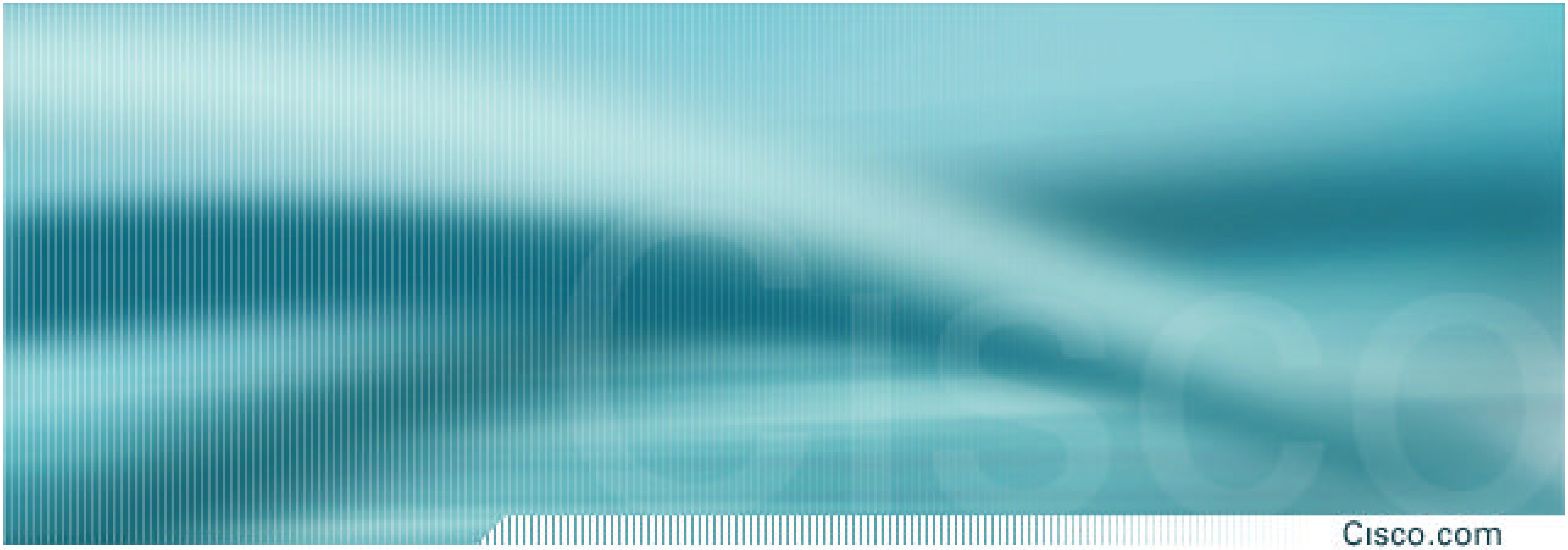


Router Architecture And IOS Internals

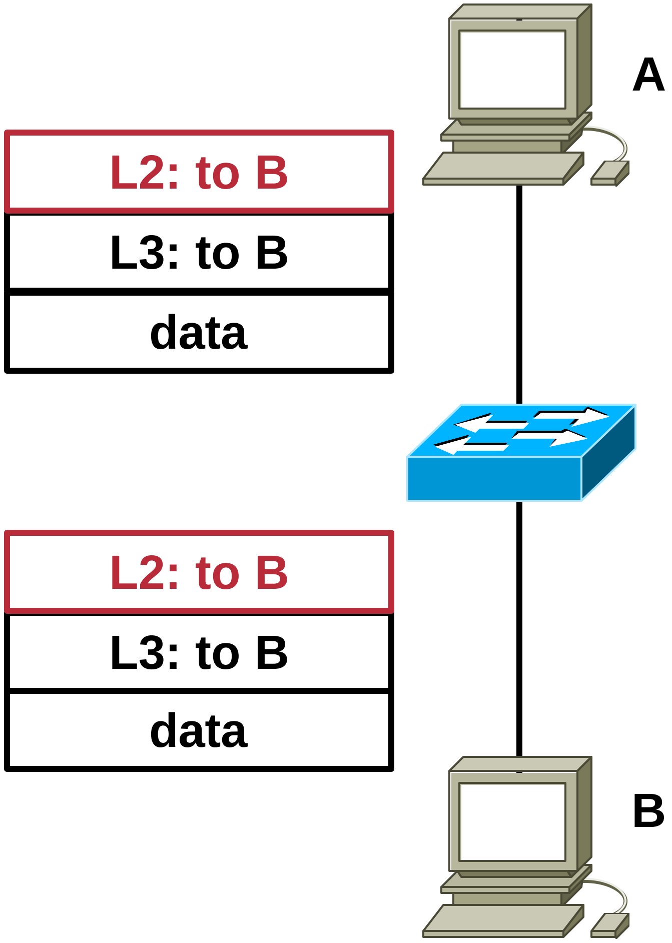
Agenda

- Routing and Switching
- Cisco IOS Switching Paths
- Cisco Express Forwarding
- Router Architectures & Parallel Express Forwarding

Routing and Switching

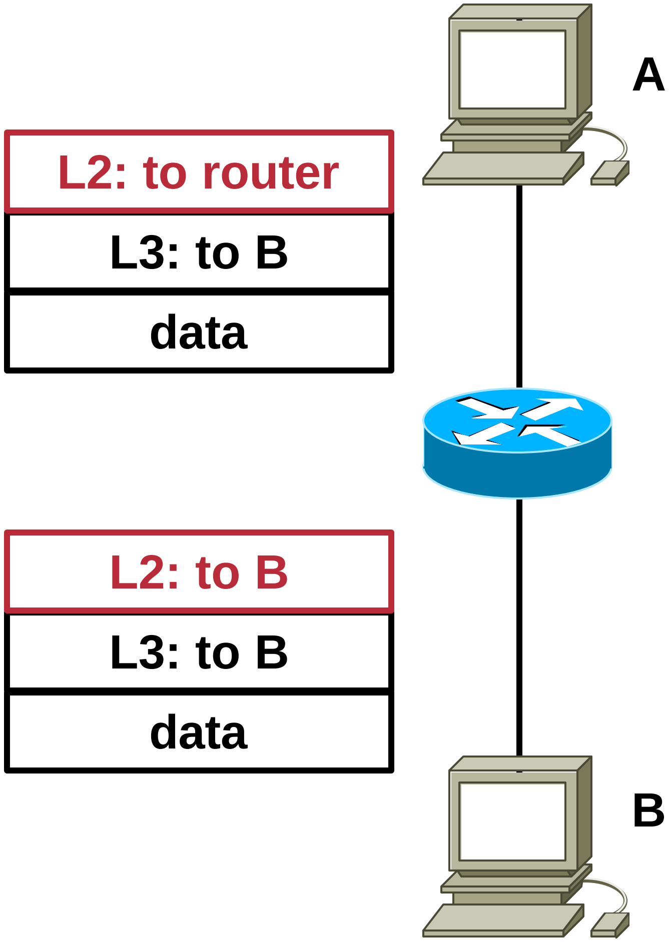
Switching

- **The destination in the layer two header remains the same when a packet passes through a switch**



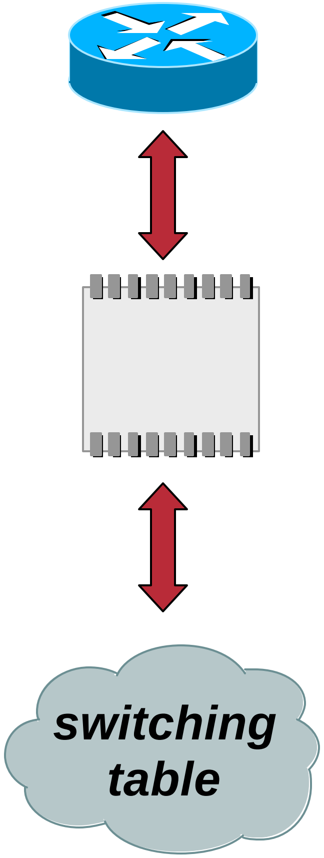
Routing

- **Host A transmits the packet to the router**
- **The router determines the correct outbound port, then rewrites the layer 2 header so the packet is now destined to B**



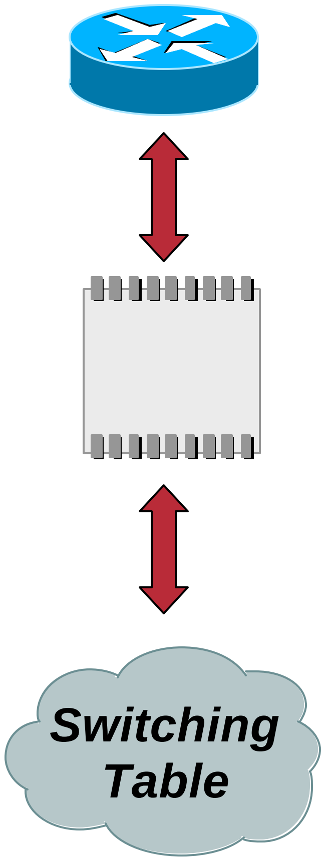
Routing

- **Switching, in the context of routers, involves this process of looking up the next hop, finding the layer 2 rewrite “string,” rewriting the layer 2 header, and transmitting the packet**

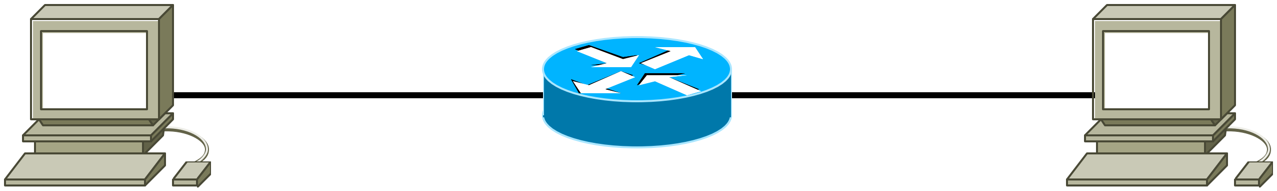


Layer 3 Switching

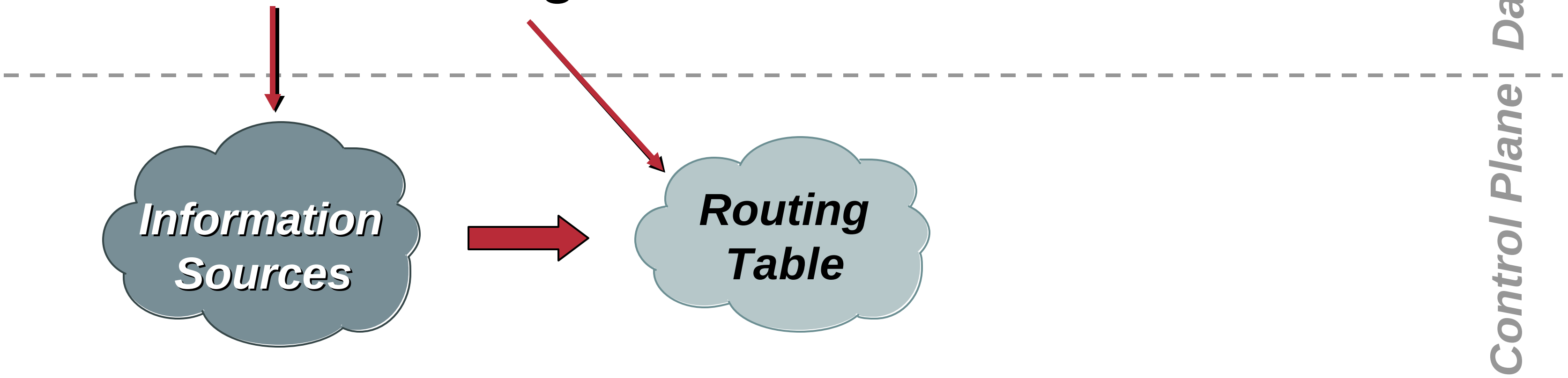
- When the term “layer 3 switch” was first coined, it meant switching packets in hardware based on the layer 3 information
- However, the lines are rarely so neatly drawn in the real world



