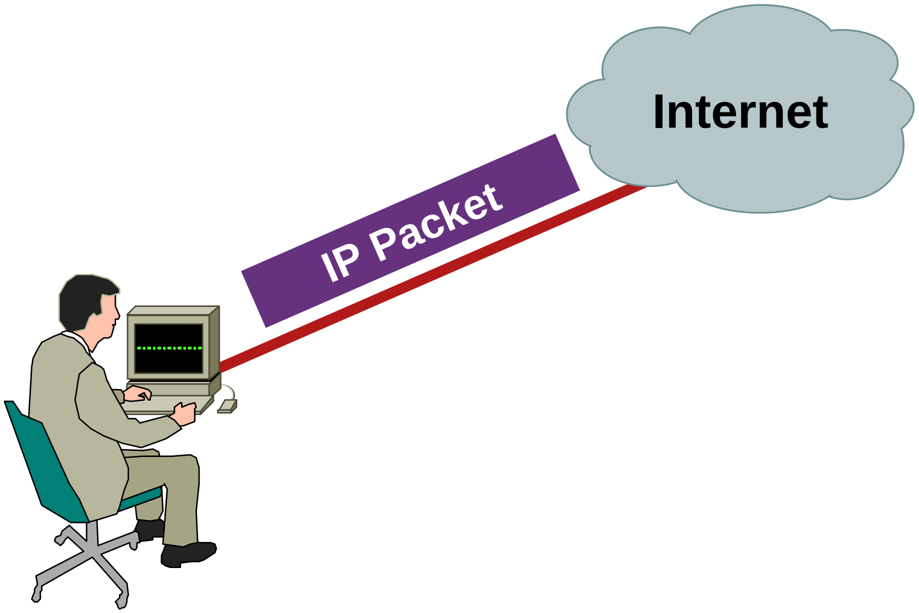
Network Availability: Protect the Infrastructure

- Security is the heart of internetworking's future; we have moved from an Internet of implicit trust to an Internet of pervasive distrust
- No packet can be trusted; all packets must earn that trust through a network device's ability to inspect and enforce policy

  What does it mean for a packet to be trusted?

- Protecting the infrastructure is the most fundamental security requirement
- Infrastructure protection should be included in all high availability designs
- A secure infrastructure forms the foundation for continuous business operations

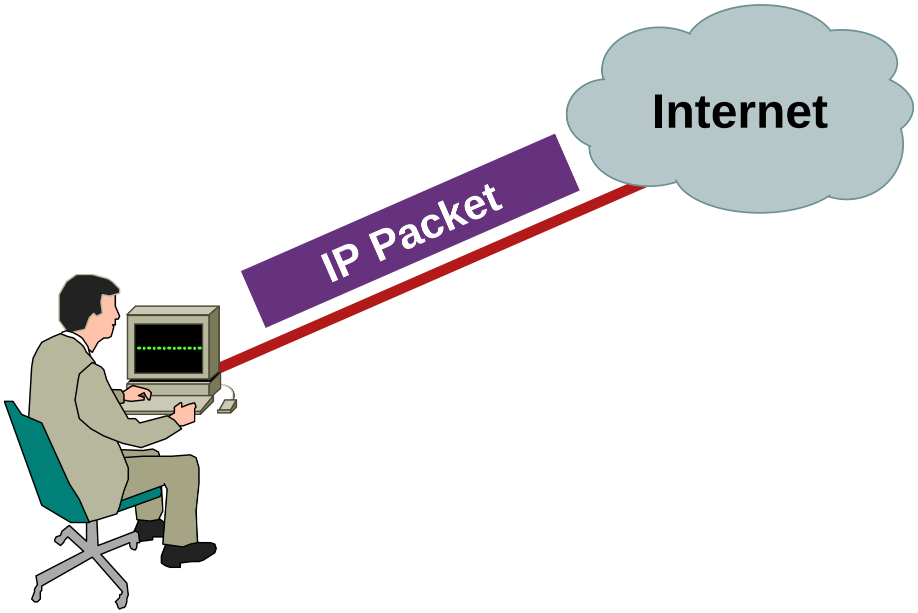
It Is All About the Packet



Once a packet gets into the Internet, some device, somewhere has to do one of two things:

- Deliver the packet
- Drop the packet

It Is All About the Packet



- In the context of an attack, the questions are by whom and where will that packet be dropped

Understand the Threats

- Internal
  - Inadvertent human error (fat finger attack)
  - Malicious insider
- External
  - Worms
  - Packet floods
  - Security vulnerability
  - Intrusion
  - Route hijacking
  - Service attacks (DNS, voice, video, etc.)

Understand the Threats

- Internal
  - Inadvertent human error (fat finger attack)
  - Malicious insider
- External
  - **Worms**
  - **Packet floods**
  - **Security vulnerability**
  - Intrusion
  - Route hijacking
  - Service attacks (DNS, voice, video, etc.)

These Threats Will Be Our Main Focus

Taking a Measured Approach

The Techniques We Will Be Discussing Are Extremely Useful, but Must Be Applied in an Architecturally Sound, Situationally Appropriate, and Operationally Feasible Manner

- Don't try to do all of this at once—pick a technique with which you are comfortable and which you think will benefit you the most
- Pilot your chosen technique in a controlled manner, in a designated portion of your network
- Take the lessons learned from the pilot and work them into your general deployment plan and operational guidelines
- It is not uncommon to take 9–12 months to deploy

Agenda

- Infrastructure Protection Overview
- Understanding Routers and Planes
- Infrastructure Protection from the Inside Out
    - Router Hardening: Traditional Methods
    - Router Hardening: Protecting the CPU
    - Network Hardening

Understanding Routers and Planes

Routers and Planes

- A network device typically handles traffic in several different forwarding planes
- There are nuances to the definition of these planes

  IETF RFC3654 defines two planes: control and forwarding

  ITU X805 defines three planes: control, management, and end-user

  Cisco defines three planes: control, management, and data

Routers and Planes

- Traffic to the control and management plane is always destined to the device and is handled at process level ultimately:

  In hardware switched platforms, control/management plane traffic is sent to the RP/MSFC and then sent to the process level for processing

  In software switched platforms, it is sent directly to the process level for processing

- Traffic in the data plane is always destined through the device and is:

  Implemented in hardware on high end platforms

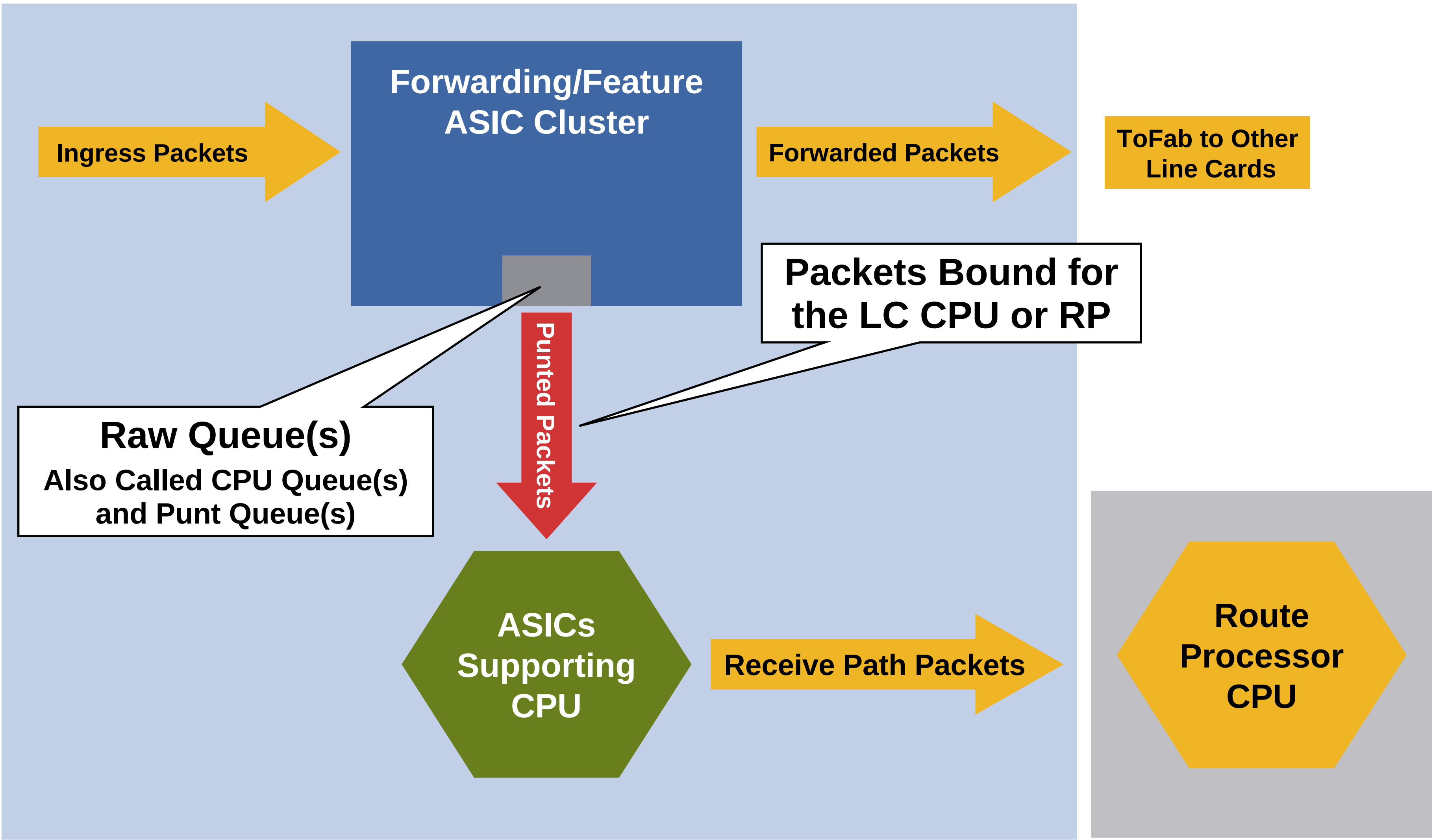
  CEF switched (in the interrupt) in software switched platforms