Layer 3 Switching

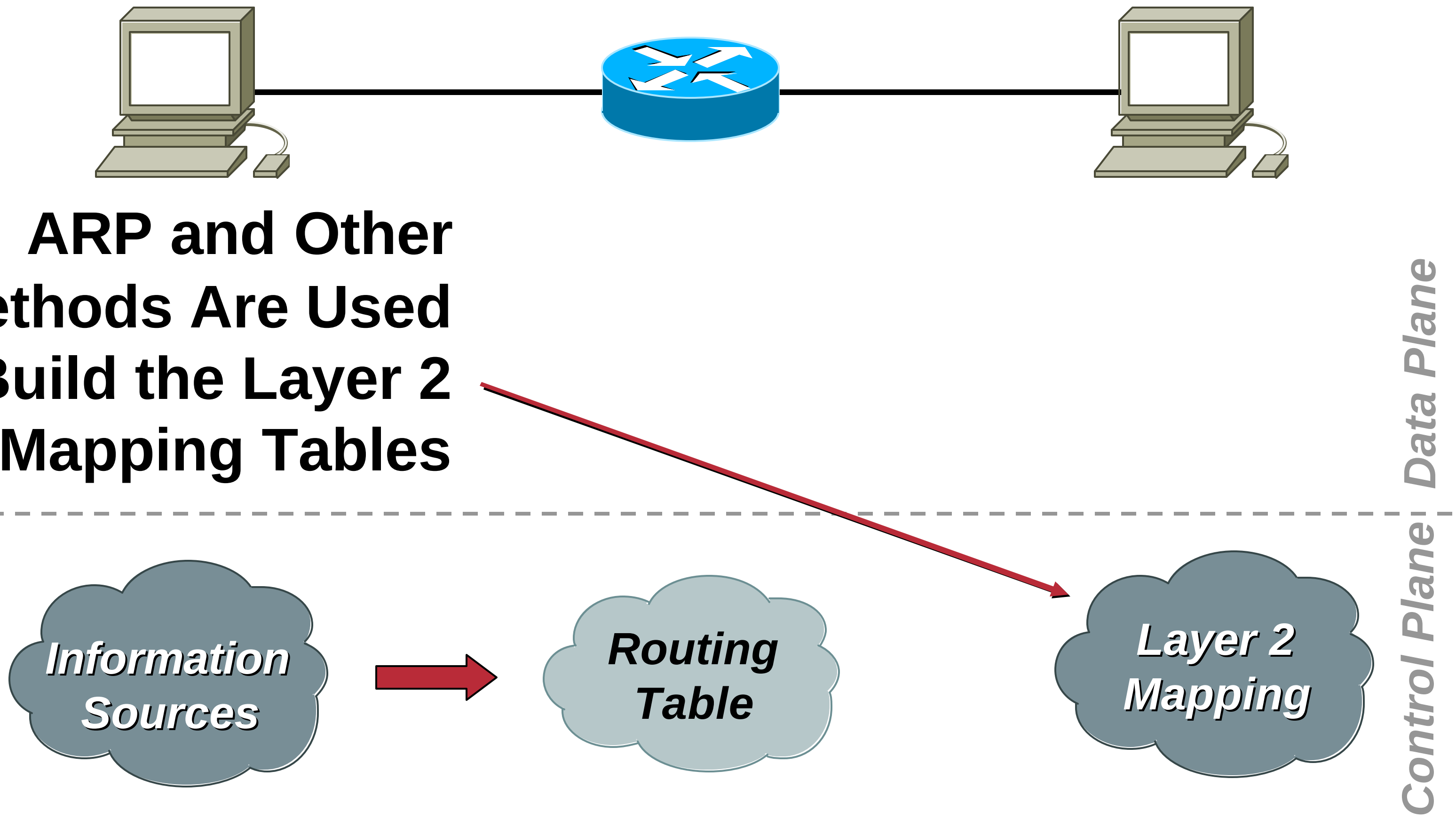


Routing Protocols and Other Sources Are Used to Build the Routing Table


Information Sources

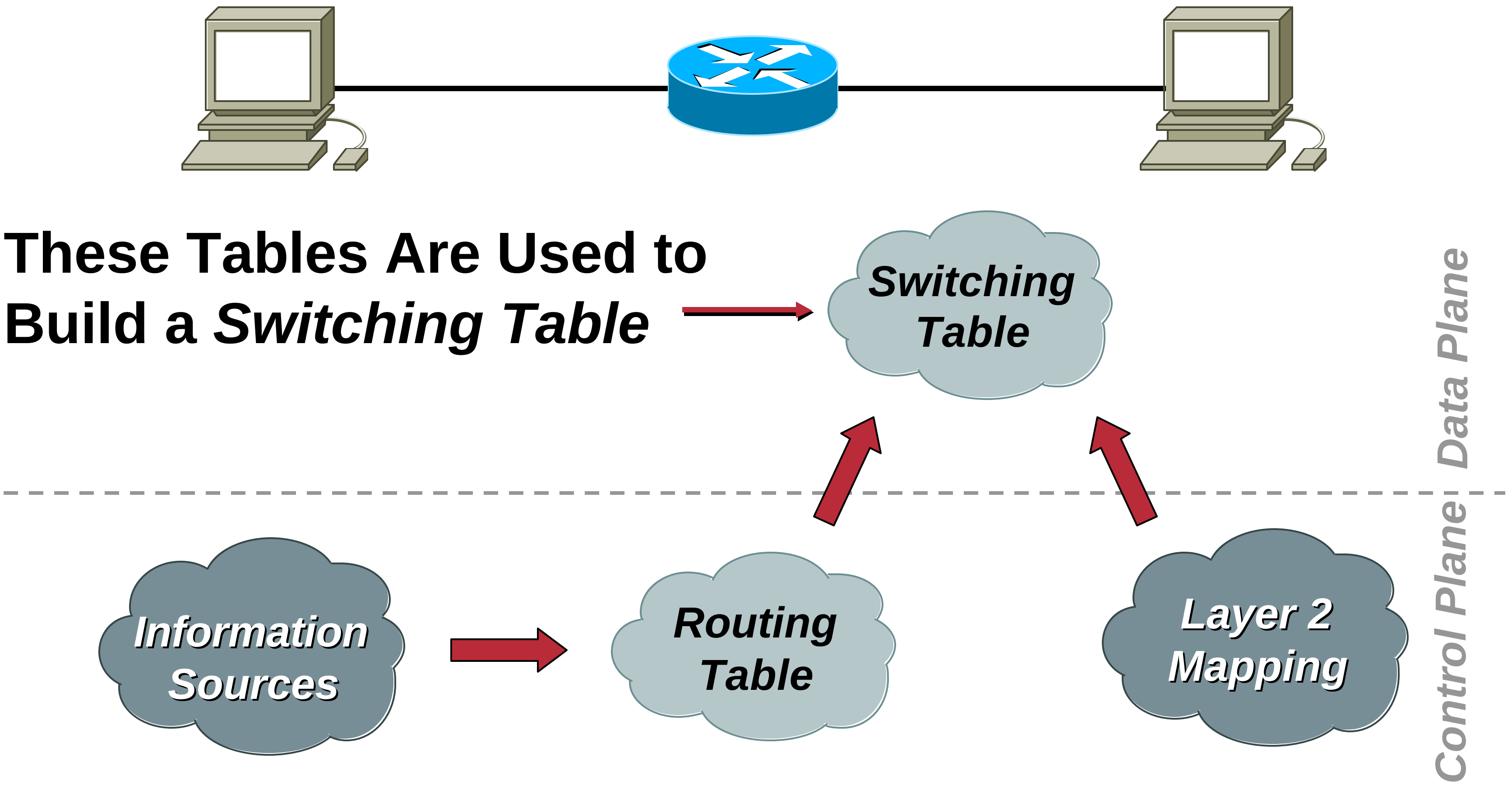
Routing Table

Layer 3 Switching



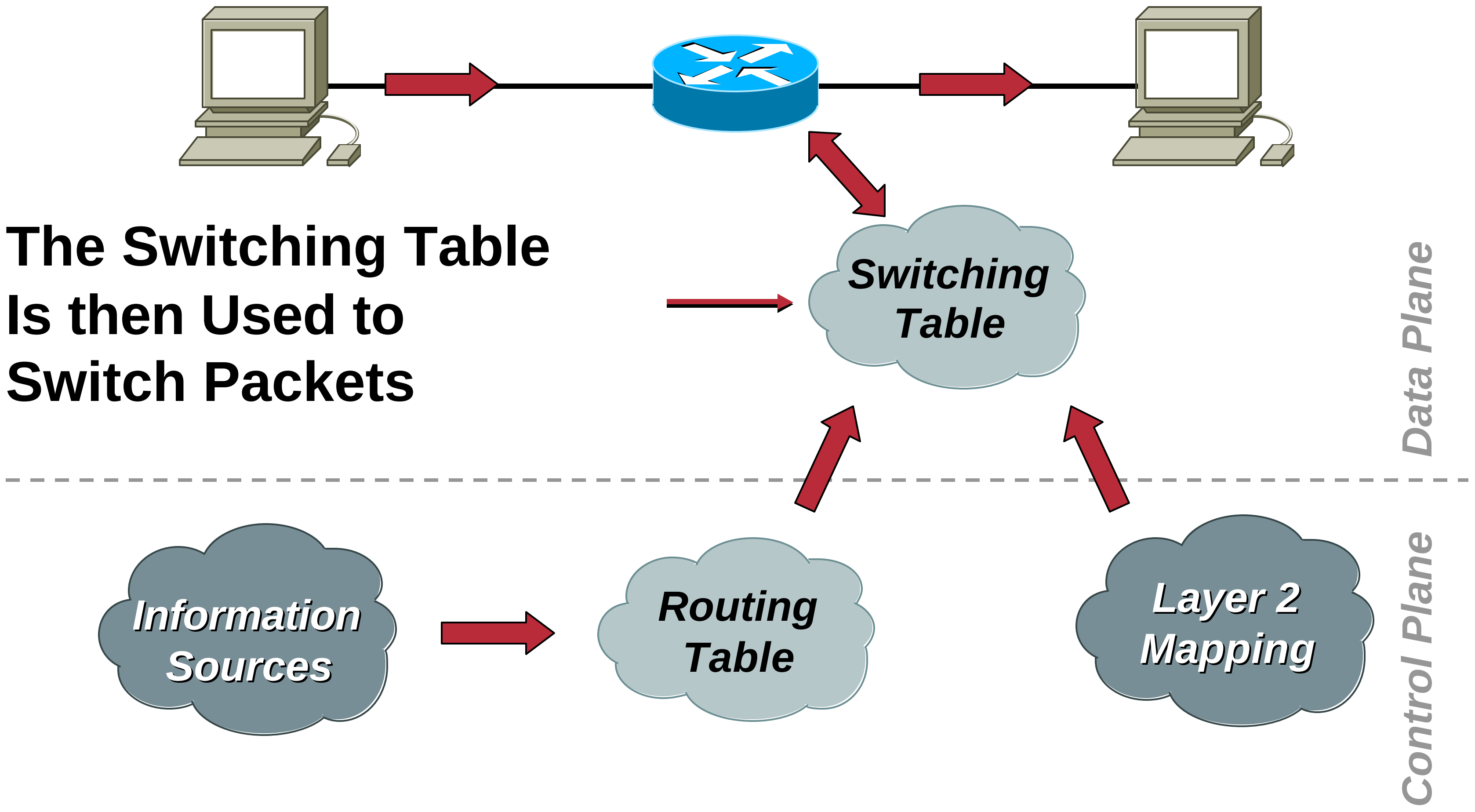
ARP and Other Methods Are Used to Build the Layer 2 Mapping Tables

Layer 3 Switching



These Tables Are Used to Build a *Switching Table*

Layer 3 Switching



The Switching Table Is then Used to Switch Packets

Layer 3 Switching

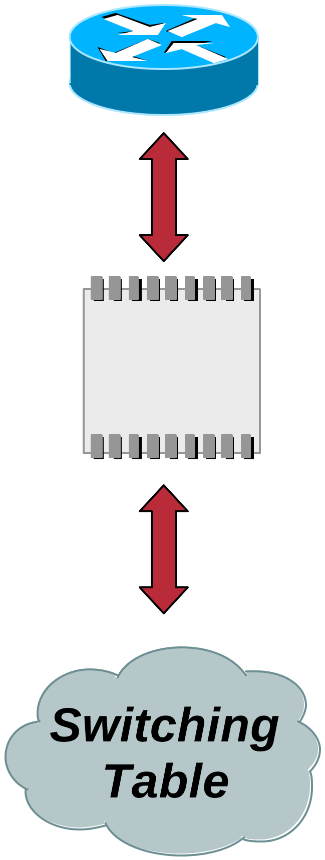
- Where is switching done?

  On the main processor, in a "normal" process

  On the main processor in a special mode (interrupt context)

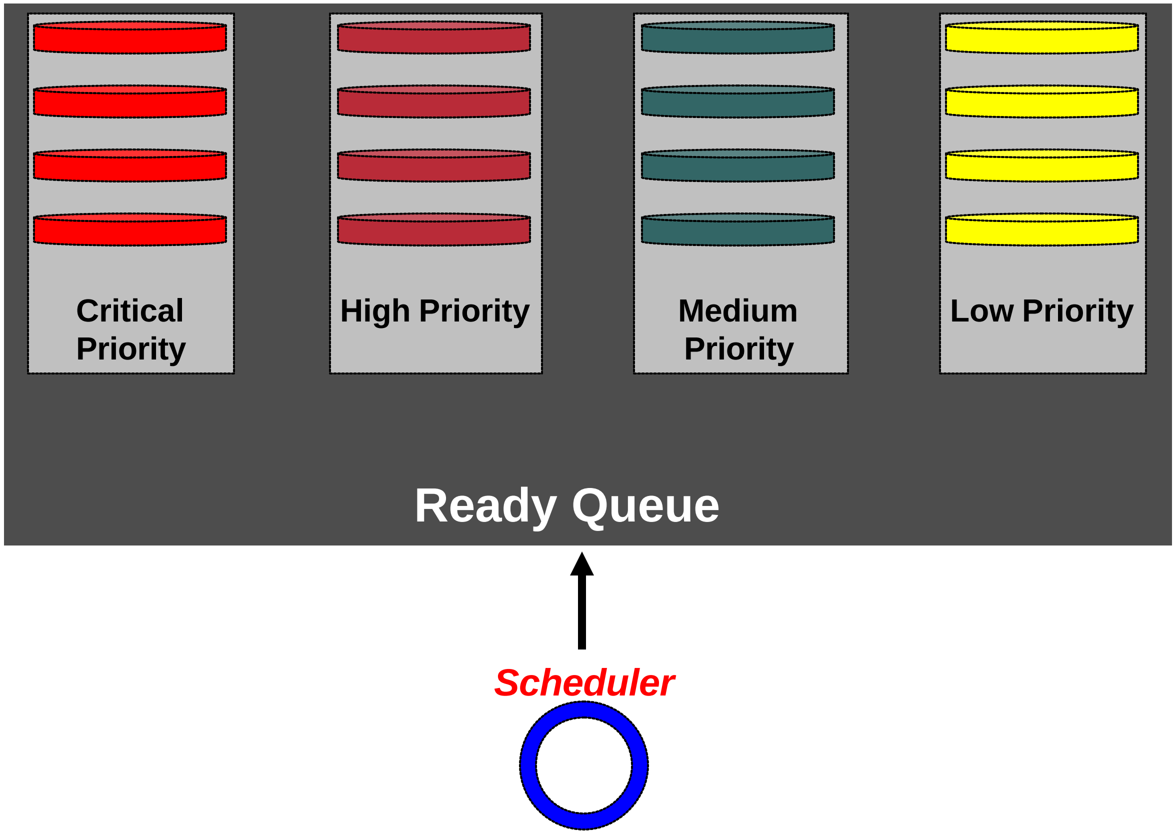
  On a separate general purpose processor

  On an application specific chip (ASIC)



Cisco IOS Switching Paths

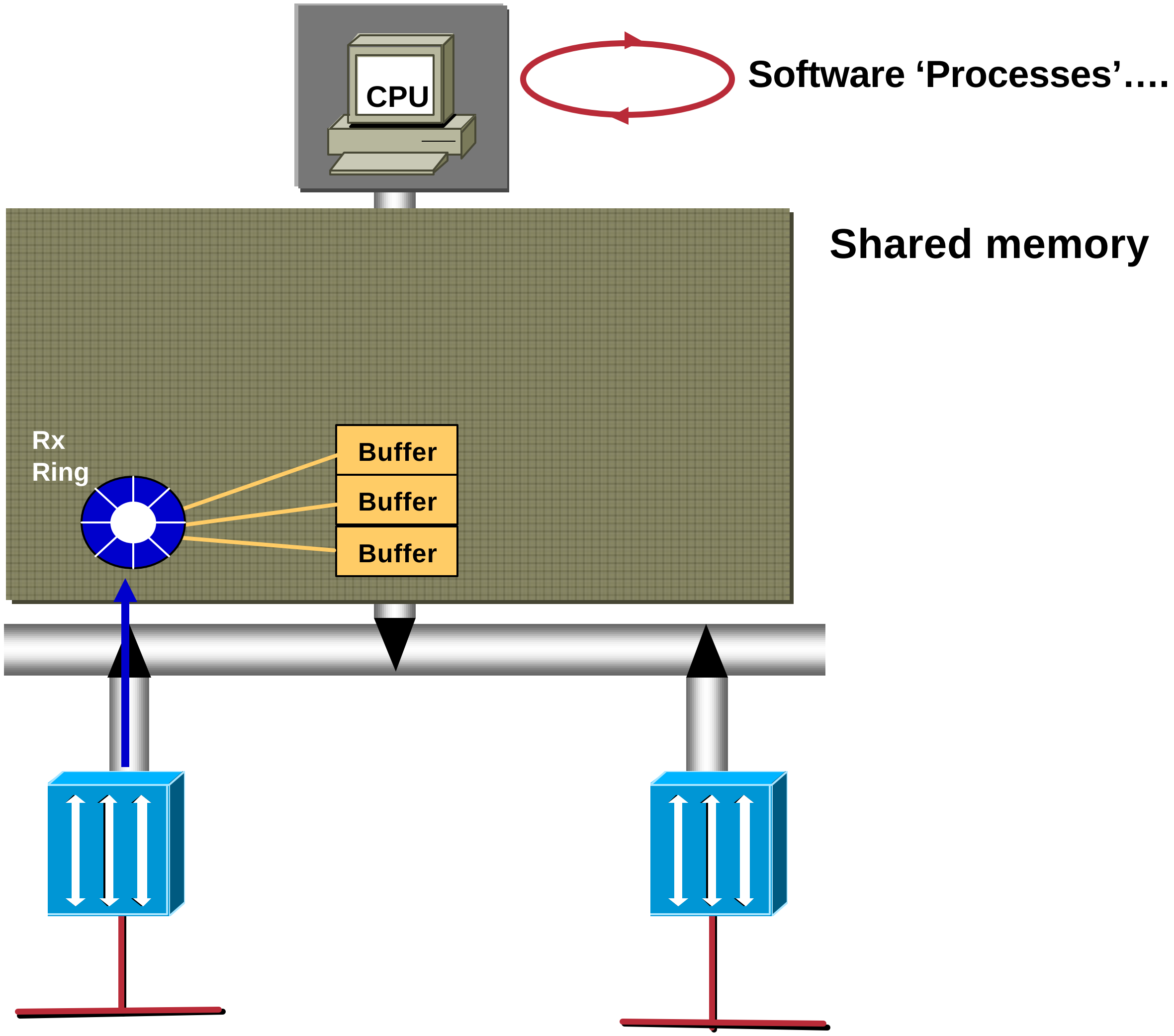
IOS Process Scheduling



- •Each disk represents a *Process* in the *Process Ready Queue.*

- •Each *Process* is assigned a *Priority* *(Critical, High, Medium or Low)*

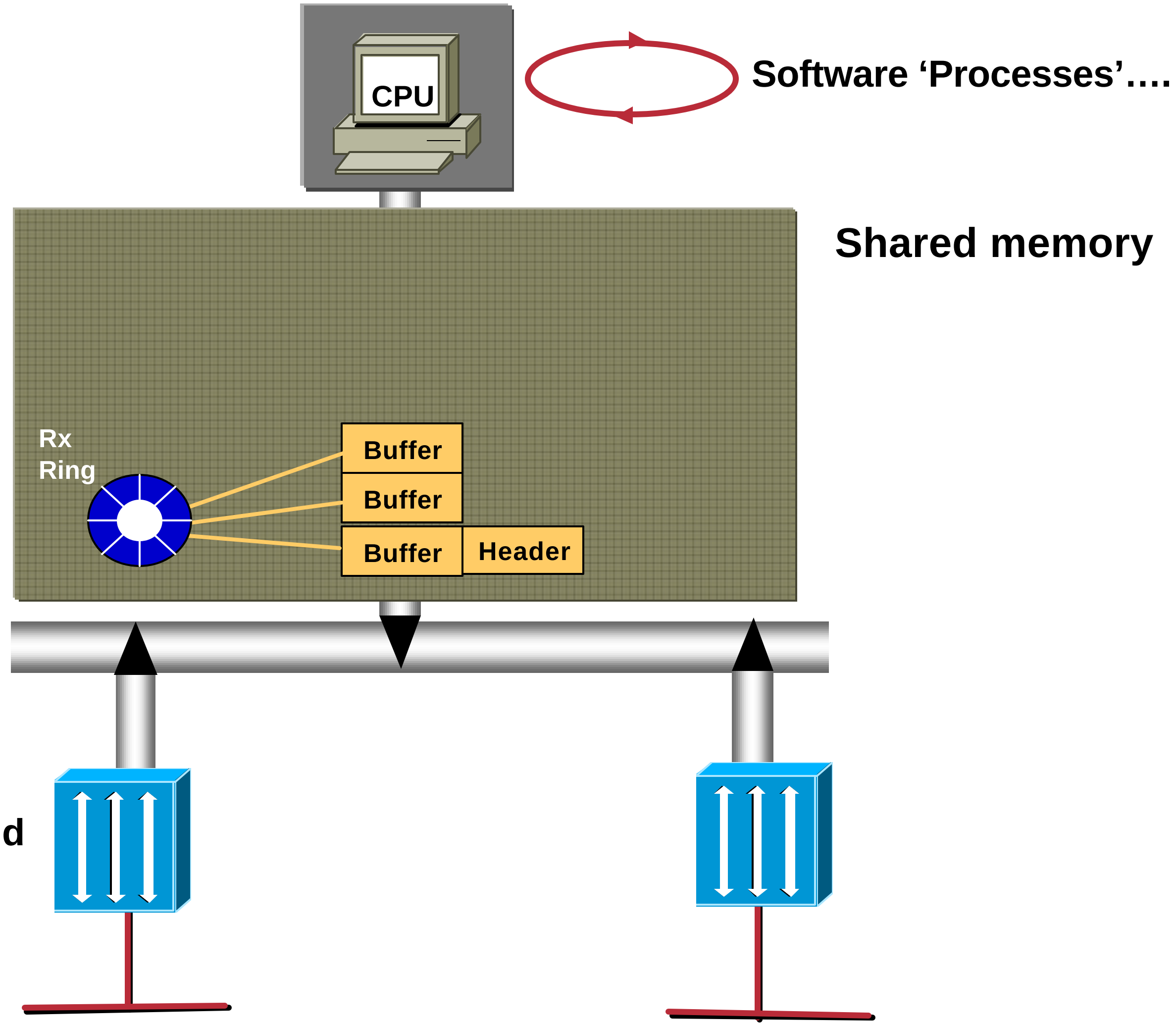
Router Switching Operation
“Process Switching”



1

Interface Processor DMA’s packet into RX Ring Buffer

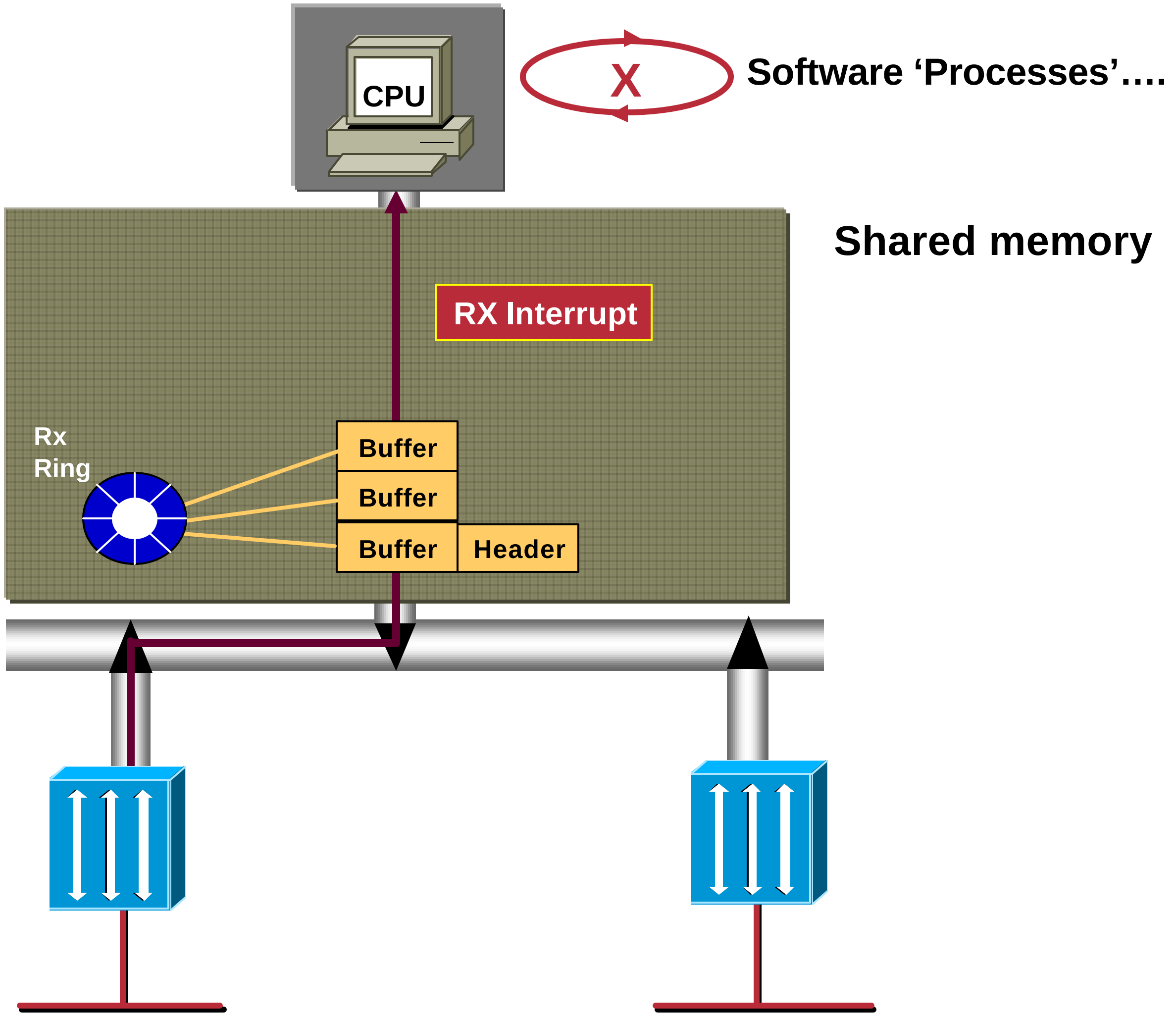
Router Switching Operation
"Process Switching"



2

Interface Driver Code Decodes packet header and builds Buffer Header with L3 Info

Router Switching Operation
“Process Switching”



3

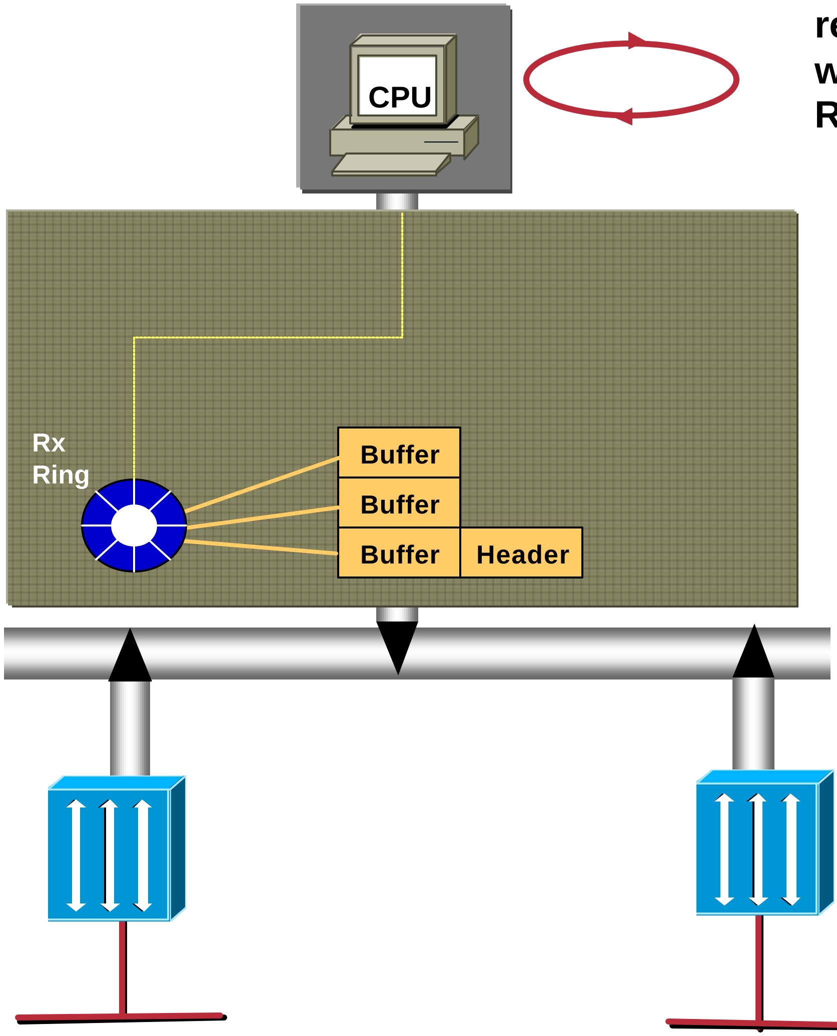
Interface processor generates RX Interrupt to CPU.

Router Switching Operation
“Process Switching”

4

When Packet passed to Processor, Buffer ownership transferred to Processor.

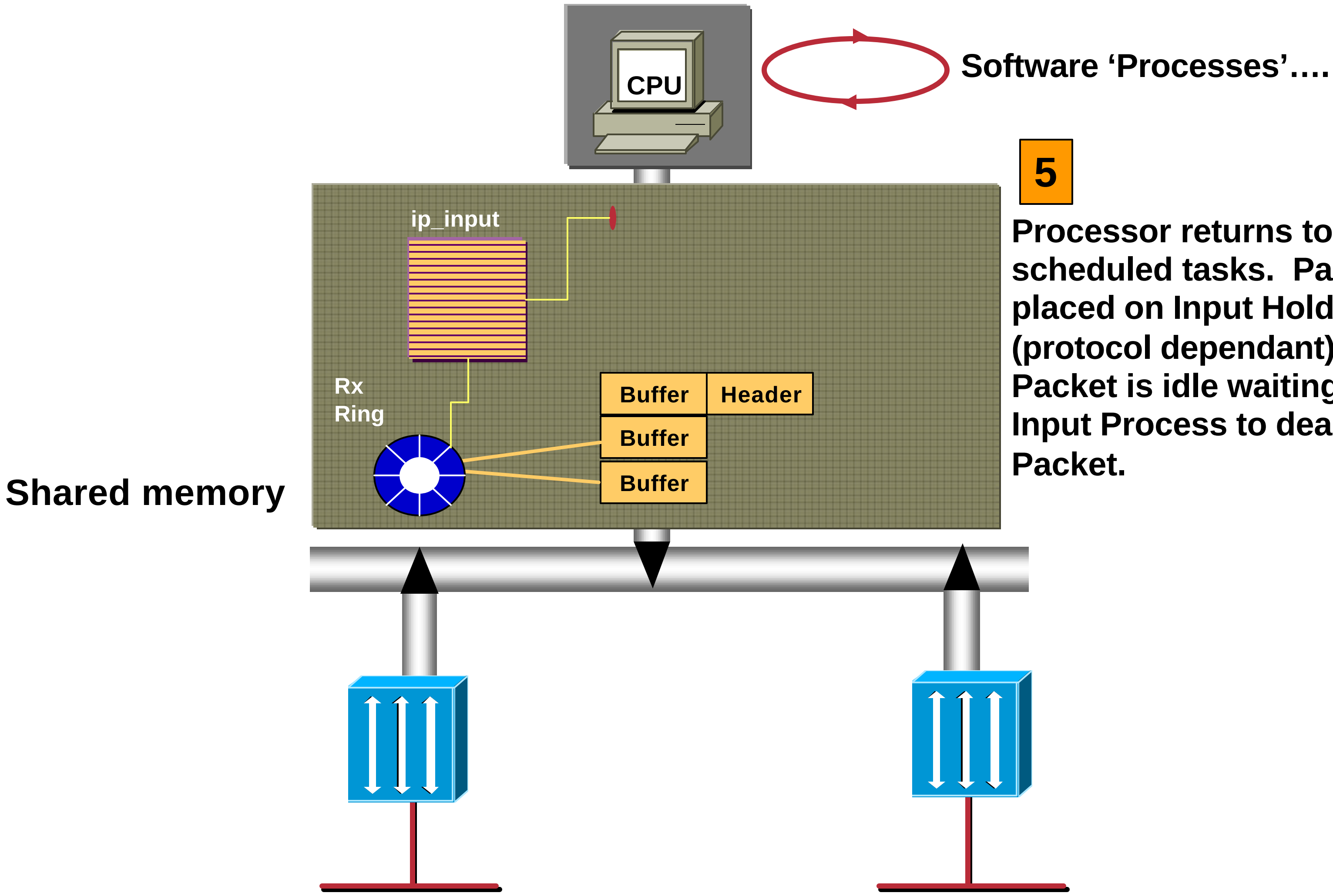
As Ownership has passed Interrupt released.



Software ‘Processes’ are resumed at the point the were suspended when the RX Interrupt arrived

Shared memory

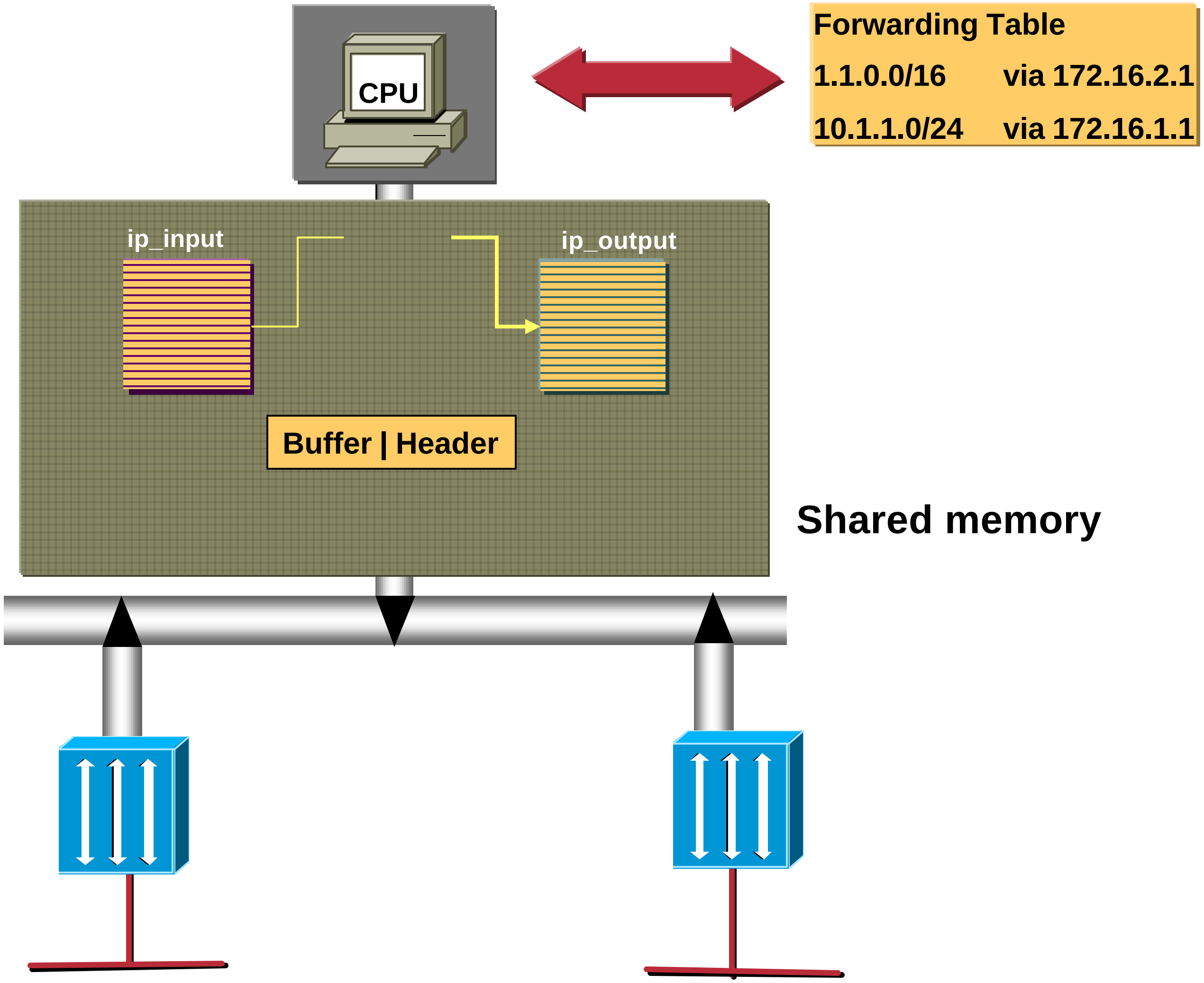
Router Switching Operation
“Process Switching”



5

Processor returns to scheduled tasks. Packet is placed on Input Hold Q (protocol dependant). Packet is idle waiting for Input Process to deal with Packet.

Router Switching Operation
“Process Switching”



Forwarding Table

1.1.0.0/16	via 172.16.2.1
10.1.1.0/24	via 172.16.1.1

CPU

ip_input

ip_output

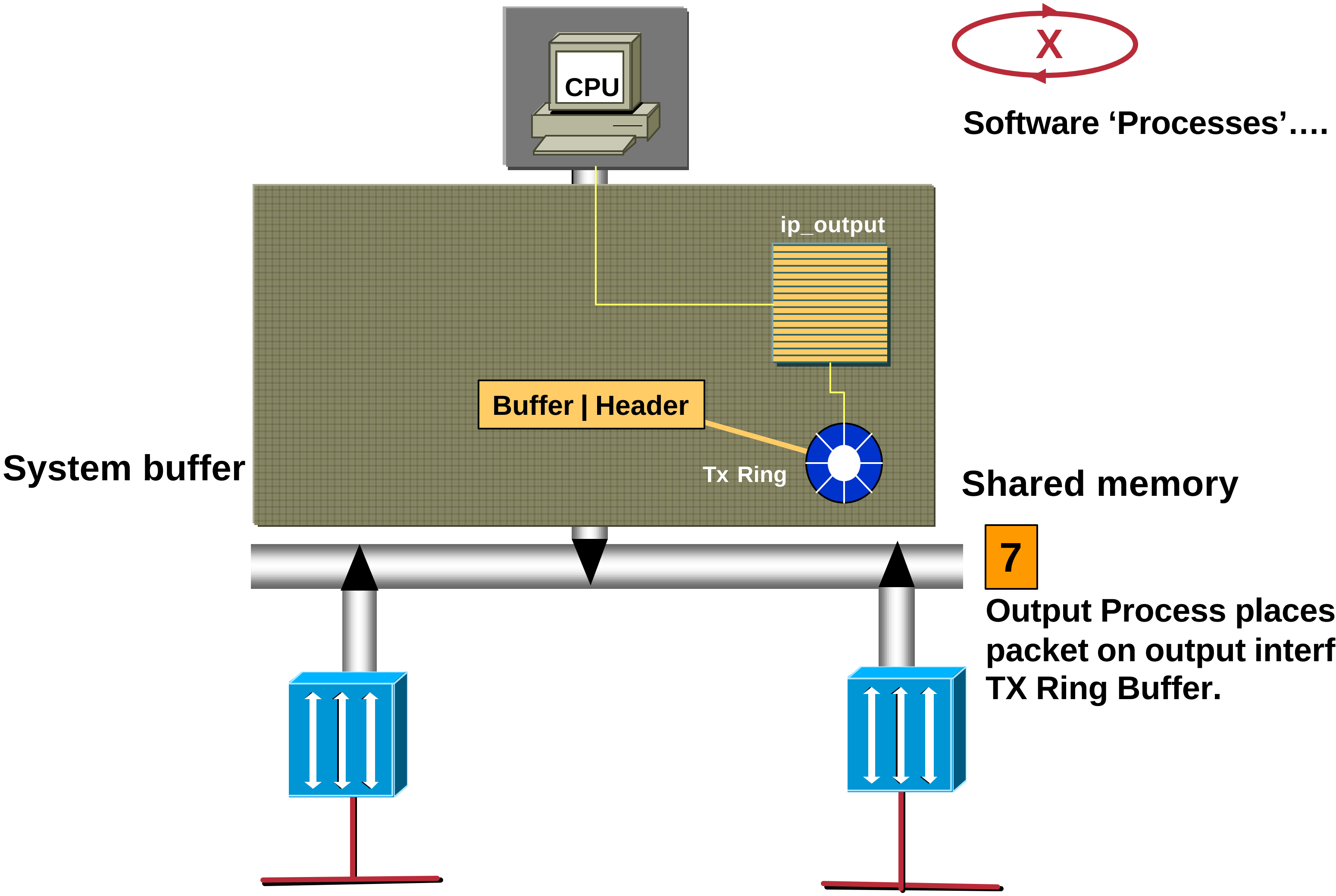
Buffer | Header

Shared memory

6

Input Process Looks up Destination in Forwarding Table. Determines O/P interface. Writes new MAC header. Places Packet in Output Q

Router Switching Operation "Process Switching"



Software 'Processes'….

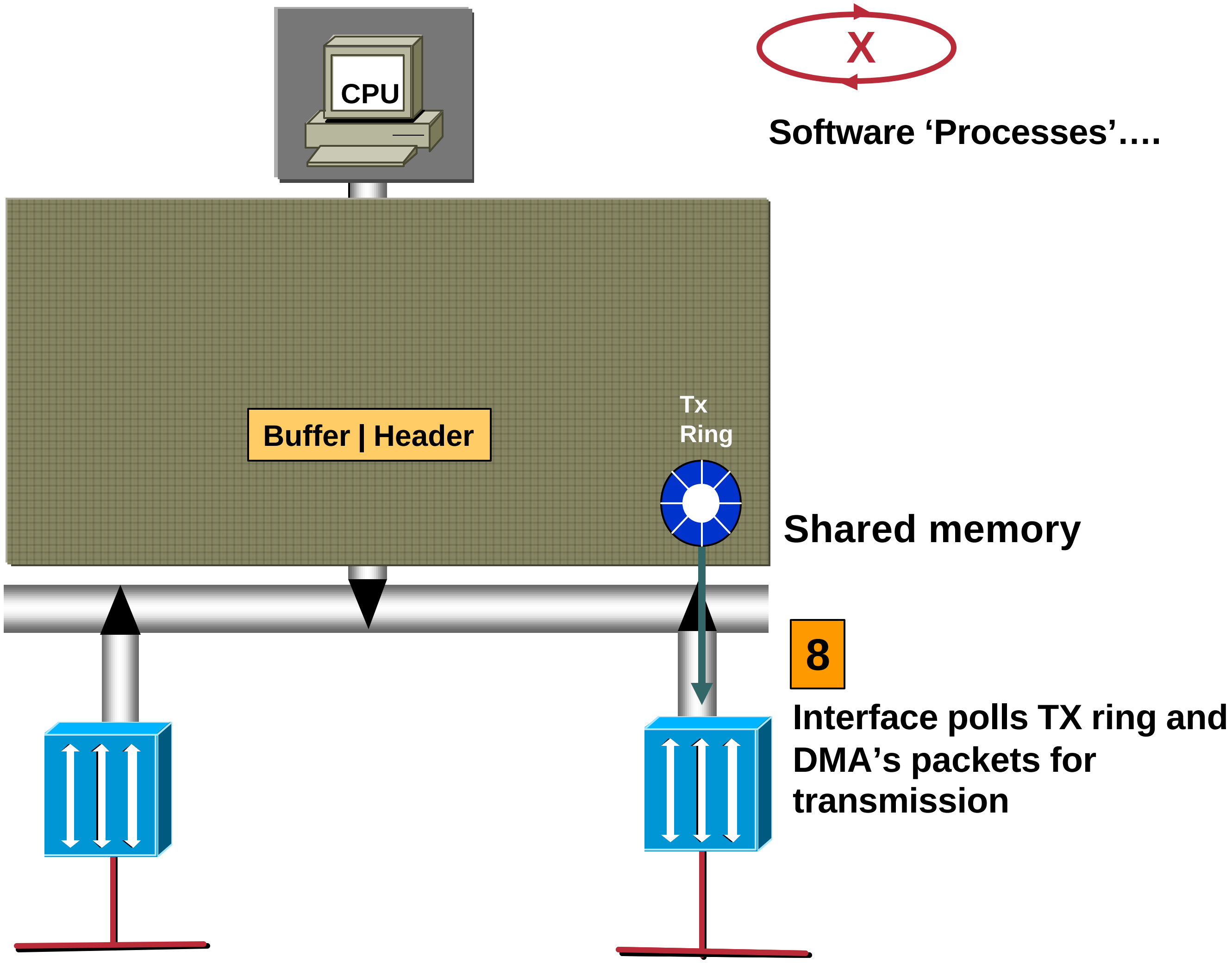
System buffer

Shared memory

7

Output Process places packet on output interface TX Ring Buffer.