Routers and Planes

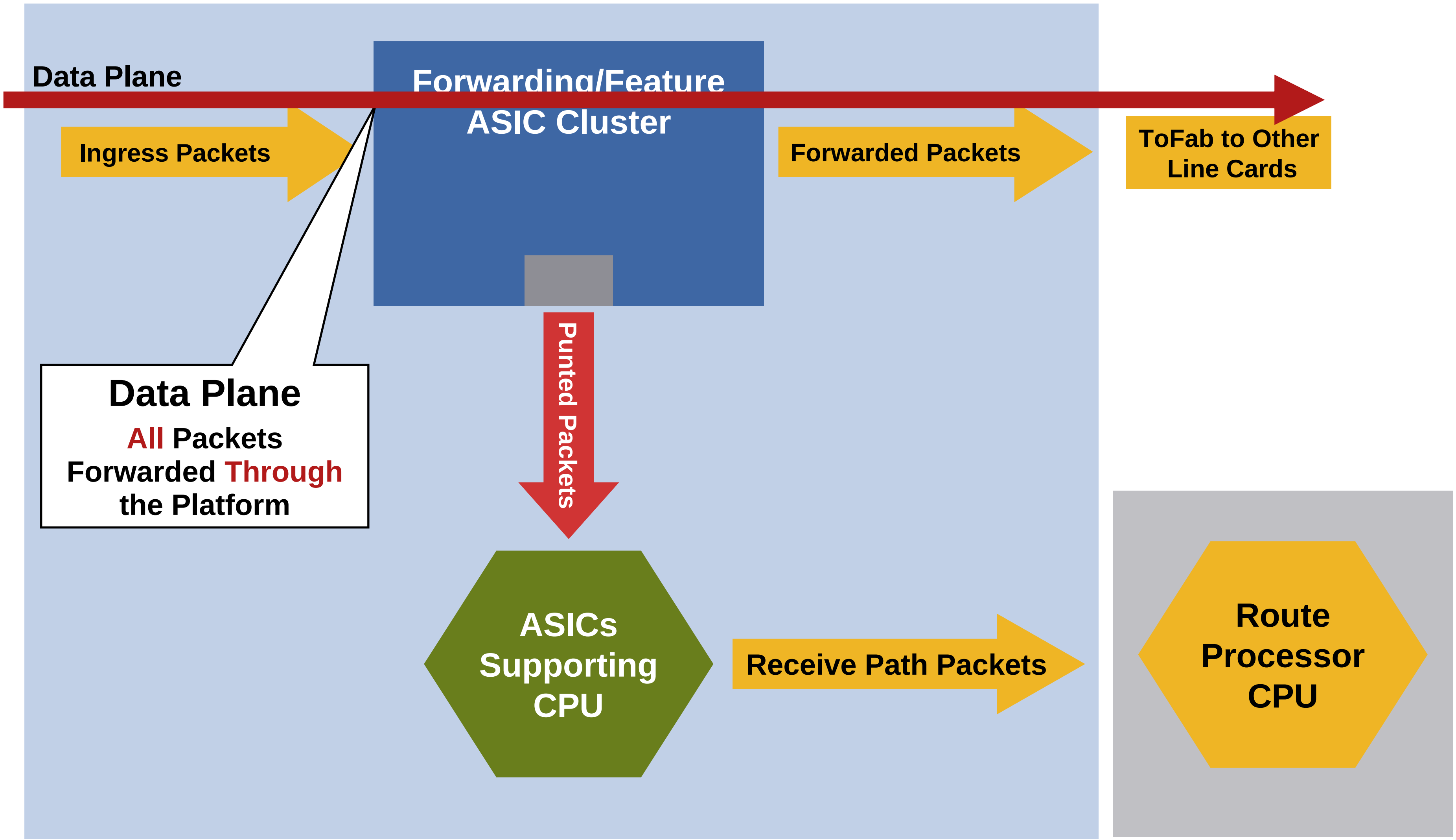
Some Data Plane Traffic May Also Reach the Control Plane

- Packets that are not routable reach the control plane so that ICMP unreachable messages can be generated
- Packets that have IP options set are also handled by the processor