Router Switching Operation
“Process Switching”



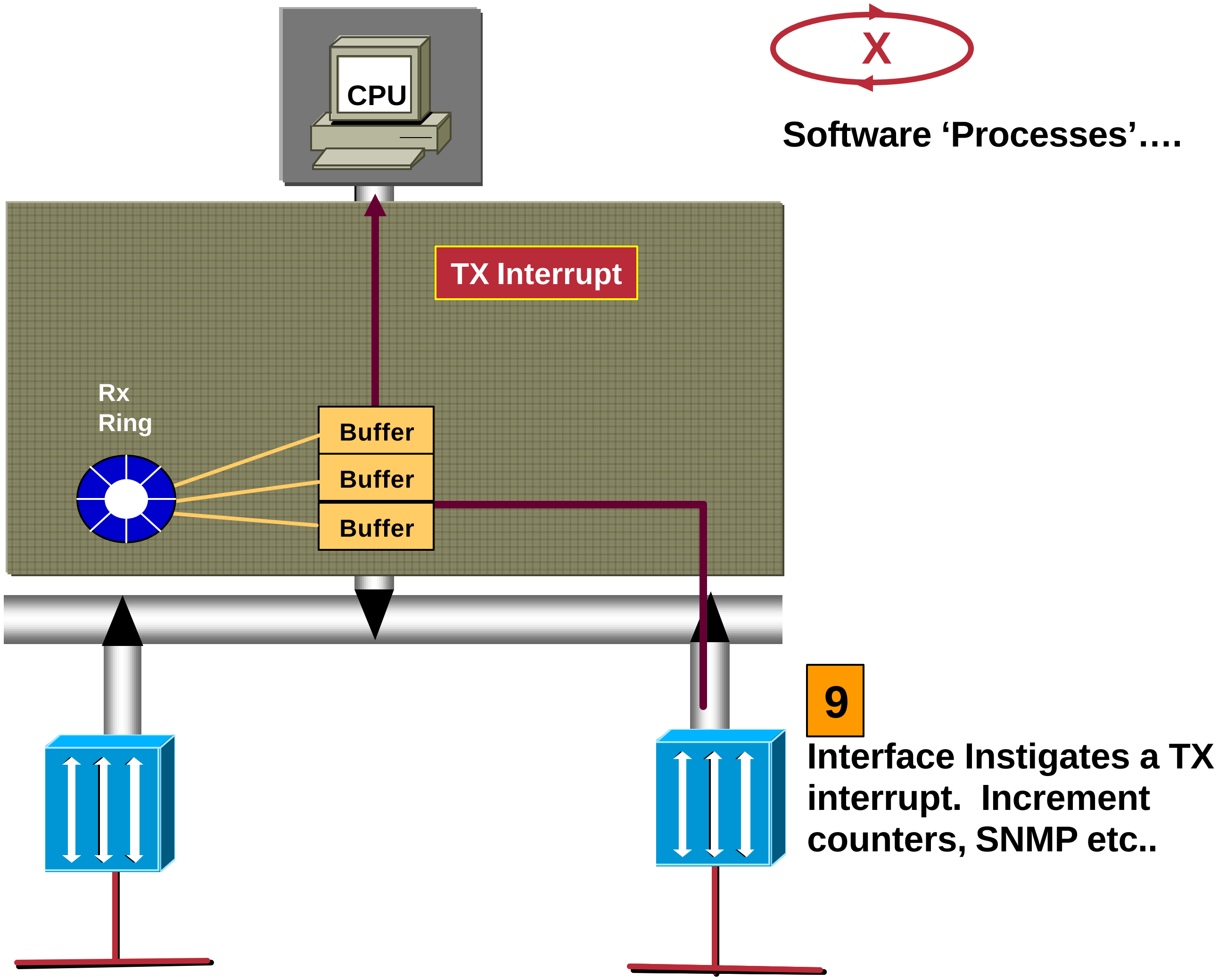
Software ‘Processes’….

Shared memory

8

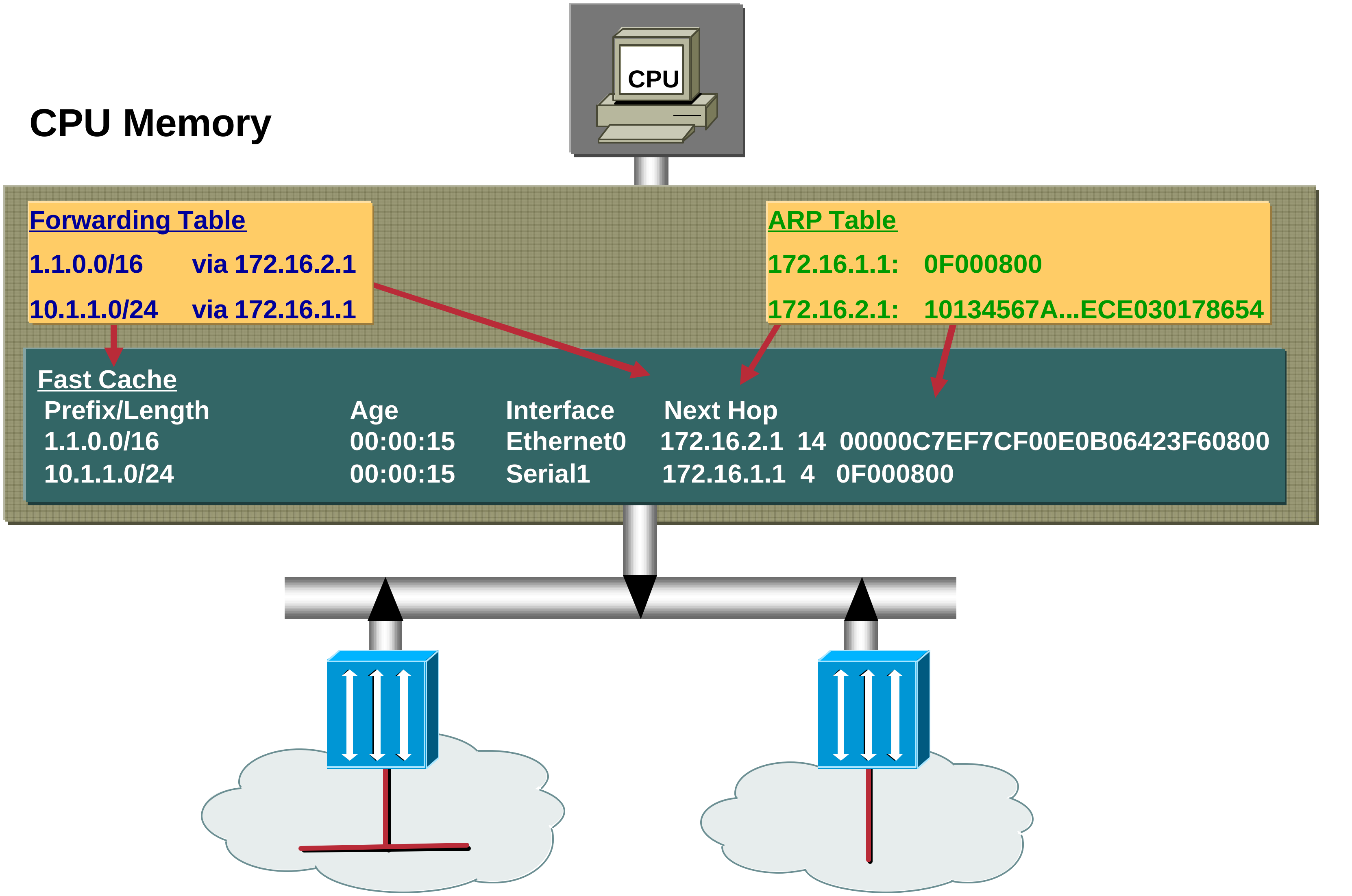
Interface polls TX ring and DMA’s packets for transmission

Router Switching Operation "Process Switching”



Interface Instigates a TX interrupt. Increment counters, SNMP etc..

Demand Generated Cache Based Switching ("Fast" Switching)



CPU Memory

Forwarding Table

1.1.0.0/16	via 172.16.2.1
10.1.1.0/24	via 172.16.1.1

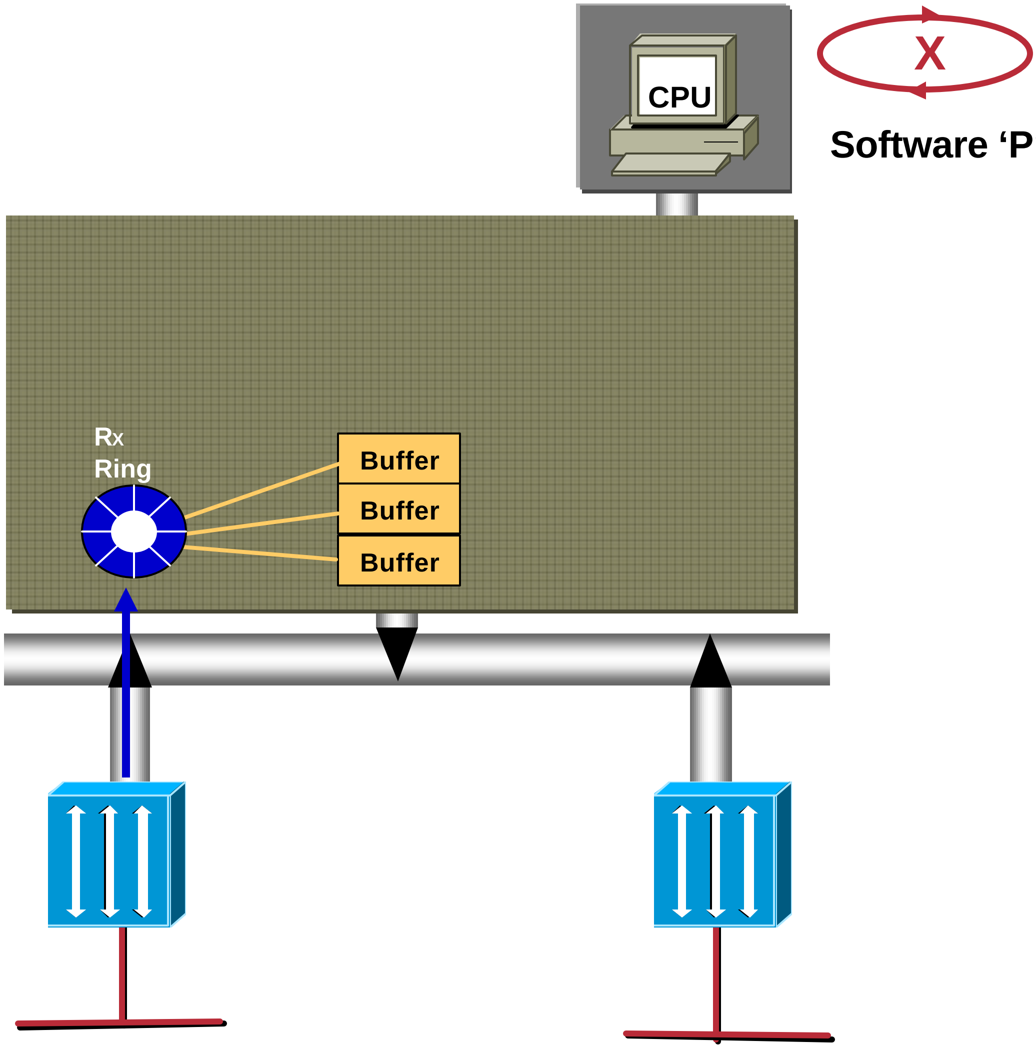
ARP Table

172.16.1.1:	0F000800
172.16.2.1:	10134567A...ECE030178654

Fast Cache

Prefix/Length	Age	Interface	Next Hop
1.1.0.0/16	00:00:15	Ethernet0	172.16.2.1 14 00000C7EF7CF00E0B06423F60800
10.1.1.0/24	00:00:15	Serial1	172.16.1.1 4 0F000800

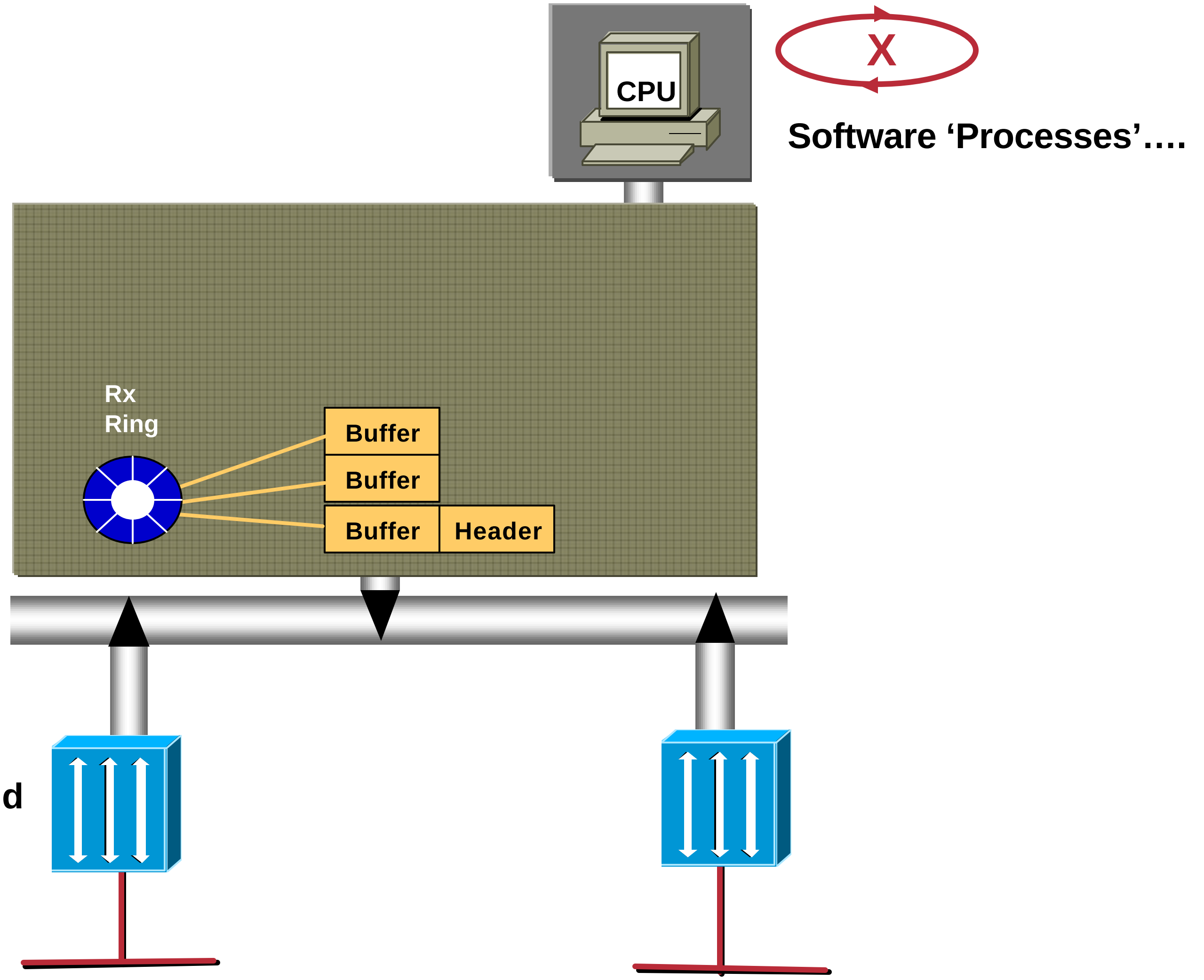
Router Switching Operation
”Fast” Switching



1

Interface Processor DMA’s packet into RX Ring Buffer

Router Switching Operation ”Fast” Switching



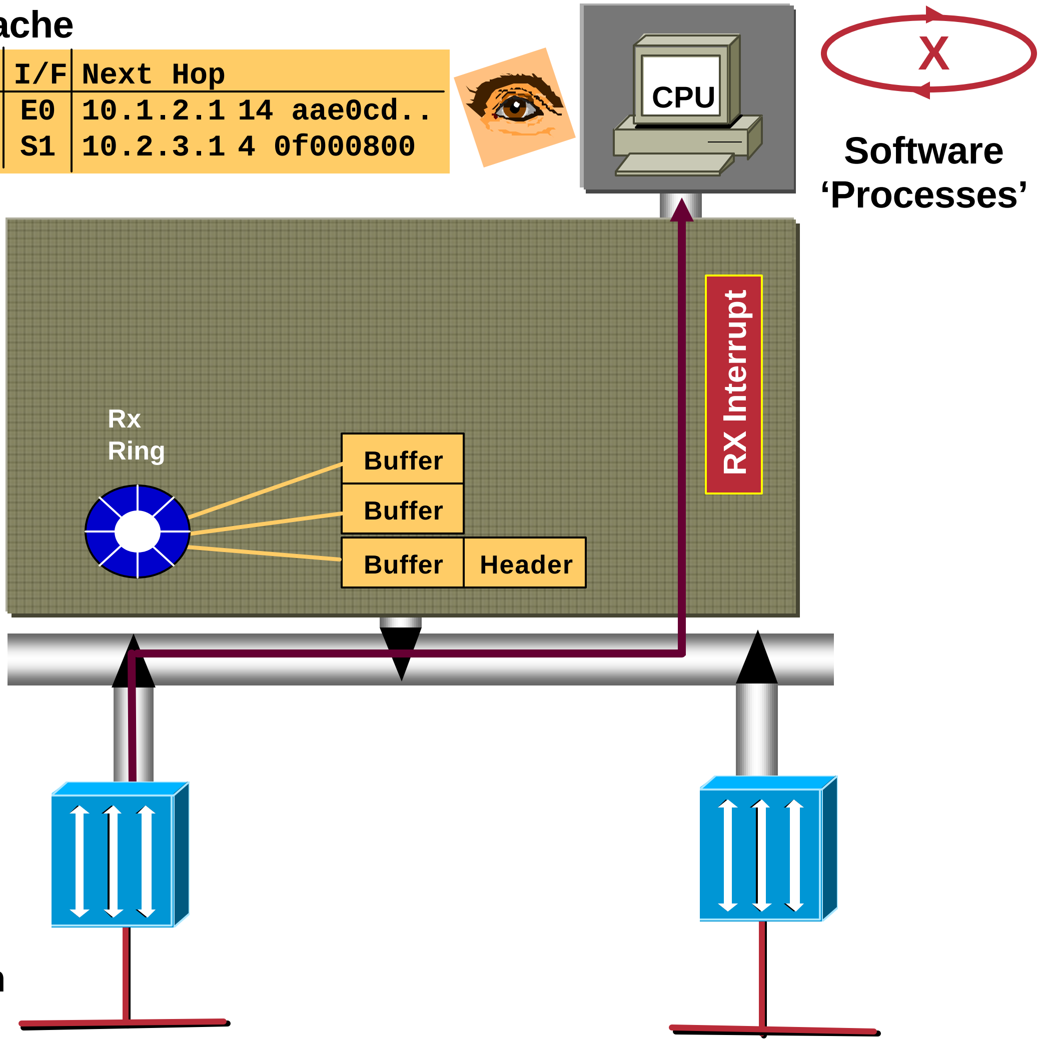
2

Interface Driver Code Decodes packet header and builds Buffer Header with L3 Info

Router Switching Operation
”Fast” Switching

Simplified Optimum Cache

Prefix	Age	I/F	Next Hop
10.1.2.3/32	00:00:15	E0	10.1.2.1 14 aae0cd..
11.1.2.0/24	00:00:15	S1	10.2.3.1 4 0f000800



3

Interface processor generates RX Interrupt to CPU.