ASIC Based Platform— Main Components

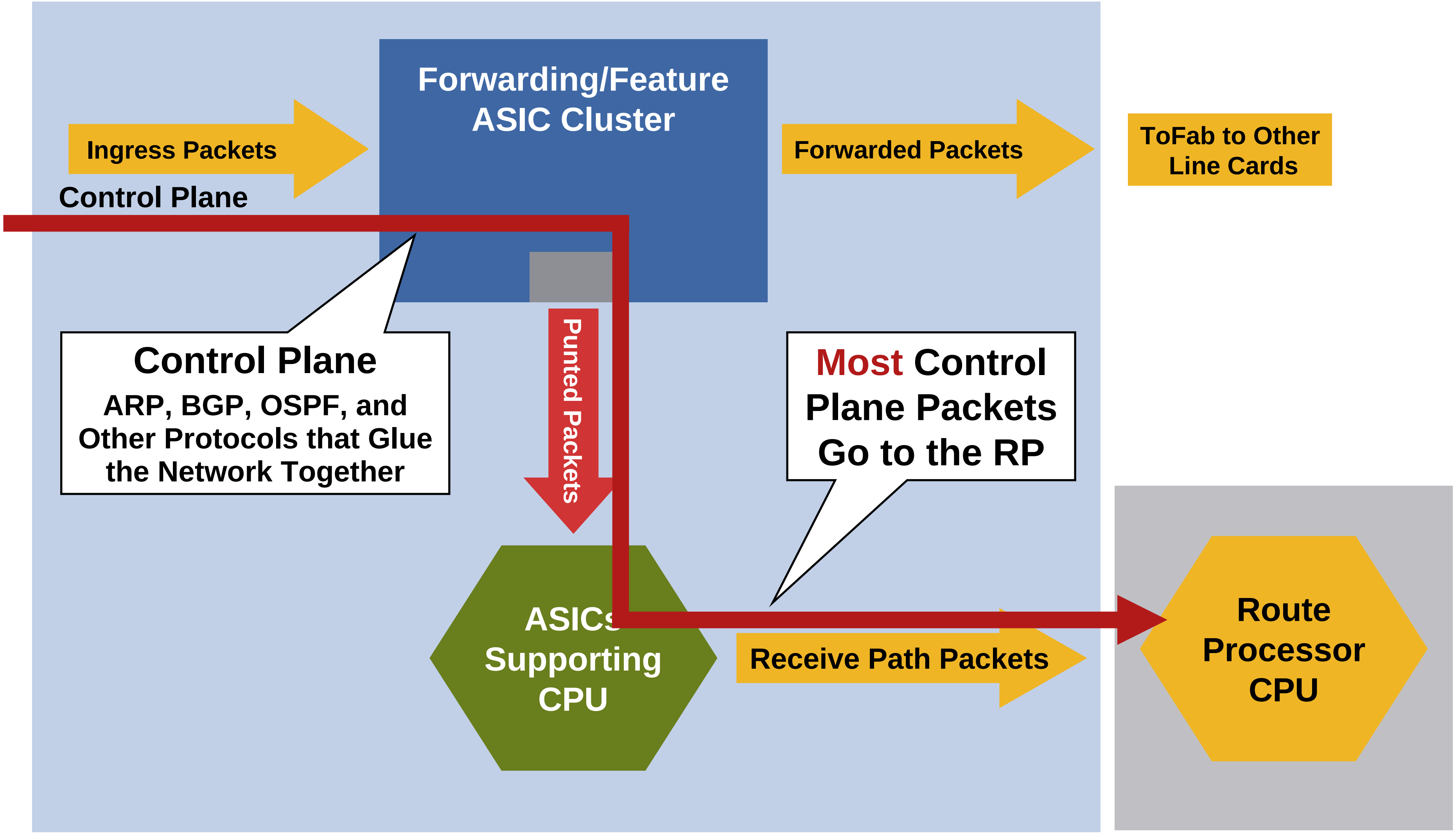


Data Plane

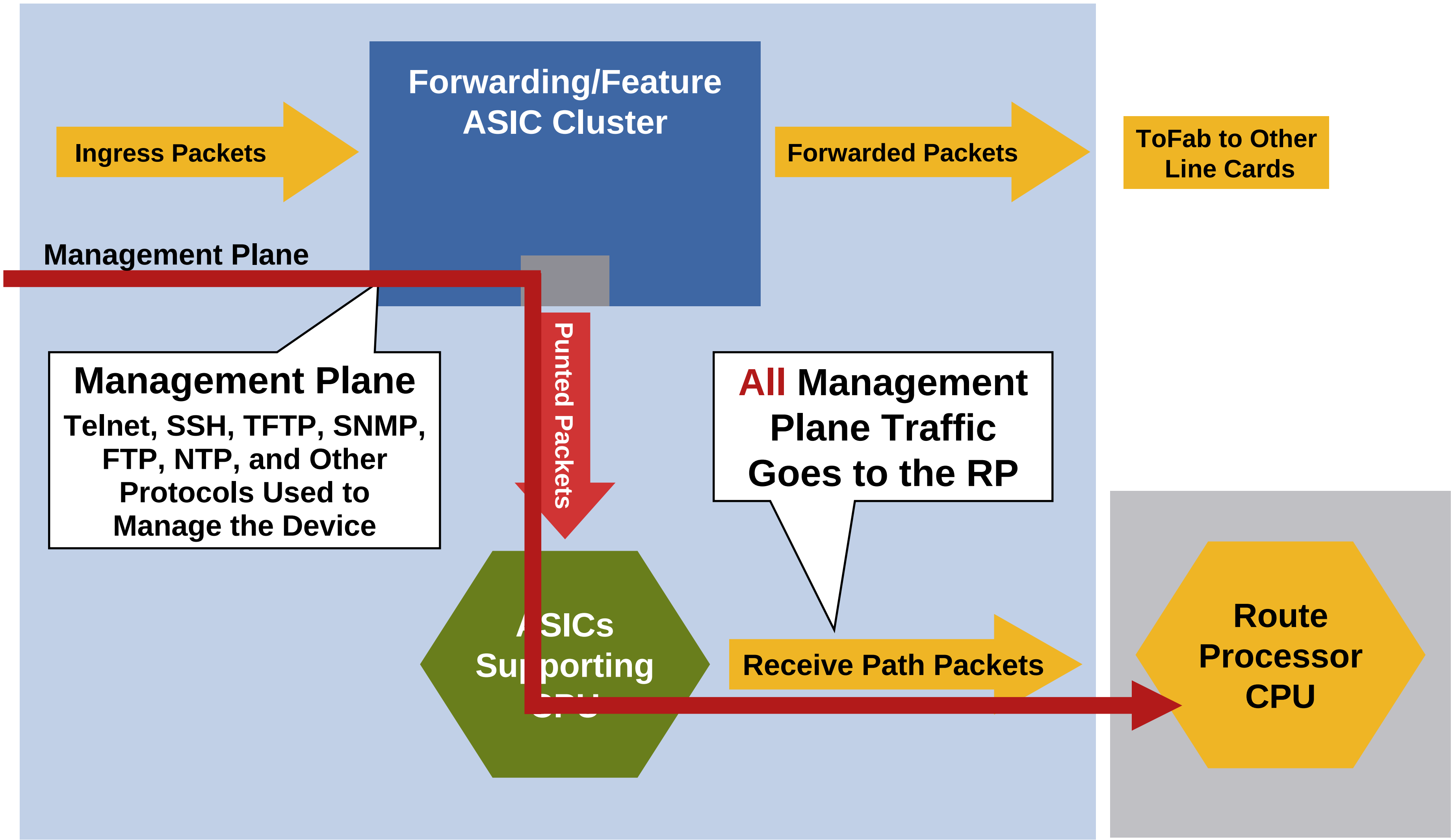


Data Plane
Forwarding/Feature ASIC Cluster
Ingress Packets
Forwarded Packets
ToFab to Other Line Cards
Data Plane
All Packets Forwarded Through the Platform
Punted Packets
ASICs Supporting CPU
Receive Path Packets
Route Processor CPU

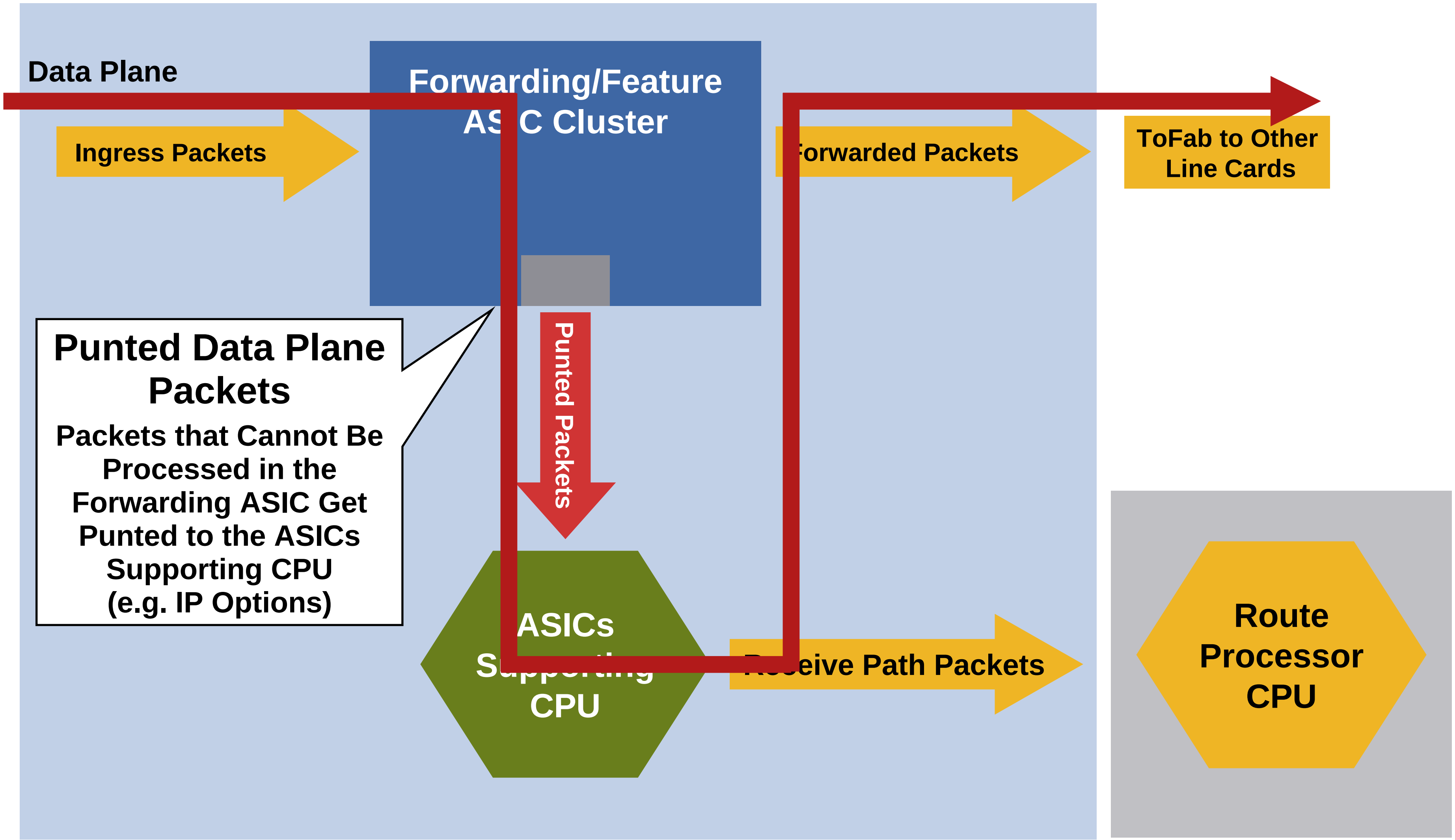
Control Plane



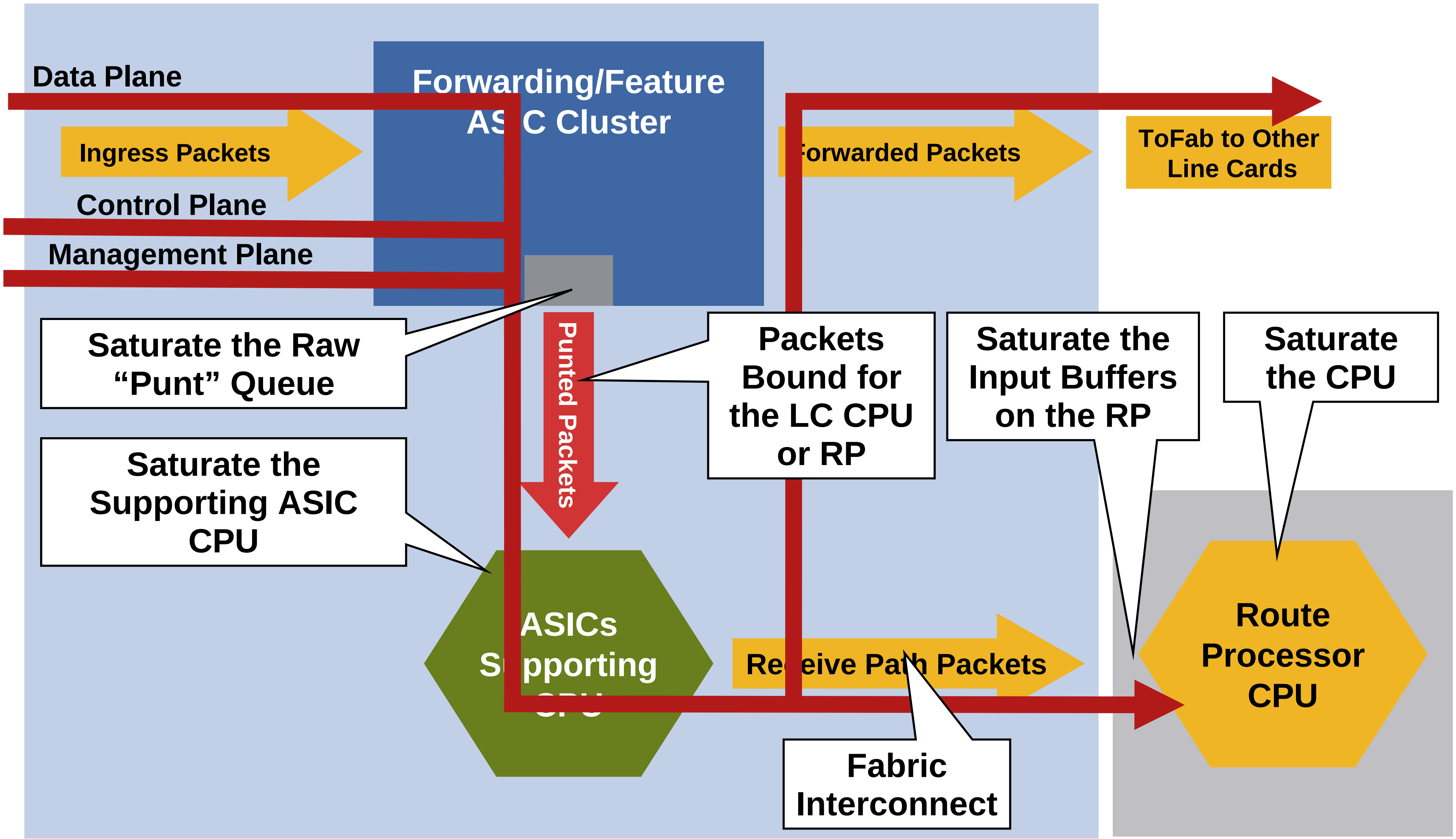
Management Plane



Data Plane Feature Punt



Attack Vectors



Agenda

- Infrastructure Protection Overview
- Understanding Routers and Planes
- Infrastructure Protection from the Inside Out
    - Router Hardening: Traditional Methods
    - Router Hardening: Protecting the CPU
    - Network Hardening

Router Hardening: Traditional Methods

Router Security Best Practices

- Many organizations publish guides to best practices around router security
- In addition to CCO resources, these include:

  http://www.first.org/resources/guides/

  http://www.sans.org/resources/policies/

  http://www.ietf.org/html.charters/opsec-charter.html

- Many of these best practices address threats that are outside the scope of this session
- There is usually an incident or story behind why various techniques are deployed
- Therefore, we will review a sample of the key points and features

Router Hardening: Traditional Methods

- Disable any unused protocols
  - no service tcp-small-servers
  - no cdp run
  - no crypto isakmp enable
- VTY ACLs
- SNMP Community ACL
- SNMP views
- Disable SNMP RW
  - Use SNMPv3 for RW if needed
- Prevent dead TCP sessions from utilizing all VTY lines
  - service tcp-keepalives-in
- Edge QoS enforcement
- Use secret password
  - Service password encryption is reversible and is only meant to prevent shoulder surfing
- Run AAA
  - Don’t forget Authorization and Accounting
- Disable extraneous interface features
  - no ip directed-broadcast
  - no ip proxy-arp
  - no ip redirects

Router Hardening: Traditional Methods

- Source address validation (RFC2827/BCP38, RFC3704/BCP84)

    ip verify unicast source reachable-via {any|rx}

    cable source-verify [dhcp]

    ip verify source [port-security]

- Disable source-routing

    no ip source-route

- Prefix-list filtering on eBGP peers
- BGP dampening
- BGP maximum-prefix
- MD5 on BGP and IGP
- Hardware-dependent issues

    Control ICMP unreachable generation

        ip icmp rate-limit unreachable

        ip icmp rate-limit unreachable DF

        interface null0
          no ip unreachables

    Ensure CPU cycles for management

        scheduler allocate

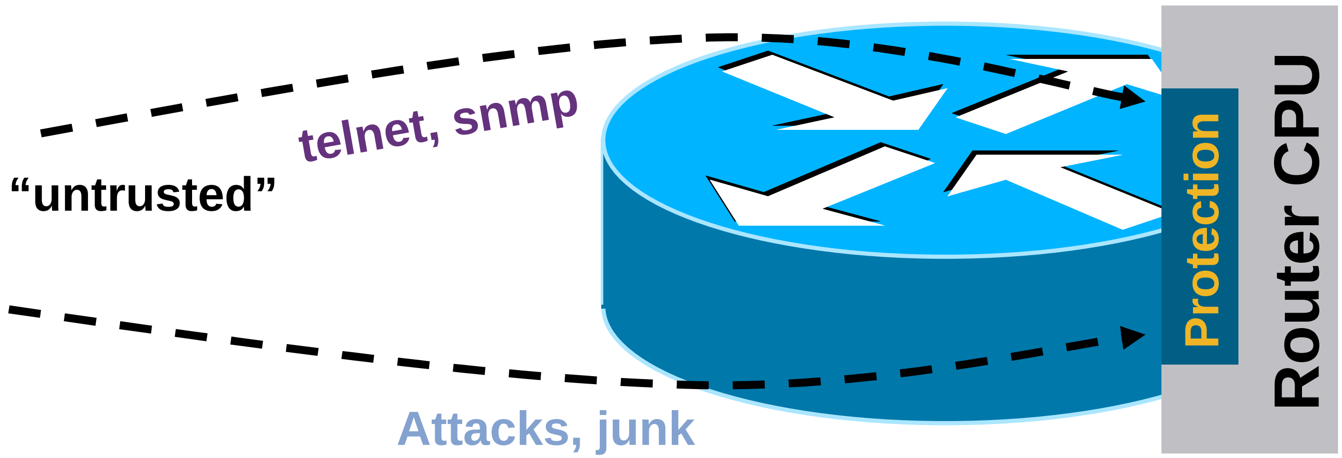
    Selective Packet Discard (SPD)

Agenda

- Infrastructure Protection Overview
- Understanding Routers and Planes
- Infrastructure Protection from the Inside Out
    - Router Hardening: Traditional Methods
    - Router Hardening: Protecting the CPU
    - Network Hardening

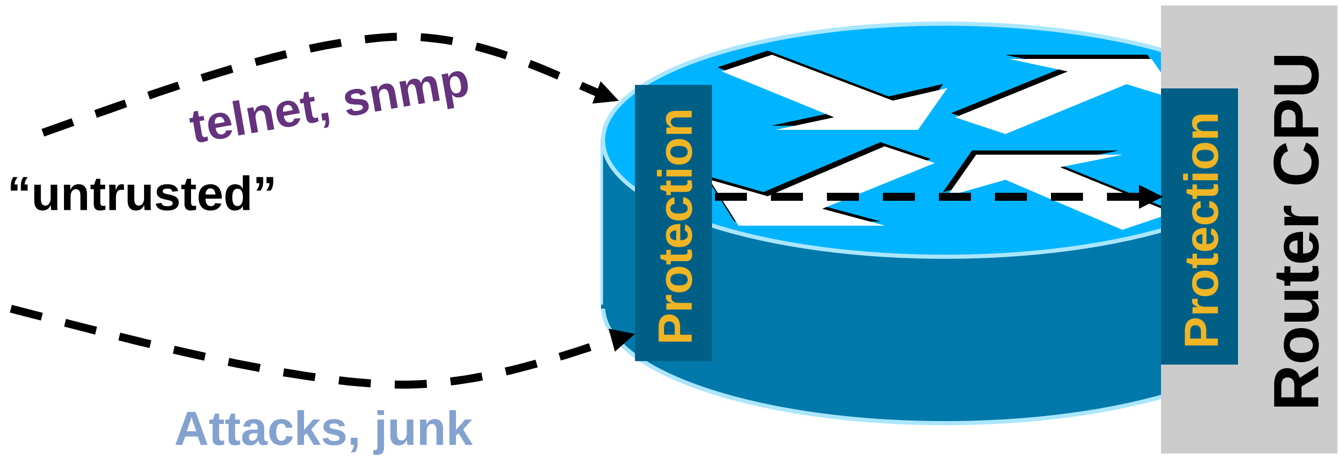
Router Hardening: Protecting the CPU

The Old World: Router Hardening



- Policy enforced at process level (VTY ACL, SNMP ACL, etc.)