CPU Halts current process and attempts to fast switch packet

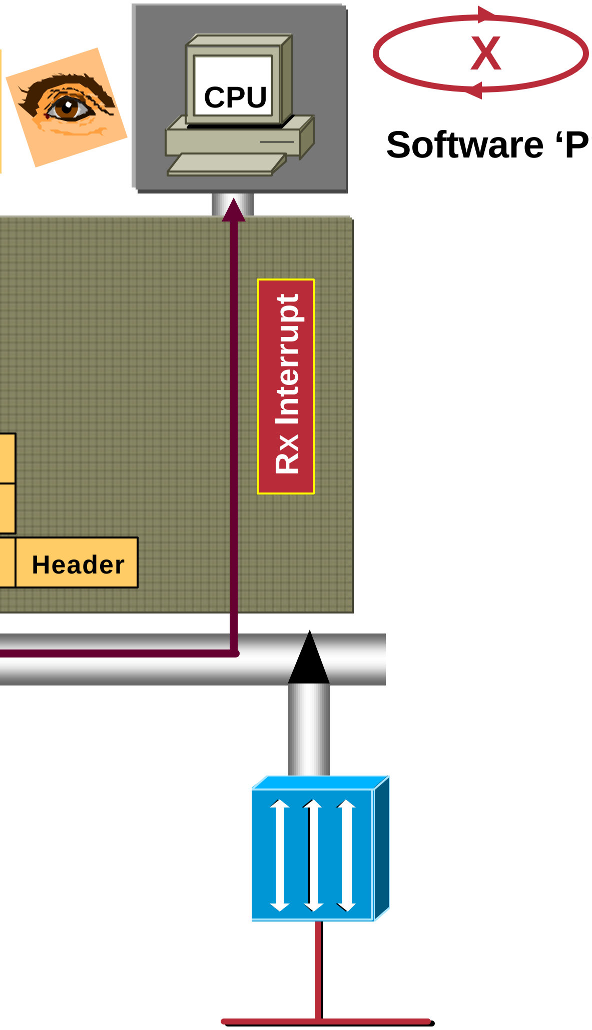
Router Switching Operation ”Fast” Switching

Simplified Optimum Cache

Prefix	Age	I/F	Next Hop
10.1.2.3/32	00:00:15	E0	10.1.2.1 14 aae0cd..
11.1.2.0/24	00:00:15	S1	10.2.3.1 4 0f000800

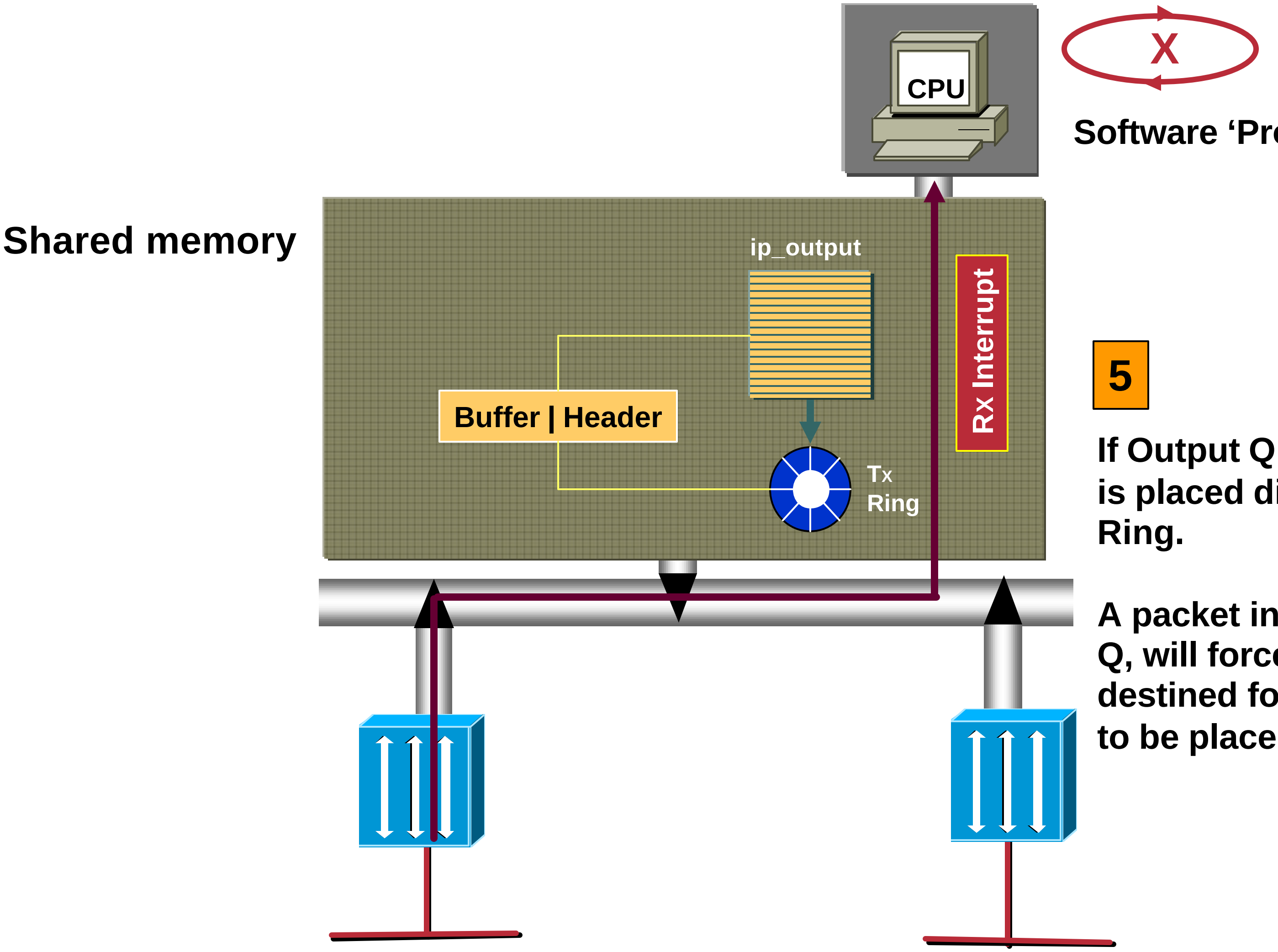
4

Optimum Cache entry used to Write MAC header



Software ‘Processes’….

Router Switching Operation
”Fast” Switching



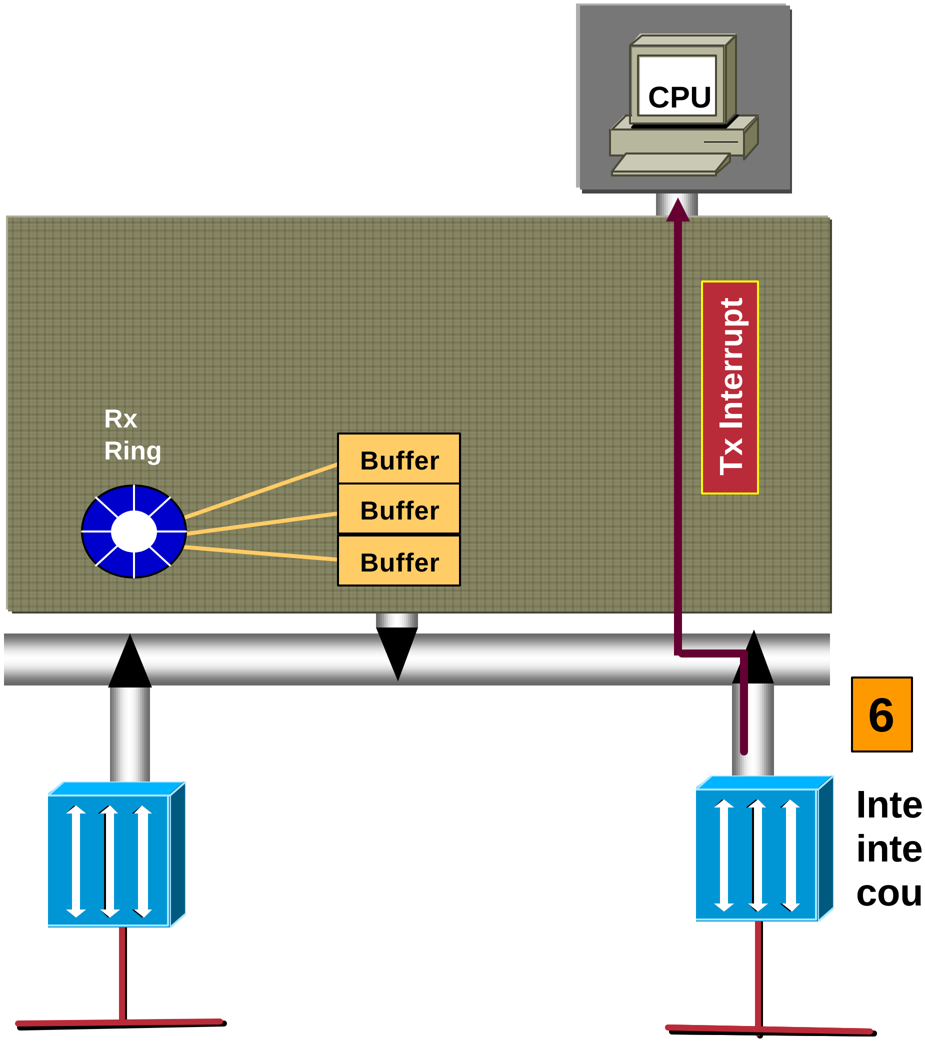
Software ‘Processes’….

5

If Output Q is Empty packet is placed directly on the TX Ring.

A packet in the Output Hold Q, will force other packets destined for that interface to be placed in the Q.

Router Switching Operation
”Fast” Switching



Interface Instigates a TX interrupt. Increment counters, SNMP etc..

Demand Generated Cache Based Switching Issues

- First packet towards a given destination is always process switched
- Fast cache entries must be timed out periodically to prevent stale information from being used in switching
- When an arp entry or the routing table changes, we must clear some portion of the fast cache and wait for process switched traffic to rebuild it
- We store a prebuilt mac header for each possible destination. This waste space and causes duplicated effort

Show Processes

```
7206#show processes
CPU utilization for five seconds: 0%/0%; one minute: 0%; five minutes: 0%
 PID QTy       PC Runtime (ms)    Invoked   uSecs    Stacks TTY Process
   2 M*         0            8         86      93 9888/12000  0 Exec
   3 Lst 60655C58       345736     129733    2664 5740/6000   0 Check heaps
   4 Cwe 6064C268            4          1    4000 5568/6000   0 Chunk Manager
   5 Cwe 6065BC70           12         17     705 5596/6000   0 Pool Manager
  14 Lwe 60719100         5604     103710      54 5236/6000   0 ARP Input
  20 Cwe 60661090            0          1       0 5608/6000   0 Critical Bkgnd
  21 Mwe 6061BC70          232     209650       110164/12000  0 Net Background
  22 Lwe 605ACD38            0         26       011504/12000  0 Logger
  24 Msp 6061B1C0        32336    1277140      25 6920/9000   0 Per-Second Jobs
  35 Mwe 60747998         4276      64668      6610648/12000  0 IP Input
  82 Msp 6061B200        85188      21328    3994 5660/6000   0 Per-minute Jobs
```

For the 5 Sec window we have both the total CPU time and the Interrupt time

Show Processes CPU

```
7206#show processes cpu
CPU utilization for five seconds: 0%/0%; one minute: 0%; five minutes: 0%
 PID  Runtime(ms)  Invoked  uSecs    5Sec   1Min   5Min TTY Process
   2           68      227    299   0.00%  0.00%  0.00%   0 Exec
   3       368920   138425   2665   0.08%  0.02%  0.00%   0 Check heaps
   4            4        1   4000   0.00%  0.00%  0.00%   0 Chunk Manager
   5           20       21    952   0.00%  0.00%  0.00%   0 Pool Manager
  14         6608   119562     55   0.00%  0.00%  0.00%   0 ARP Input
  20            0        1      0   0.00%  0.00%  0.00%   0 Critical Bkgnd
  21          248   218242      1   0.00%  0.00%  0.00%   0 Net Background
  22            0       28      0   0.00%  0.00%  0.00%   0 Logger
  24        35704  1362619     26   0.00%  0.00%  0.00%   0 Per-Second Jobs
  35         4520    68993     65   0.00%  0.00%  0.00%   0 IP Input
  82        90896    22759   3993   0.00%  0.00%  0.00%   0 Per-minute Jobs
```

More specific information on the CPU time occupied by the Processes

Cisco Express Forwarding

Cisco Express Forwarding

- Background
- CEF Theory
- The CEF Mtrie
- The Adjacency Table
- Adjacency Table Entries
- Load Sharing with CEF
- CEF Accounting

Background: Process Level Switching

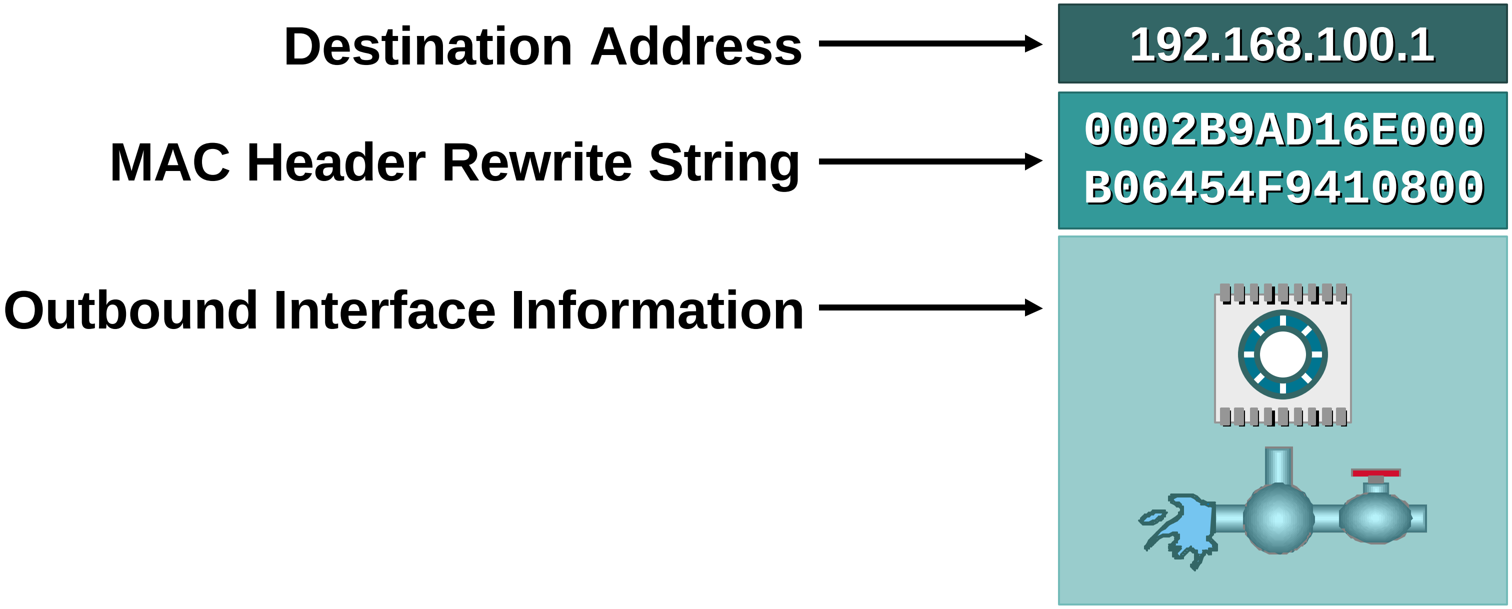
- Process Level Switching has speed limitations on high speed networks

Background: Fast Switching

- Caching the results of the lookup routines was the first solution and is known as Fast Switching
- This solution encounters scalability problems on Internet backbone routers where the routing table is changing rapidly and there are many different flows of traffic
- CEF (Cisco Express Forwarding) was developed to address the scalability issues of Process and Fast Switching
- CEF doesn’t cache switching information, it builds switching tables

CEF Theory

What Do We Need to Switch a Packet?