The New World: Router Hardening



- Central policy enforcement, prior to process level
- Granular protection schemes
- On high-end platforms, hardware implementations

Router Hardening: Protecting the CPU Receive Access-Lists

Receive ACL Command

- Introduced in:

  12000: 12.0(21)S2/12.0(22)S

  7500: 12.0(24)S

  10720: 12.0(31)S

  10K-PRE2: 12.3(7)XI1

  Router(config)# ip receive access-list [number]

- Standard, extended, or compiled ACL
- As with other ACL types, show access-list provide ACE hit counts
- Log keyword can be used for more detail

Receive ACLs (rACLs)

- Receive ACLs filter traffic destined to the RP via receive adjacencies (generally control and management plane only)
- rACLs explicitly permit or deny traffic destined to the RP
- rACLs do **not** affect the data plane
- Traffic is filtering on the ingress line card (LC), prior to route processor (RP) processing
- rACLs enforce security policy by filtering who/what can access the router

Receive Adjacencies

- CEF entries for traffic destined to router, not through it

    Real interface(s)

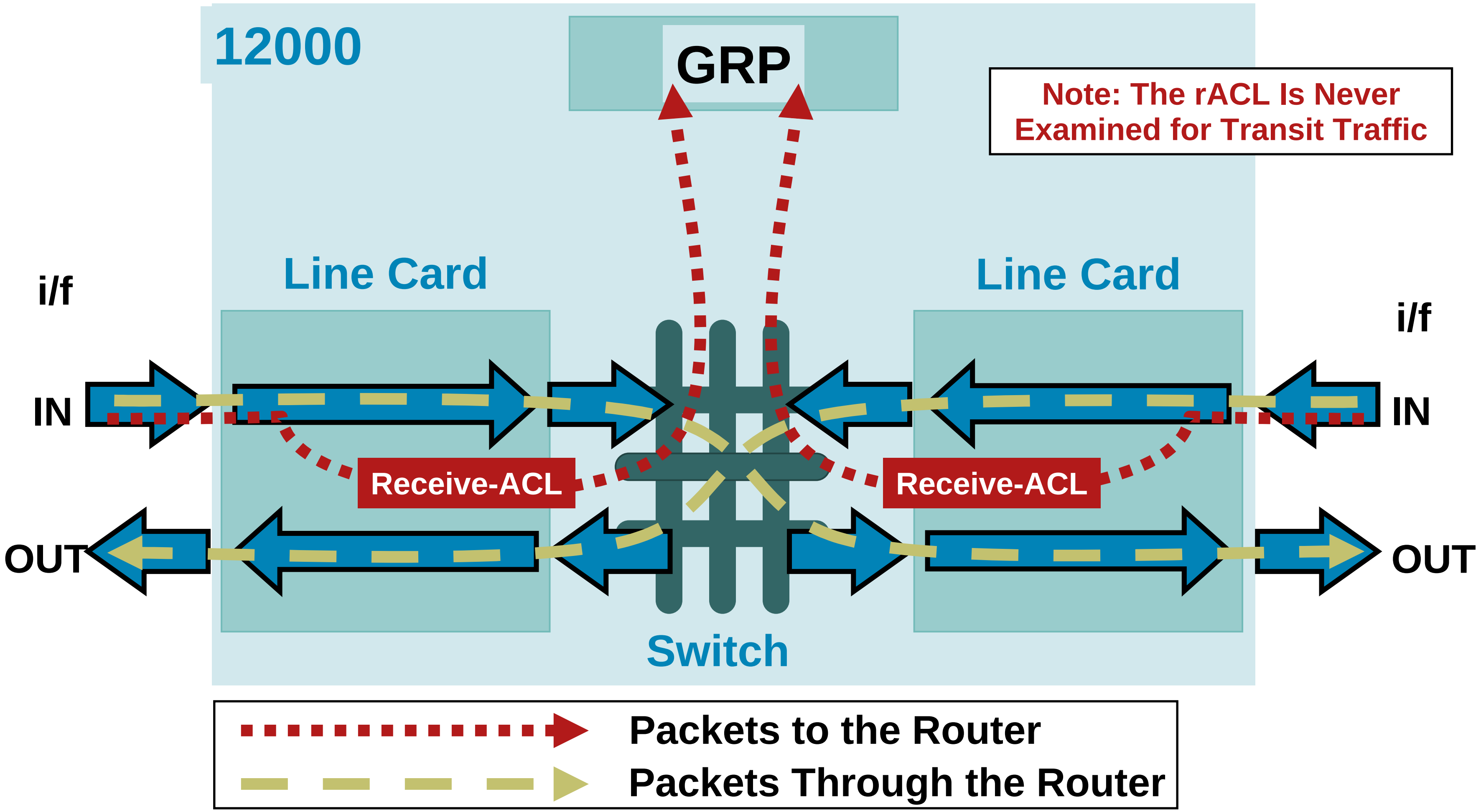
    Loopback interface(s)

```
c12008#sh ip cef
Prefix              Next Hop        Interface
0.0.0.0/32          receive
10.0.10.1/32        receive
10.1.1.0/24         10.0.3.1        Serial6/0
10.0.3.0/30         attached        Serial6/0
10.0.3.0/32         receive
10.0.3.2/32         receive
10.0.3.3/32         receive
224.0.0.0/24        receive
255.255.255.255/32  receive
```

- Packets with next hop receive are sent to the RP for processing

Receive ACL Traffic Flow

Router(config)# [no] ip receive access-list <num>



rACLs and Fragments

- Fragments can be denied via an rACL
- Denies fragments and classify fragment by protocol:

```
access-list 110 deny tcp any any fragments
access-list 110 deny udp any any fragments
access-list 110 deny icmp any any fragments
```

Using Classification ACL to build rACL

- Iterative Deployment
- Develop list of required protocols
- Develop address requirements
- Determine interface on router
    - Does the protocol access 1 interface?
    - Many interfaces?
    - Loopback or real?
- Deployment is an iterative process
    - Start with relatively “open” lists → tighten as needed

rACL: Iterative Deployment

- Step 1: Identify protocols/ports used in the network with a classification ACL

  Permit any any for various protocols/ports

  Get an understanding of what protocols communicate with the router

  Permit any any log at the end can be used to identify any missed protocols

  This should be done slowly to ensure no protocols are missed

- Step 2: Review identified packets, begin to filter access to the GRP/PRP

  Using list developed in step 1, permit only those protocols

  Deny any any at the end → basic protection

rACL: Iterative Deployment

- Step 3: Restrict a macro range of source addresses

  Only permit your CIDR block in the source field

  eBGP peers are the exception: they may fall outside CIDR block

- Step 4: Narrow the rACL permit statements: authorized source addresses

  Increasingly limit the source addresses to known sources: management stations, NTP peers, AAA server, etc.

rACL: Iterative Deployment

- Step 5: Limit the destination addresses on the rACL
  - Filter what interfaces are accessible to specific protocols
  - Does the protocol access loopbacks only? Real interfaces?
- Rinse, repeat
  - Remember, start slow and open
  - Gradually improve security over time
  - If you try and start very secure, you are increasing your chance of dropping legitimate traffic

rACL: Sample Entries

```
ip receive access-list 110
```

- Fragments

```
access-list 110 deny any any fragments
```

- OSPF

```
access-list 110 permit ospf host ospf_neighbour host 224.0.0.5
! DR multicast address, if needed
access-list 110 permit ospf host ospf_neighbour host 224.0.0.6
access-list 110 permit ospf host ospf_neighbour host local_ip
```

- BGP

```
access-list 110 permit tcp host bgp_peer host loopback eq bgp
```

- EIGRP

```
access-list 110 permit eigrp host eigrp_neighbour host 224.0.0.10
access-list 110 permit eigrp host eigrp_neighbour host local_ip
```

rACL: Sample Entries

- SSH/Telnet

```
access-list 110 permit tcp management_addresses host loopback eq 22
access-list 110 permit tcp management_addresses host loopback eq telnet
```

- SNMP

```
access-list 110 permit udp host NMS_stations host loopback eq snmp
```

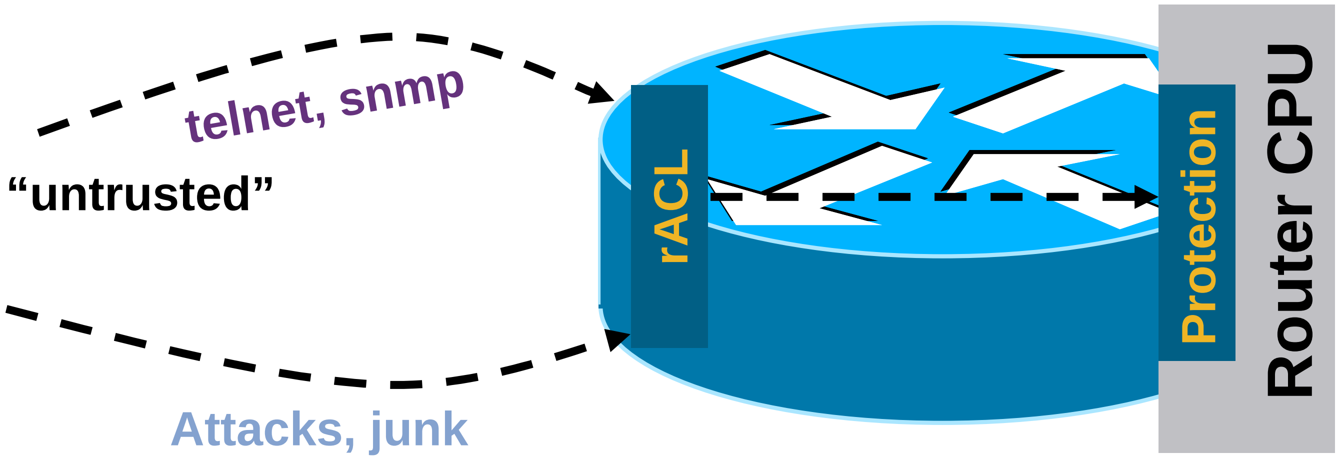
- Traceroute (router originated)

```
!Each hop returns a ttl exceeded (type 11, code 3) message and the final
destination returns an ICMP port unreachable (type 3, code 0)
access-list 110 permit icmp any routers_interfaces ttl-exceeded
access-list 110 permit icmp any routers_interfaces port-unreachable
```

- Deny Any

```
access-list 110 deny ip any any
```

rACLs: Summary



- Contain the attack: compartmentalize

    Protect the RP

- Widely deployed and highly effective

    If you have platforms that support rACLs, start planning a deployment

    Many ISPs use rACLs in conjunction with CoPP (next topic)

Router Hardening: Protecting the CPU Control Plane Policing

Control Plane Policing (CoPP)

- rACLs are great but

  Limited platform availability

  Limited granularity—permit/deny only

- Need to protect all platforms

  To achieve protection today, need to apply ACL to all interfaces

  Some platform implementation specifics

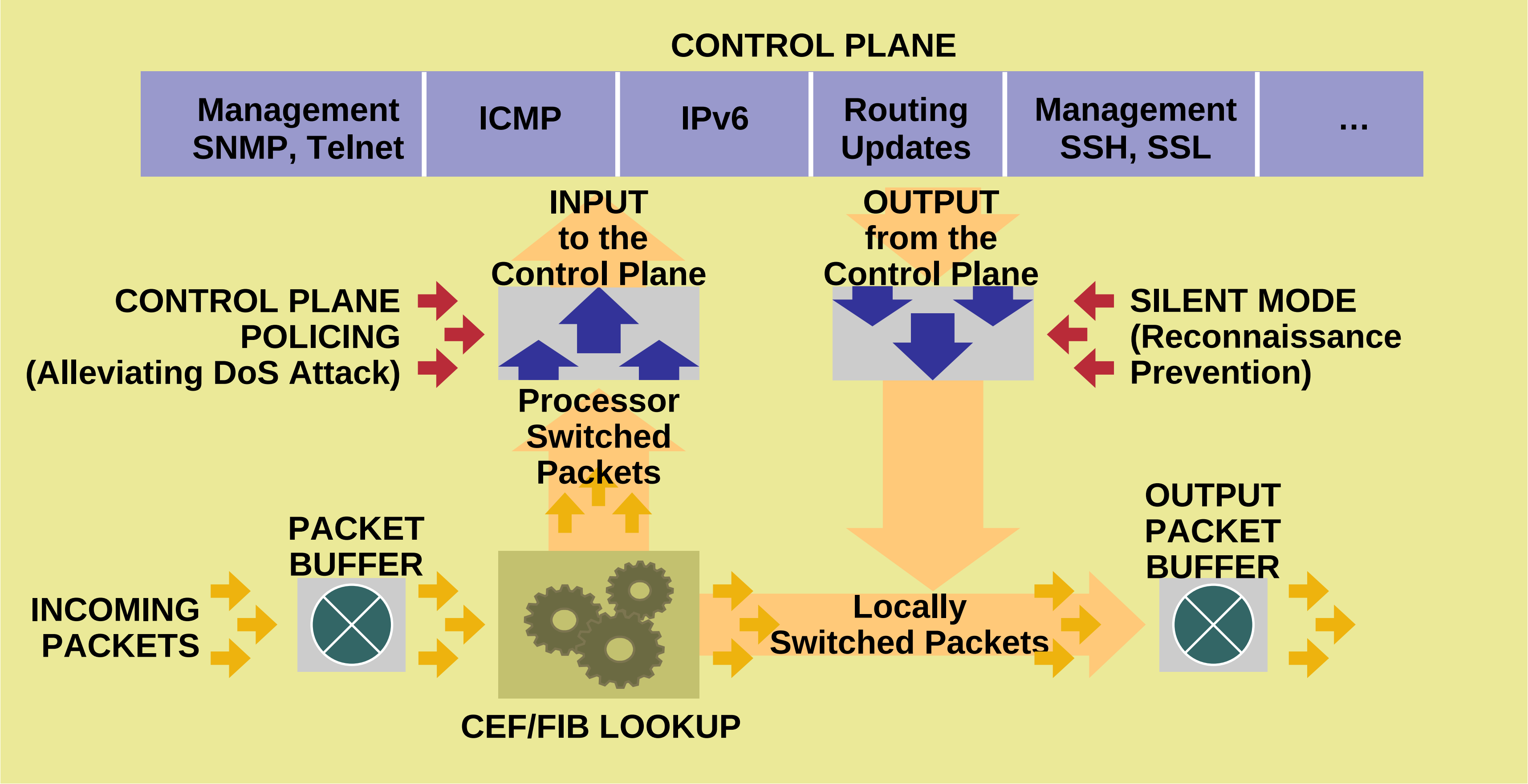
- Some packets need to be permitted but at limited rate

  Think ping :-)

Control Plane Policing (CoPP)

- CoPP uses the Modular QoS CLI (MQC) for QoS policy definition
- Consistent approach on all boxes
- Dedicated control-plane "interface"
  - Single point of application
- Highly flexible: permit, deny, rate limit
- Extensible protection
  - Changes to MQC (e.g. ACL keywords) are applicable to CoPP

Control Plane Policing Feature



Control Plane Policing (CoPP) Command

- Introduced in:

  12000: 12.0(29)S (aggregate mode)

  12000: 12.0(30)S (distributed mode)

  6500/7600: 12.2(18)SXD1

  10720: 12.0(32)S

  Most other platforms: 12.2(18)S/12.3(4)T

  ```
  Router(config)# control-plane [slot slot-number]
  Router(config-cp)# service-policy input control-plane-policy
  ```

- Uses the Modular QoS CLI (MQC) syntax for QoS policy definition
- Dedicated control-plane “interface” for applying QoS policies—single point of application
- Unlike rACL, CoPP handles data plane punts as well as control/management plane traffic

Deploying CoPP

- One option: attempt to mimic rACL behavior
  - CoPP is a superset of rACL
  - Apply rACL to a single class in CoPP
  - Same limitations as with rACL: permit/deny only
- Recommendation: develop multiple classes of control plane traffic
  - Apply appropriate rate to each
  - "Appropriate" will vary based on network, risk tolerance, and risk assessment
  - Be careful what you rate-limit
- Flexible class definition allows extension of model
  - Fragments, TOS, ARP

Configuring CoPP

Four Required Steps:

1. Define ACLs

    Classify traffic

2. Define class-maps

    Setup class of traffic

3. Define policy-map

    Assign QoS policy action to class of traffic (police, drop)

4. Apply CoPP policy to control plane “interface”

Step 1: Define ACLs

Group IP Traffic Types into Different Classes

- Pre-Undesirable—traffic that is deemed “bad” or “malicious” to be denied access to the RP
- Critical—traffic crucial to the operation of the network
- Important—traffic necessary for day-to-day operations
- Normal—traffic expected but not essential for network operations
- Post-Undesirable—traffic that is deemed “bad” or “malicious” to be denied access to the RP
- Catch-All—all other IP traffic destined to the RP that has not been identified
- Default—all remaining non-IP traffic destined to the RP that has not been identified

Step 1: Define ACLs

The Router IP Address for Control/Management Traffic Is 10.1.1.1

- Pre-Undesirable—ACL 120
- Critical—ACL 121
- Important—ACL 122
- Normal—ACL 123
- Post-Undesirable—ACL 124
- Catch All—ACL 125
- Default—no ACL required

```
! Pre-Undesirable – Traffic that should never touch the RP
access-list 120 permit tcp any any fragments
access-list 120 permit udp any any fragments
access-list 120 permit icmp any any fragments
access-list 120 permit ip any any fragments
access-list 120 permit udp any any eq 1434
```

Step 1: Define ACLs

The Router IP Address for Control/Management Traffic Is 10.1.1.1

- Pre-Undesirable—ACL 120
- Critical—ACL 121
- Important—ACL 122
- Normal—ACL 123
- Post-Undesirable—ACL 124
- Catch All—ACL 125
- Default—no ACL required

```
! CRITICAL -- Defined as routing protocols
access-list 121 permit tcp host 10.1.1.2 eq bgp host 10.1.1.1 gt 1024
access-list 121 permit tcp host 10.1.1.2 gt 1024 host 10.1.1.1 eq bgp
access-list 121 permit tcp host 10.1.1.3 eq bgp host 10.1.1.1 gt 1024
access-list 121 permit tcp host 10.1.1.3 gt 1024 host 10.1.1.1 eq bgp
access-list 121 permit ospf any any precedence internet
access-list 121 permit ospf any any precedence network
access-list 121 permit pim any host 224.0.0.13
```