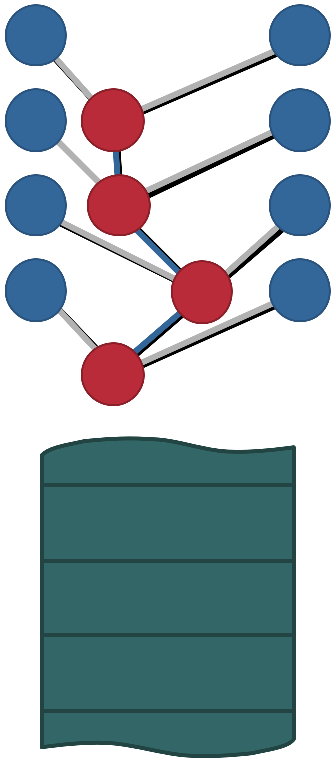
Destination Address → 192.168.100.1

MAC Header Rewrite String → 0002B9AD16E000 B06454F9410800

Outbound Interface Information →

CEF Theory

CEF Builds Two Tables to Contain this Information:



The CEF Mtrie

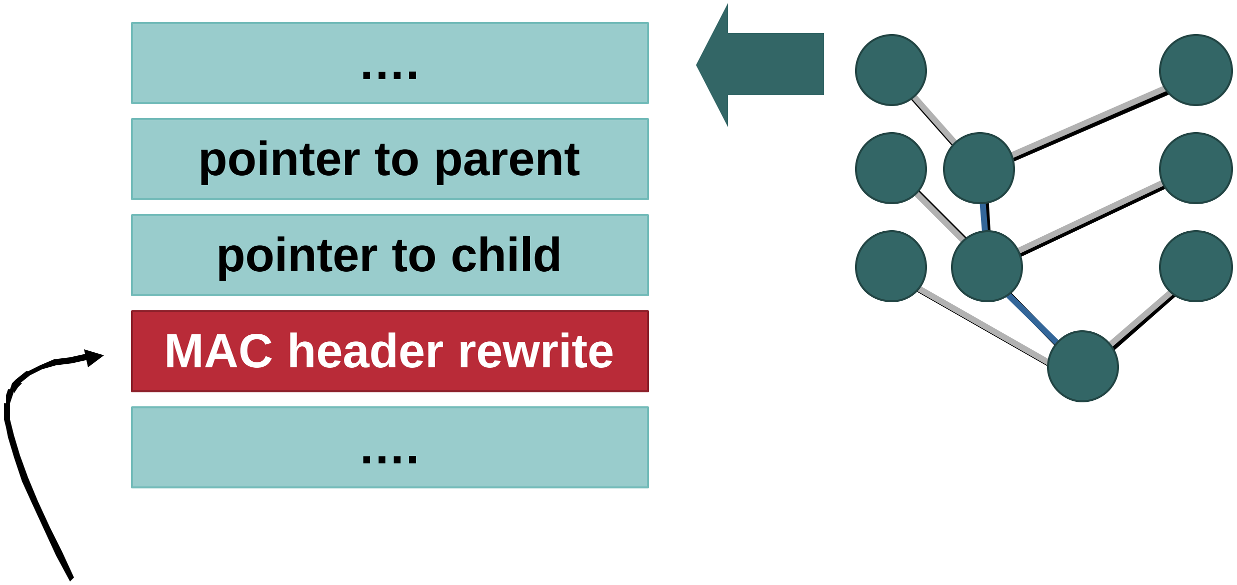
The Adjacency Table

CEF Packet Switching

- **Read in packet from the interface and store packet into memory**
- **Raise an interrupt to the processor; the rest of the packet switching takes place within the interrupt**
- **Use CEF mtrie to lookup packet destination; determine correct next-hop info by following pointer in the last CEF mtrie node**
- **Use Adjacency table info to rewrite physical layer header**
- **Place packet on the outbound interface queue**

CEF Theory

What’s the Difference between a Tree and a Trie?



....

pointer to parent

pointer to child

MAC header rewrite

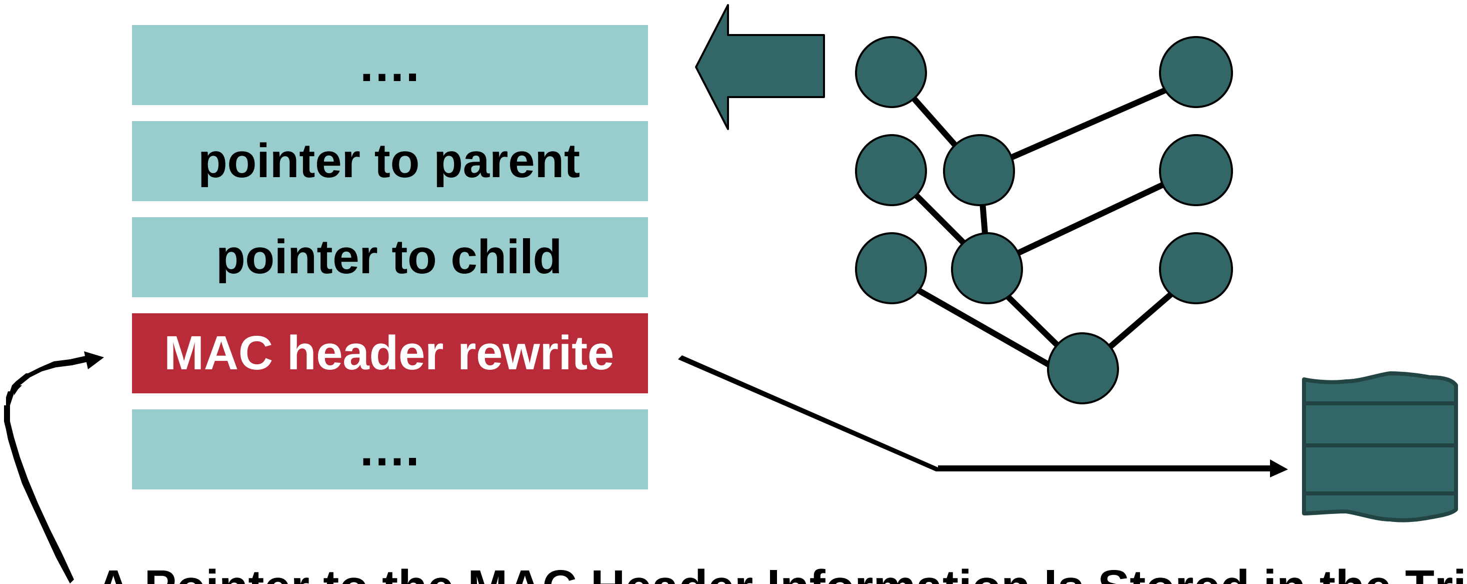
....

The MAC Header Rewrite Information Is Stored in the Tree Itself

CEF Theory

Cisco.com

What's the Difference between a Tree and a Trie?



....

pointer to parent

pointer to child

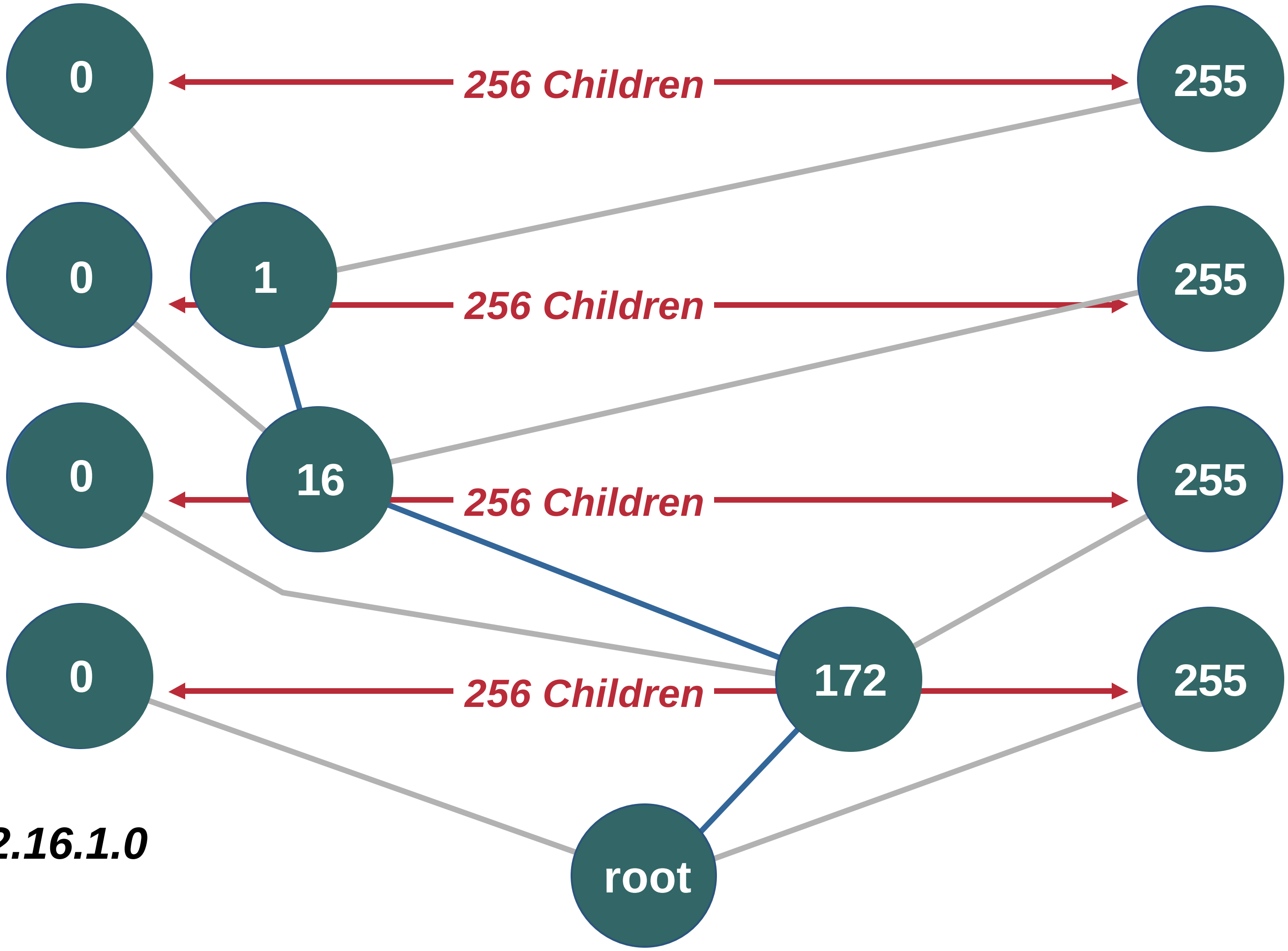
MAC header rewrite

....

A Pointer to the MAC Header Information Is Stored in the Trie, and the MAC Header Information Itself Is Stored in a Separate Table

The CEF Mtrie

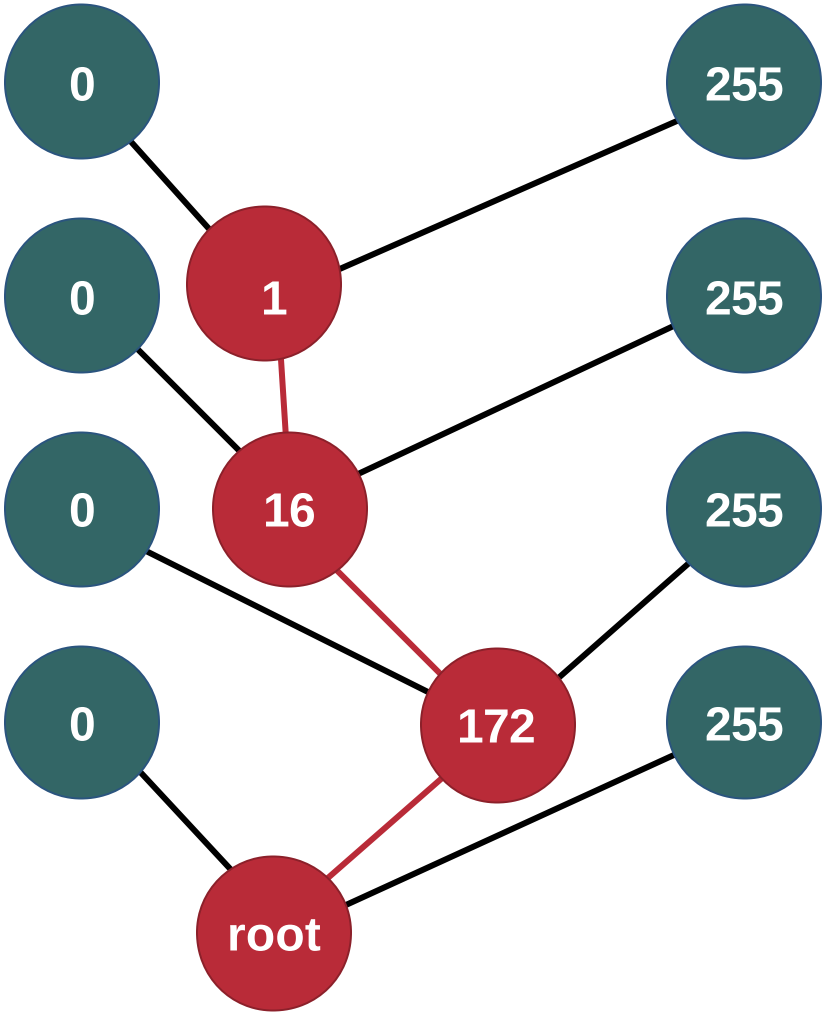
Cisco.com



172.16.1.0

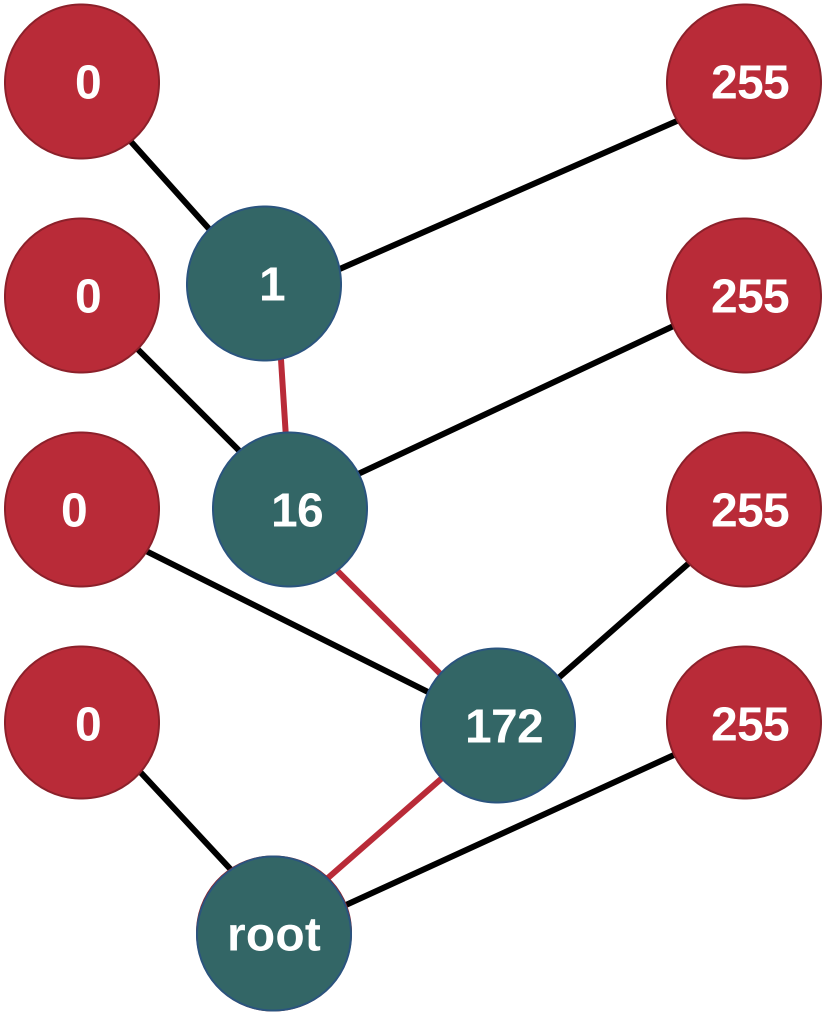
The CEF Mtrie

- **Nodes point to other nodes or leaves**



The CEF Mtrie

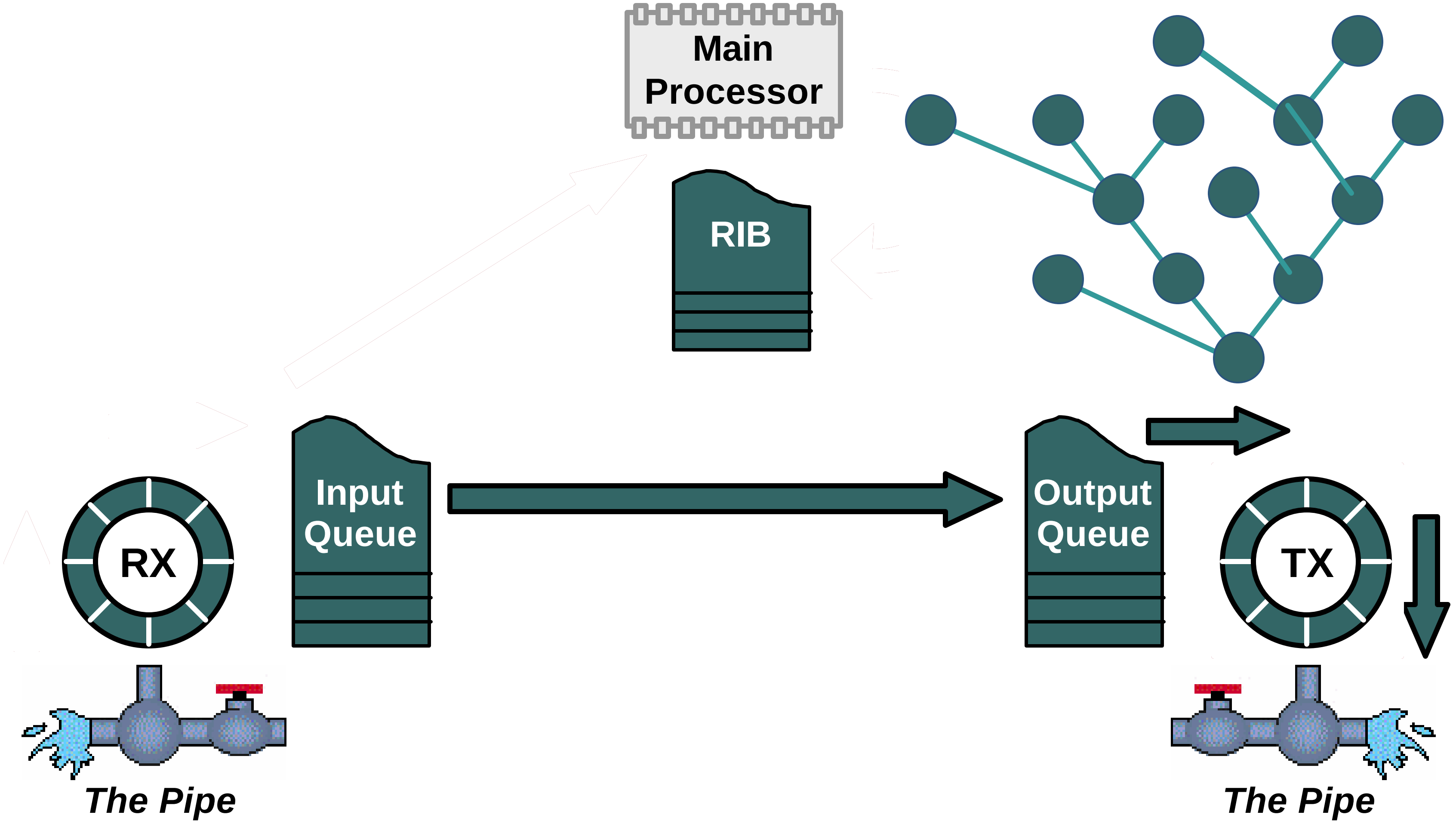
- **Leaves point to the adjacency table**



The CEF MTrie

```
Router#sh ip cef summary
IP CEF with switching (Table Version 4)
  4 routes, 0 reresolve, 0 unresolved (0 old, 0 new), peak 0
  4 leaves, 8 nodes, 8832 bytes, 4 inserts, 0 invalidations
  0 load sharing elements, 0 bytes, 0 references
  universal per-destination load sharing algorithm, id 20340B24
  1 CEF resets, 0 revisions of existing leaves
  0 in-place/0 aborted modifications
  Resolution Timer: Exponential (currently 1s, peak 1s)
  refcounts:  533 leaf, 536 node
```