Step 1: Define ACLs

The Router IP Address for Control/Management Traffic Is 10.1.1.1

- Pre-Undesirable—ACL 120
- Critical—ACL 121
- Important—ACL 122
- Normal—ACL 123
- Post-Undesirable—ACL 124
- Catch All—ACL 125
- Default—no ACL required

```
! IMPORTANT -- Defined as traffic required to manage the router
access-list 122 permit tcp 10.2.1.0 0.0.0.255 eq 22 host 10.1.1.1
established
access-list 122 permit tcp 10.2.1.0 0.0.0.255 host 10.1.1.1 eq 22
access-list 122 permit tcp 10.2.1.0 0.0.0.255 host 10.1.1.1 eq telnet
access-list 122 permit udp host 10.2.2.1 eq tftp host 10.1.1.1
access-list 122 permit udp host 10.2.2.2 host 10.1.1.1 eq snmp
access-list 122 permit udp host 10.2.2.3 host 10.1.1.1 eq ntp
```

Step 1: Define ACLs

The Router IP Address for Control/Management Traffic Is 10.1.1.1

- Pre-Undesirable—ACL 120
- Critical—ACL 121
- Important—ACL 122
- Normal—ACL 123
- Post-Undesirable—ACL 124
- Catch All—ACL 125
- Default—no ACL required

```
! NORMAL -- Defined as other traffic destined to the router to track and limit
access-list 123 permit icmp any any ttl-exceeded
access-list 123 permit icmp any any port-unreachable
access-list 123 permit icmp any any echo-reply
access-list 123 permit icmp any any echo
access-list 123 permit icmp any any packet-too-big
```

Step 1: Define ACLs

The Router IP Address for Control/Management Traffic Is 10.1.1.1

- Pre-Undesirable—ACL 120
- Critical—ACL 121
- Important—ACL 122
- Normal—ACL 123
- Post-Undesirable—ACL 124
- Catch All—ACL 125
- Default—no ACL required

```
! Post-Undesirable – Traffic that should never touch the RP
access-list 124 permit tcp any  host 10.1.1.1 eq 22
access-list 124 permit tcp any  host 10.1.1.1 eq telnet
access-list 124 permit tcp any host 10.1.1.1 eq bgp
access-list 124 permit udp any eq tftp host 10.1.1.1
access-list 124 permit udp any  host 10.1.1.1 eq snmp
access-list 124 permit udp any host 10.1.1.1 eq ntp
```

Step 1: Define ACLs

The Router IP Address for Control/Management Traffic Is 10.1.1.1

- Pre-Undesirable—ACL 120
- Critical—ACL 121
- Important—ACL 122
- Normal—ACL 123
- Post-Undesirable—ACL 124
- Catch All—ACL 125
- Default—no ACL required

```
! CATCH ALL -- Defined as other IP traffic destined to the router
access-list 125 permit ip any any
```

Step 2: Define Class-Maps

- Create class-maps to complete the traffic-classification process

  Use the access-lists defined on the previous slides to specify which IP packets belong in which classes

- Class-maps permit multiple match criteria, and nested class-maps

  match-any requires that packets meet only one "match" criteria to be considered "in the class"

  match-all requires that packets meet all of the "match" criteria to be considered "in the class"

- A "match-all" classification scheme with a simple, single-match criteria will satisfy initial deployments
- Traffic destined to the "undesirable" class should follow a "match-any" classification scheme

Step 2: Define Class-Maps

```
! Define a class for each "type" of traffic and associate the
! appropriate ACL
class-map match-any CoPP-pre-undesirable
  match access-group 120
class-map match-any CoPP-critical
  match access-group 121
class-map match-any CoPP-important
  match access-group 122
class-map match-any CoPP-normal
  match access-group 123
class-map match-any CoPP-post-undesirable
  match access-group 124
class-map match-any CoPP-catch-all
  match access-group 125
```

Step 3: Define Policy-Map

- Class-maps defined in Step 2 need to be "enforced" by using a policy-map to specify appropriate service policies for each traffic class
- For example:

  For undesirable traffic types, all actions are unconditionally "drop" regardless of rate

  For critical, important, and normal traffic types, all actions are "transmit" to start out

  For catch-all traffic, rate-limit the amount of traffic permitted above a certain bps

  Note: all traffic that fails to meet the matching criteria belongs to the default traffic class, which is user configurable, but cannot be deleted

Step 3: Define Policy-Map

```
! Example "Baseline" service policy for each traffic classification
policy-map CoPP
 class CoPP-pre-undesirable
    police 8000 1000 4470 conform-action drop  exceed-action drop
  class CoPP-critical
    police 5000000 2500 4470 conform-action transmit  exceed-action transmit
class CoPP-important
    police 1000000 1000 4470 conform-action transmit  exceed-action transmit
class CoPP-normal
    police 1000000 1000 4470 conform-action transmit  exceed-action drop
class CoPP-post-undesirable
    police 8000 1000 4470 conform-action drop  exceed-action drop
class CoPP-catch-all
    police 1000000 1000 4470 conform-action transmit  exceed-action drop
class class-default
    police 8000 1000 4470 conform-action transmit exceed-action transmit
```

Step 4: Apply Policy to “Interface”

- Apply the policy-map created in Step 3 to the “control plane”
- The new global configuration CLI “control-plane” command is used to enter “control-plane configuration mode”
- Once in control-plane configuration mode, attach the service policy to the control plane in the “input” direction

  Input—applies the specified service policy to packets that are entering the control plane

Step 4: Apply Policy to “Interface”

- Centralized

```
Router(config)# control-plane
Router(config-cp)# service-policy [input | output] <policy-map-name>
```

- Distributed

```
Router(config)#control-plane slot <n>
Router(config-cp)#service-policy input <policy-map-name>
```

```
! Example
! This applies the policy-map to the Control Plane control-plane
  service-policy input CoPP
```

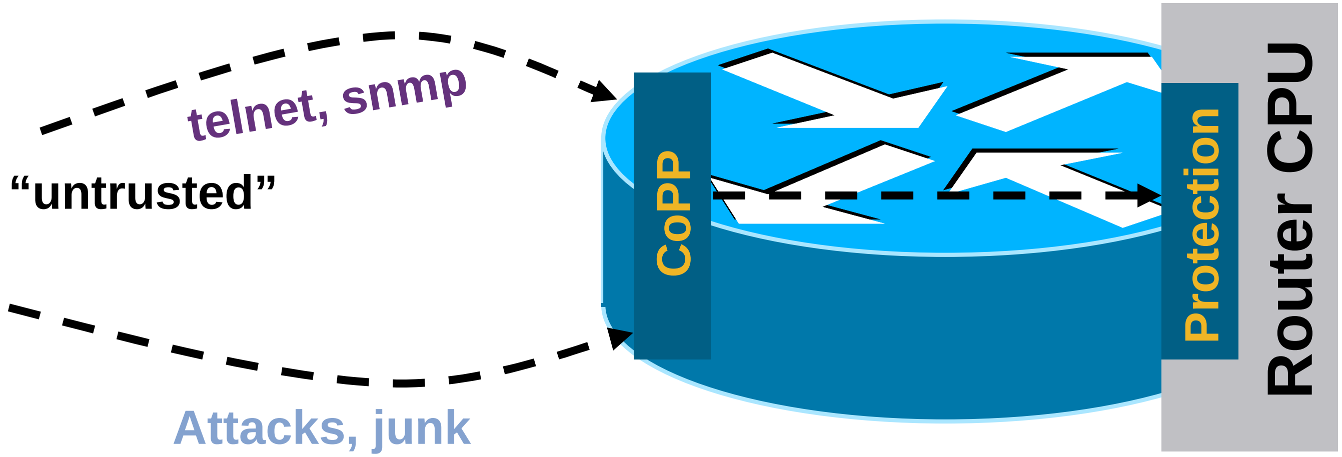
Monitoring CoPP

- "show access-list" displays hit counts on a per ACL entry (ACE) basis
  - The presence of hits indicates flows for that data type to the control plane as expected
  - Large numbers of packets or an unusually rapid rate increase in packets processed may be suspicious and should be investigated
  - Lack of packets may also indicate unusual behavior or that a rule may need to be rewritten
- "show policy-map control-plane" is invaluable for reviewing and tuning site-specific policies and troubleshooting CoPP
  - Displays dynamic information about number of packets (and bytes) conforming or exceeding each policy definition
  - Useful for ensuring that appropriate traffic types and rates are reaching the route processor
- Use SNMP queries to automate the process of reviewing service-policy transmit and drop rates
  - The Cisco QoS MIB (CISCO-CLASS-BASED-QOS-MIB) provides the primary mechanisms for MQC-based policy monitoring via SNMP

Show Policy-Map Command

```
Router#show policy-map control-plane input
 Control Plane
  Service-policy input: CoPP
    Class-map: CoPP-critical (match-all)
      16 packets, 2138 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group 121
      police:
          cir 5000000 bps, bc 2500 bytes
        conformed 16 packets, 2138 bytes; actions:
          transmit
        exceeded 0 packets, 0 bytes; actions:
          transmit
        conformed 0 bps, exceed 0 bps
…
    Class-map: class-default (match-any)
      250 packets, 84250 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      police:
          cir 8000 bps, bc 1000 bytes
        conformed 41 packets, 5232 bytes; actions:
          transmit
        exceeded 0 packets, 0 bytes; actions:
          transmit
        conformed 0 bps, exceed 0 bps
Router#
```

Control Plane Policing



- Superset of rACL: Start planning your migrations
- Provides a cross-platform methodology for protecting the control plane
  - Consistent "show" command and MIB support
- Granular: Permit, deny and rate-limit
- Platform specifics details: Centralized vs. distributed vs. hardware

Agenda

- Infrastructure Protection Overview
- Understanding Routers and Planes
- Infrastructure Protection from the Inside Out
    - Router Hardening: Traditional Methods
    - Router Hardening: Protecting the CPU
    - Network Hardening

Network Hardening

Network Hardening

- In the context of denial of service if the packet makes it to the router its already too late

  CoPP and rACL help dramatically, but they do not solve the problem

  The unwanted packets must be dropped on ingress into your network

- Three methods:

  Infrastructure ACL

  Core Hiding

  RFC2547 (MPLS) VPN

Network Hardening: Infrastructure ACL

Infrastructure ACLs

- Basic premise: filter traffic destined to your core routers

    Do your core routers really need to process all kinds of garbage?

- Develop list of required protocols that are sourced from outside your AS and access core routers

    Example: eBGP peering, GRE, IPSec, etc.

    Use classification ACL as required

- Identify core address block(s)

    This is the protected address space

    Summarization is critical → simpler and shorter ACLs

    Poor summarization may make iACLs unwieldy

Infrastructure ACLs

- Infrastructure ACL will permit only required protocols and deny all others to infrastructure space
- ACL should also provide anti-spoof filtering

    Deny your space from external sources

    Deny RFC1918 space

    Deny multicast sources addresses (224/4)

    RFC3330 defines special use IPv4 addressing

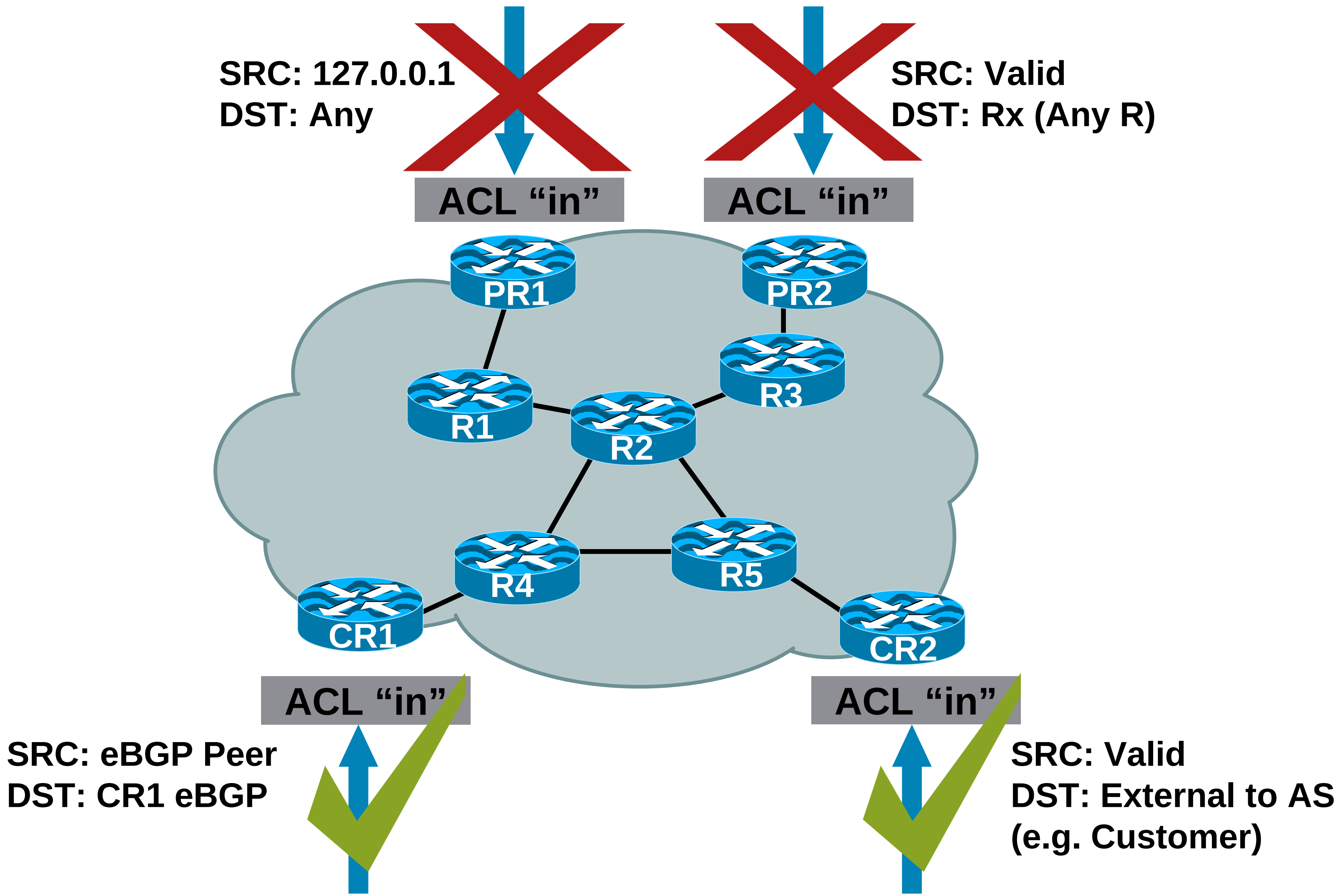
Infrastructure ACLs

- Infrastructure ACL must permit transit traffic

  Traffic passing through routers must be allowed via permit IP any any

- ACL is applied inbound on ingress interfaces
- Fragments destined to the core can be filtered via fragments keyword

Infrastructure ACL in Action



Iterative Deployment

- Typically a very limited subset of protocols needs access to infrastructure equipment
- Even fewer are sourced from outside your AS
- Identify required protocols via classification ACL
- Deploy and test your ACLs

Step 1: Classification

- Traffic destined to the core must be classified
- NetFlow can be used to classify traffic
  - Need to export and review
- Classification ACL can be used to identify required protocols
  - Series of permit statements that provide insight into required protocols
  - Log keyword can be used for additional detail; hits to ACL entry with log will increase CPU utilization: impact varies by platform
- Regardless of method, unexpected results should be carefully analyzed → do not permit protocols that you can’t explain

Step 2: Begin to Filter

- Permit protocols identified in Step 1 to infrastructure address blocks
- Deny all others to infrastructure address blocks

  Watch access control entry (ACE) counters

  Log keyword can help identify protocols that have been denied but are needed
- Last line: permit ip any any ← permit transit traffic
- The ACL now provides basic protection and can be used to ensure that the correct suite of protocols has been permitted

Steps 3 and 4: Restrict Source Addresses

- Step 3:

  ACL is providing basic protection

  Required protocols permitted, all other denied

  Identify source addresses and permit only those sources for requires protocols

  E.g., external BGP peers, tunnel end points

- Step 4:

  Increase security: deploy destination address filters to individual hosts if possible

Example: Simplistic Infrastructure ACL

```
! Deny our internal space as a source of external packets
access-list 101 deny ip our_CIDR_block any
! Deny src addresses of 0.0.0.0 and 127/8
access-list 101 deny ip host 0.0.0.0 any
access-list 101 deny ip 127.0.0.0 0.255.255.255 any
! Deny RFC1918 space from entering AS
access-list 101 deny ip 10.0.0.0 0.255.255.255 any
access-list 101 deny ip 172.16.0.0 0.0.15.255 any
access-list 101 deny ip 192.168.0.0 0.0.255.255 any
!Permit eBGP from outside out network
access-list 101 permit tcp host peerA host peerB eq 179
access-list 101 permit tcp host peerA eq 179 host peerB
! Deny all other access to infrastructure
access-list 101 deny ip any core_CIDR_block
! Permit all data plane traffic
access-list 101 permit ip any any
```

Infrastructure ACL Summary

- Infrastructure ACLs are very effective at protecting the network if properly and universally deployed
- Have been successfully used by many SPs for years
- IP Address summarization critical to successful deployment
- Infrastructure ACLs also have a few weaknesses

    Hardware restrictions associated with deploying ACLs or the ACEs required in iACL may prevent deployment

    Operational overhead in maintaining and deploying iACL

    Collisions with customer ACLs difficult to manage

Q and A

Interesting Links

- iACL Deployment Guide

  http://www.cisco.com/warp/public/707/iacl.html

- rACL Deployment Guide

  http://www.cisco.com/warp/public/707/racl.html

- CoPP Deployment Guide

  http://www.cisco.com/en/US/products/ps6642/products_white_paper0900aecd804fa16a.shtml

- Cisco Network Foundation Protection (NFP)

  http://www.cisco.com/warp/public/732/Tech/security/infrastructure/

- SP Security Archive

  ftp://ftp-eng.cisco.com/cons/isp/security/

- NANOG

  http://www.nanog.org/previous.html

  http://www.nanog.org/ispsecurity.html