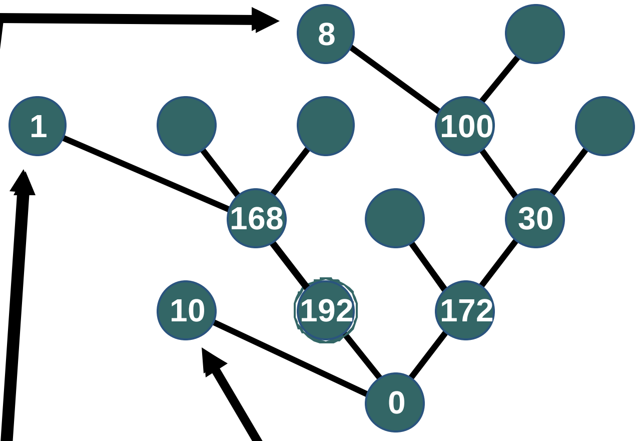
The CEF Mtrie



The CEF Mtrie Notes

- Where in the switching path do we build the CEF table?
- Nowhere! The CEF table is built from the routing table before (and while) packets are being switched
- Because the CEF table is directly related to the routing table, we can build it for every destination in the routing table without waiting on any packets to be switched

Two Separate Tables

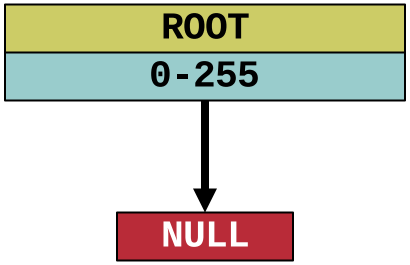


```
router#show ip route
     172.30.0.0/24 is subnetted, 1 subnets
S       172.30.100.8/32 via Serial 0/0
C    192.168.1.0/24 is directly connected, Ethernet 1/2
S    10.0.0.0/8 via POS 4/1
```

The Routing Table and the CEF Mtrie Are Directly Related

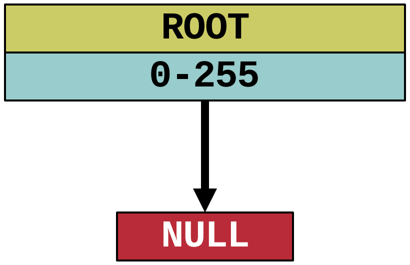
The CEF Table Contains Reachability and Next Hop Information

The CEF Mtrie



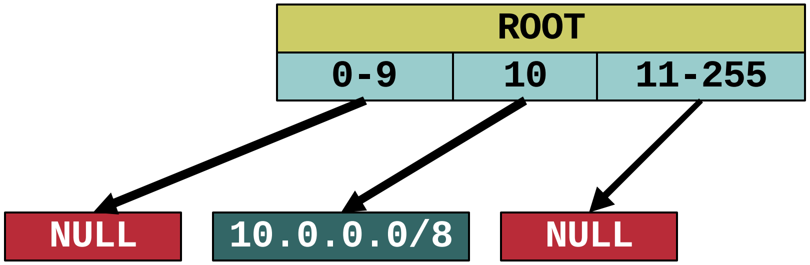
Empty Table

The CEF Mtrie



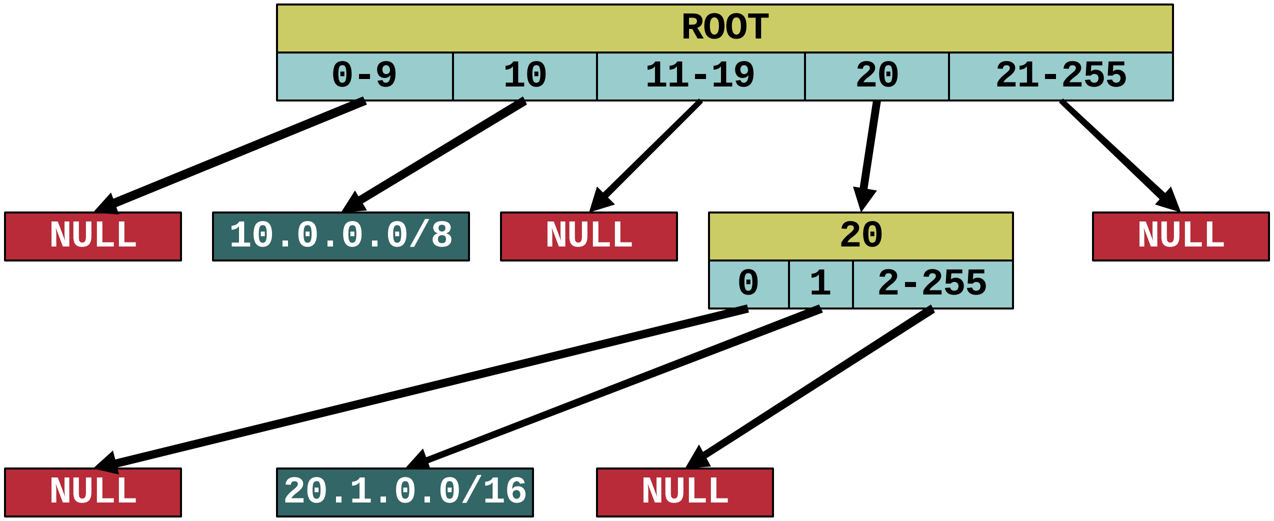
Add 10.0.0.0/8

The CEF Mtrie



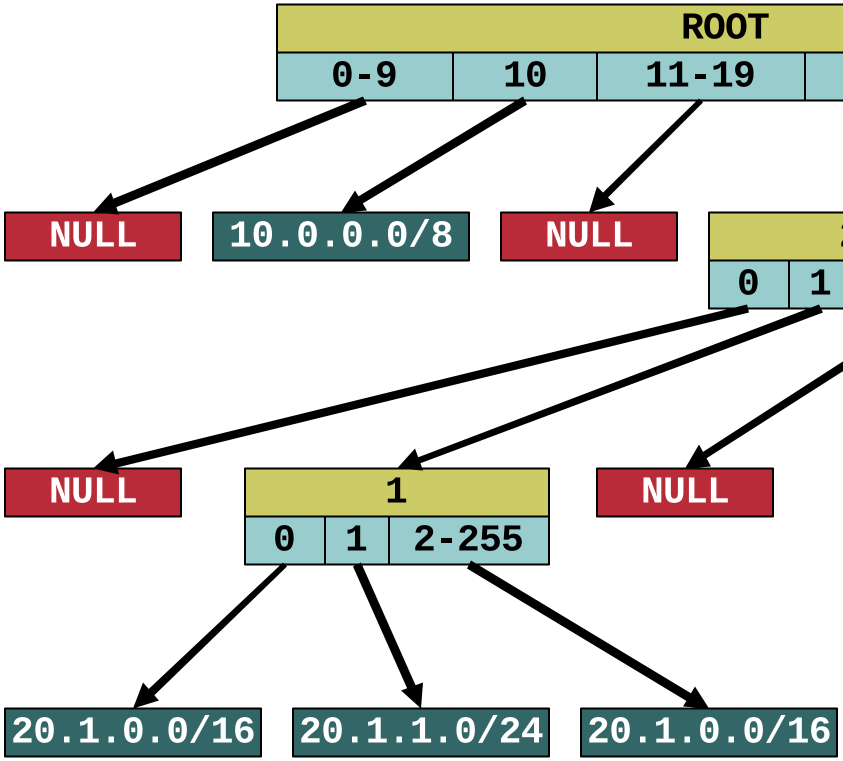
Add 20.1.0.0/16

The CEF Mtrie



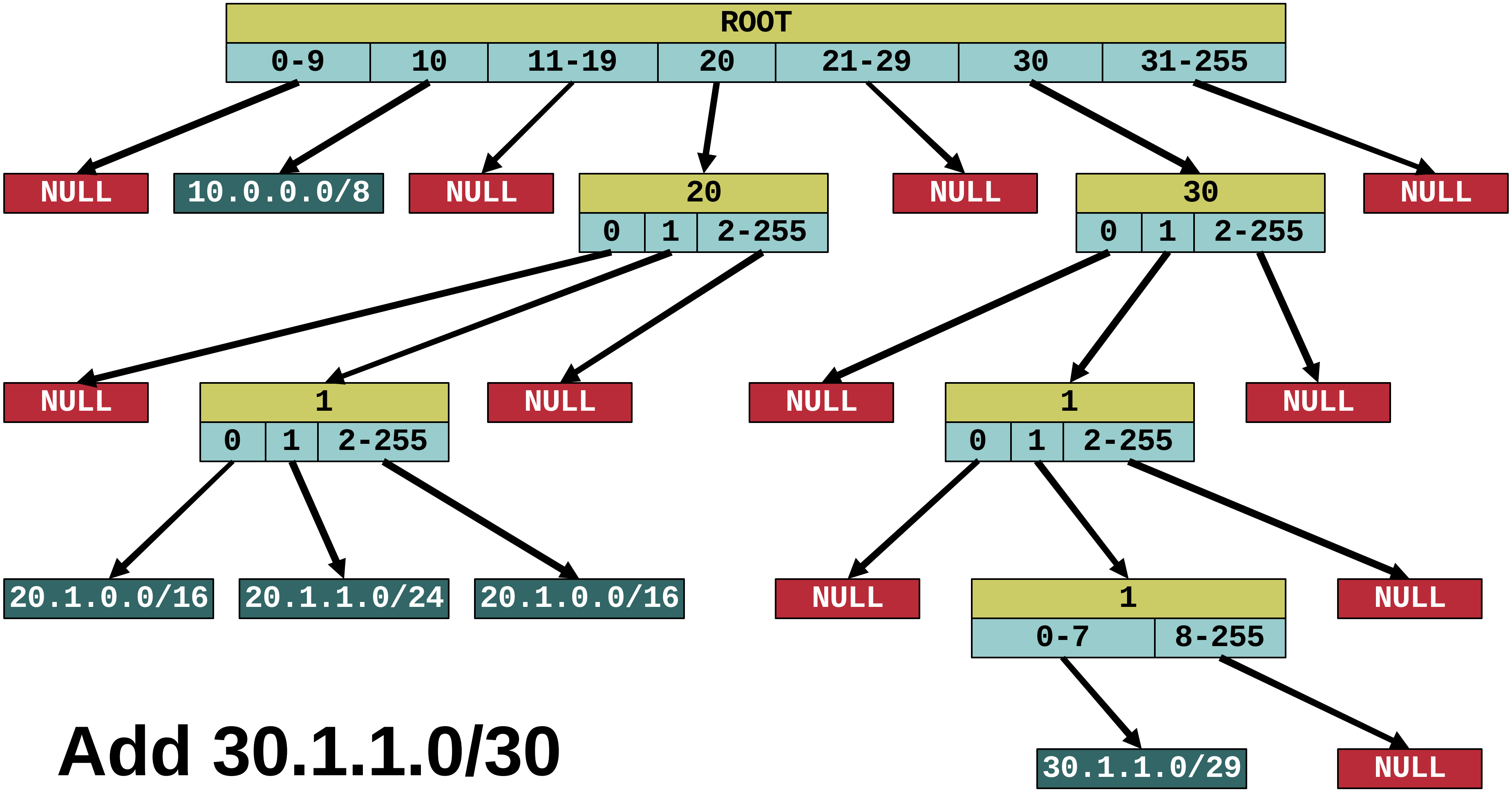
Add 20.1.1.0/24

The CEF Mtrie



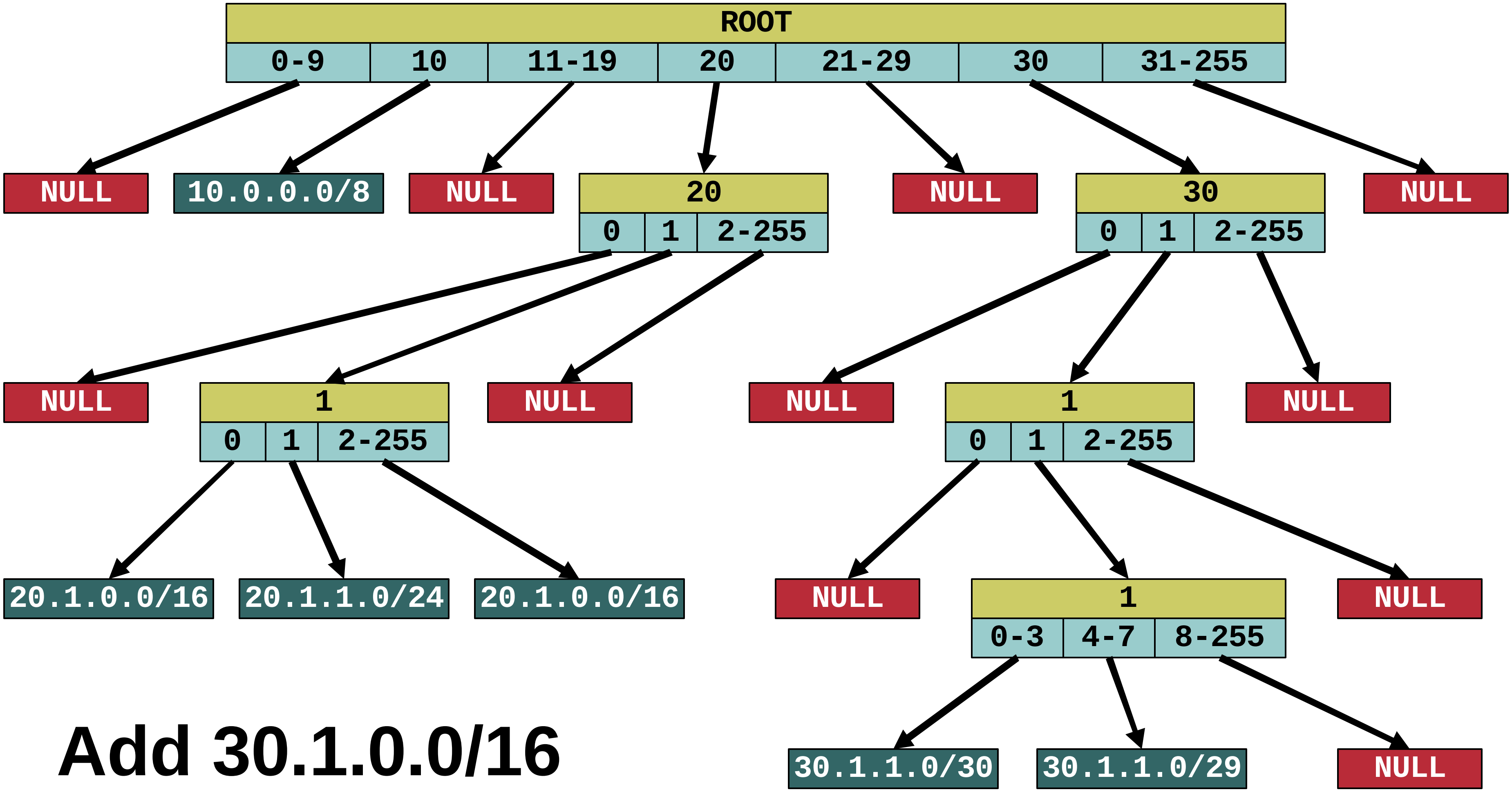
Add 30.1.1.0/29

The CEF Mtrie



Add 30.1.1.0/30

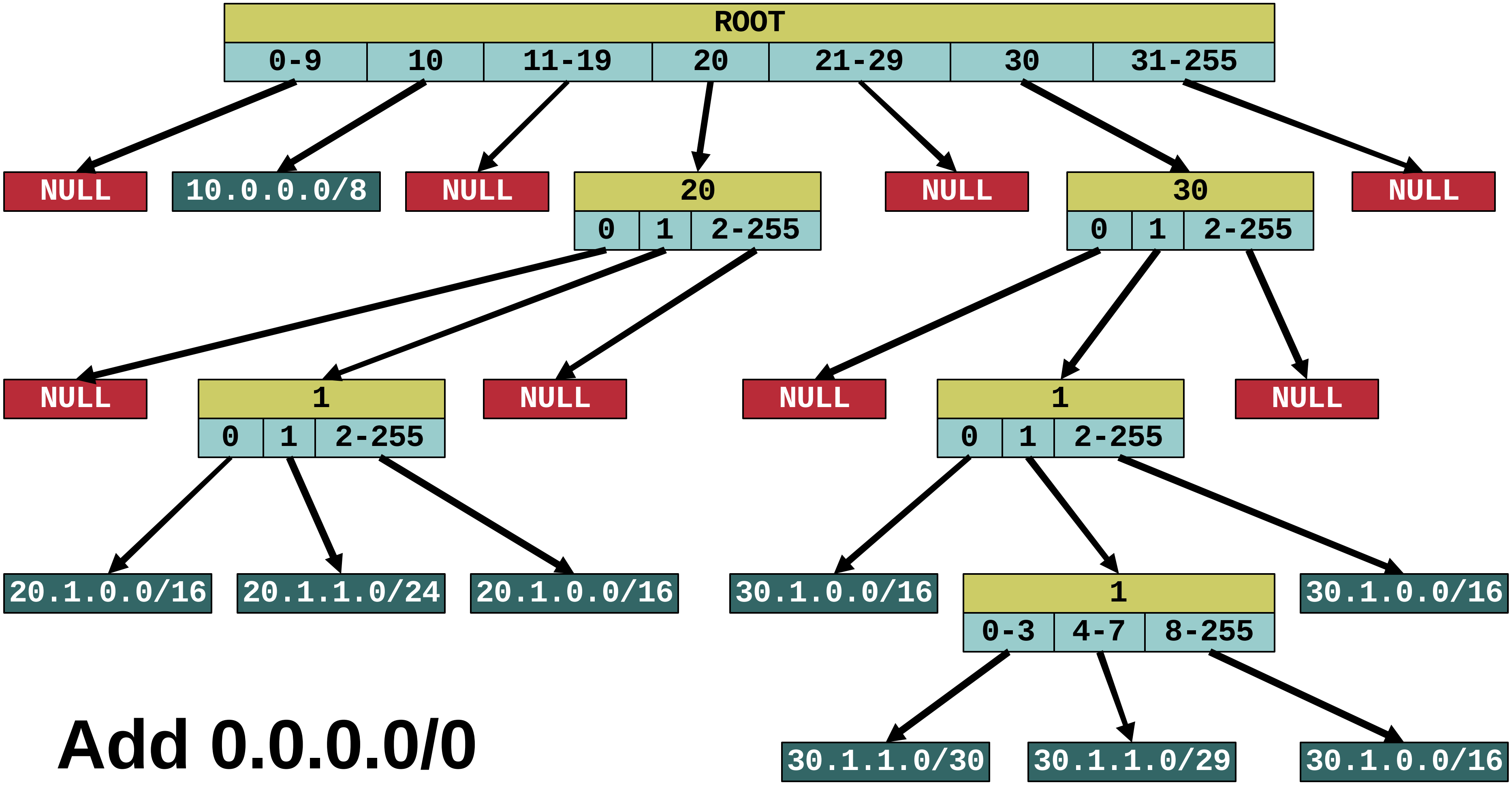
The CEF Mtrie



Add 30.1.0.0/16

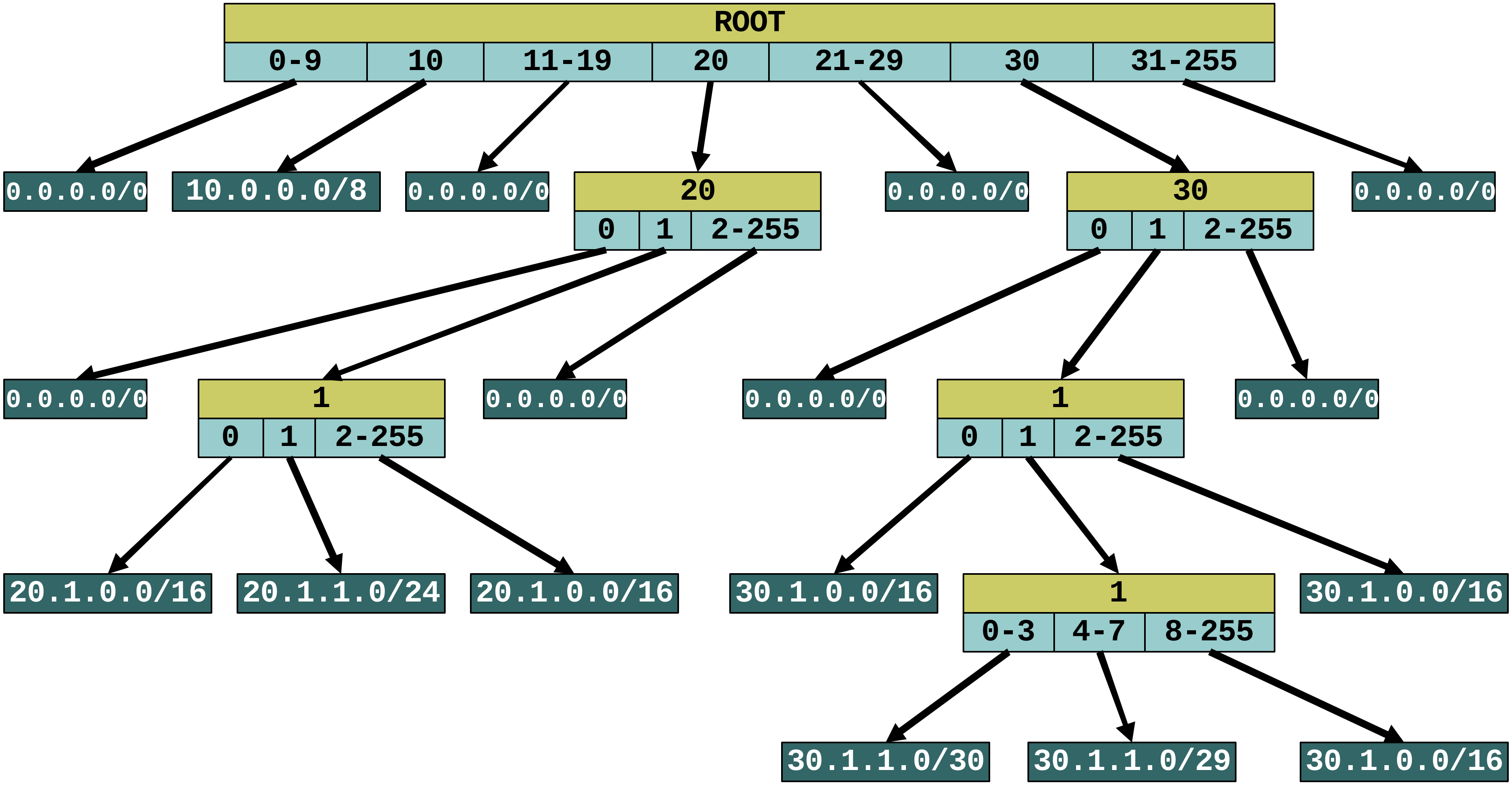
The CEF Mtrie

Cisco.com



Add 0.0.0.0/0

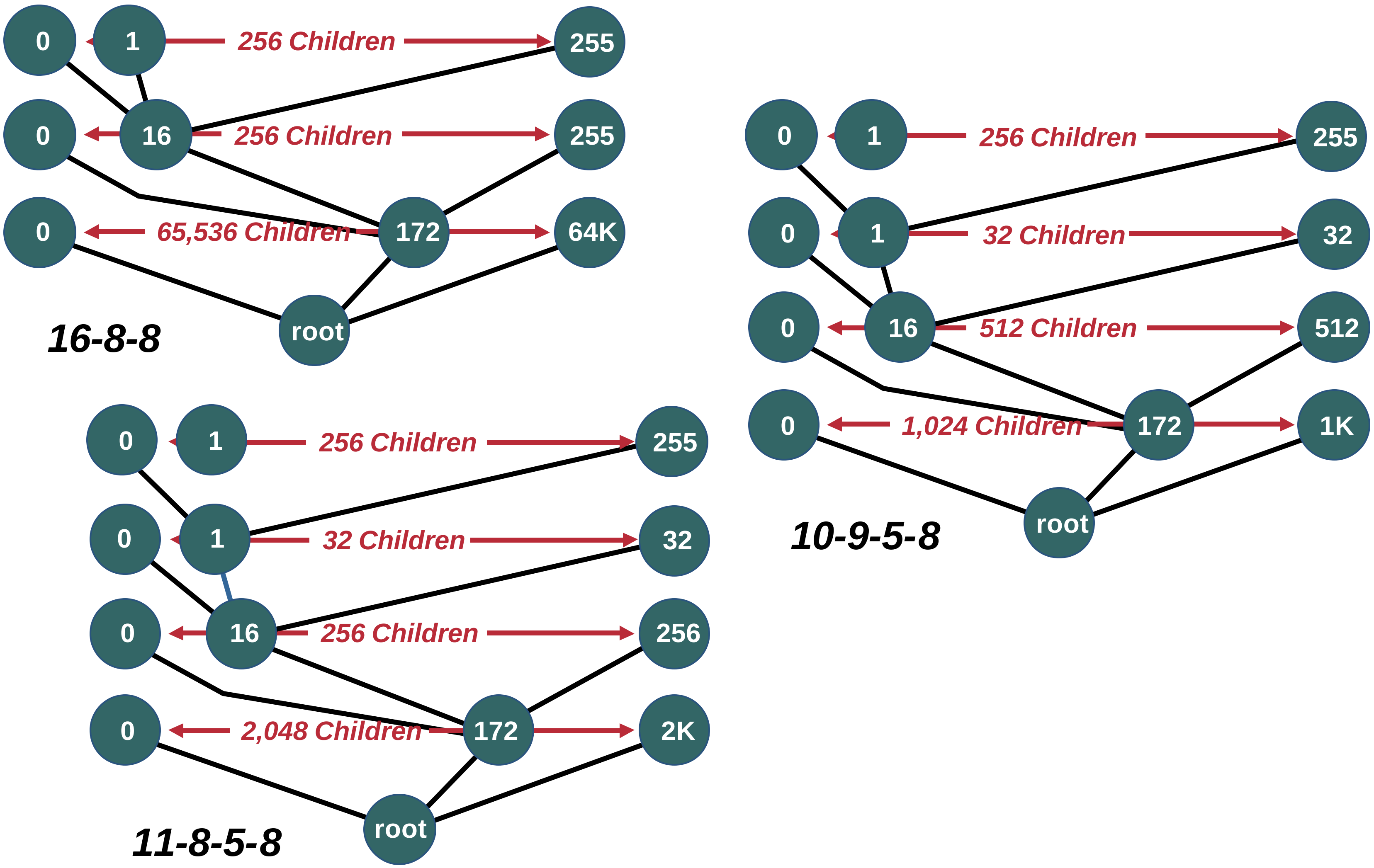
The CEF Mtrie



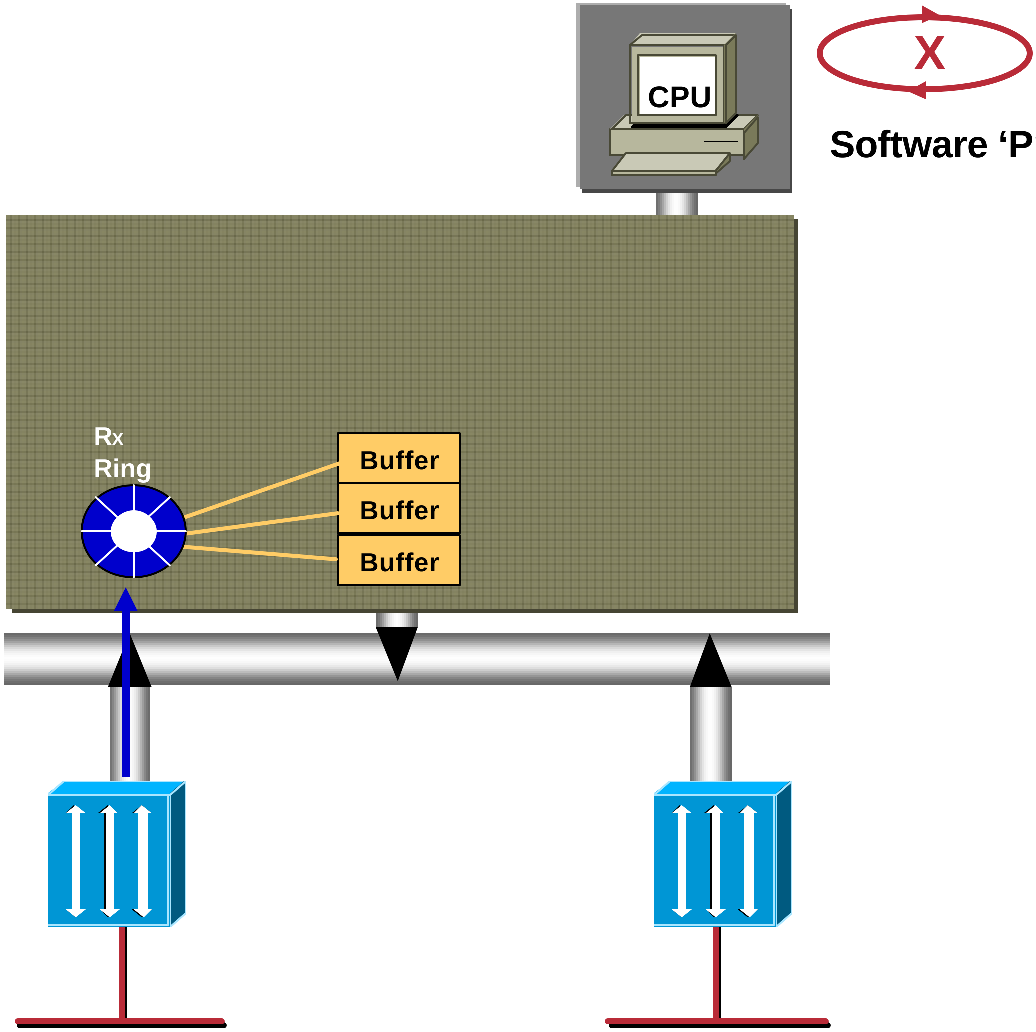
The CEF Mtrie

- Normally there are 4 levels of nodes with each node having 255 children
- Prefix and traffic distribution sometimes makes the mtrie perform better if there are different numbers of children for nodes at each level

The CEF MTrie



Path Through the CEF Switch Code

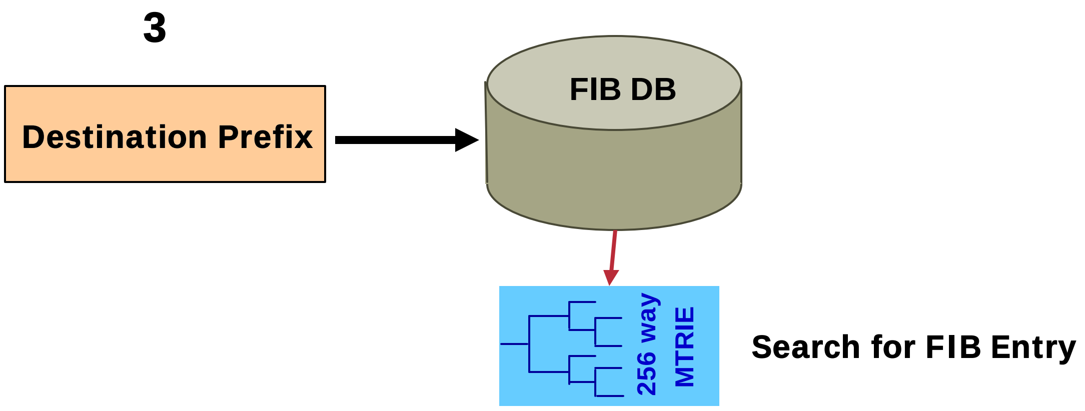


Software 'Processes'….

Interface Processor DMA's packet into RX Ring Buffer

Path Through the CEF Switch Code

1. A packet arrives at an input interface, RX Interrupt generated
2. Read IP Destination Prefix
3. Search CEF's FIB DB, using the Destination Prefix as Search Key



Path Through the CEF Switch Code

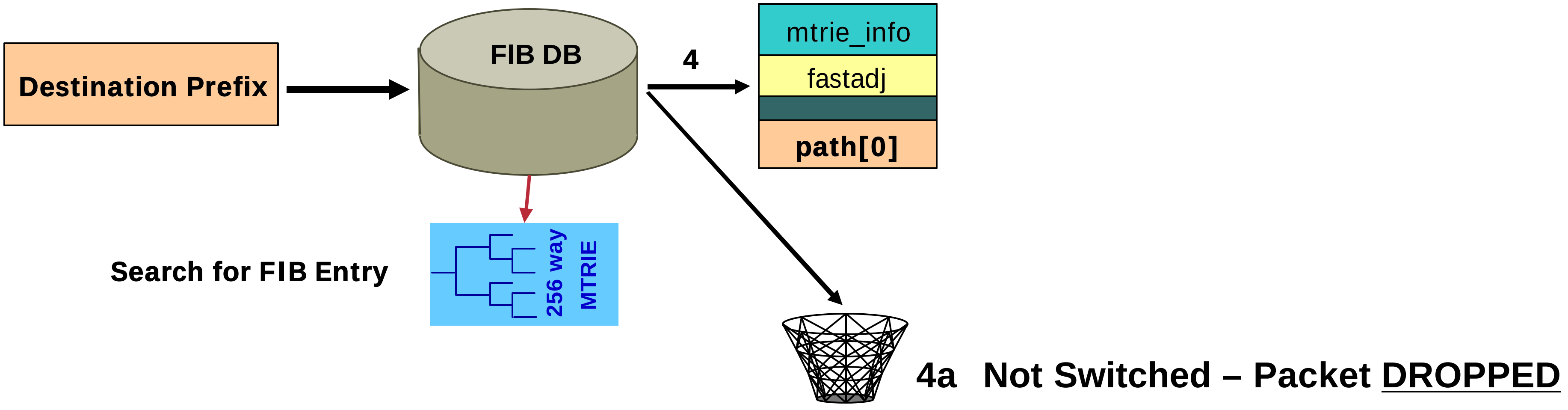
1. A packet arrives at an input interface, RX Interrupt generated

2. Read IP Destination Prefix

3. Search CEF’s FIB DB, using the Destination Prefix as Search Key

4. A Successful MTRIE Lookup will result in a FIB Entry being Found

4a. If the MTRIE Lookup is unsuccessful, the packet will be dropped



Path Through the CEF Switch Code

1. A packet arrives at an input interface, RX Interrupt generated

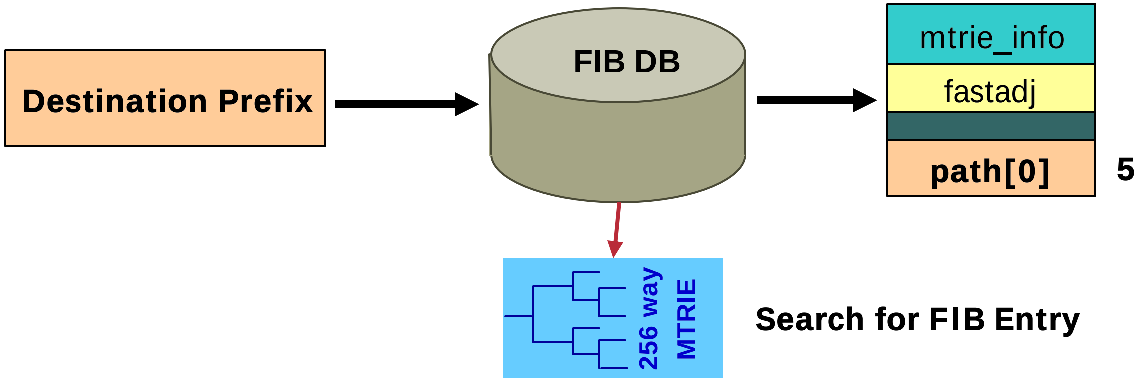
2. Read IP Destination Prefix

3. Search CEF’s FIB DB, using the Destination Prefix as Search Key

4. A Successful MTRIE Lookup will result in a FIB Entry being Found

4a. If the MTRIE Lookup is unsuccessful, the packet will be dropped

5. FIB Path is selected



Path Through the CEF Switch Code

1. A packet arrives at an input interface, RX Interrupt generated

2. Read IP Destination Prefix

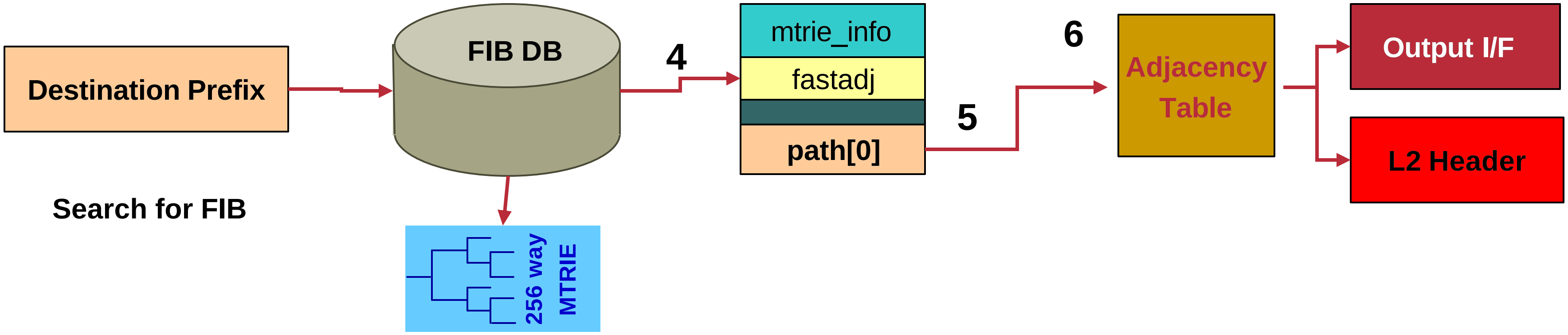
3. Search CEF’s FIB DB, using the Destination Prefix as Search Key

4. A Successful MTRIE Lookup will result in a FIB Entry being Found

4a. If the MTRIE Lookup is unsuccessful, the packet will be dropped

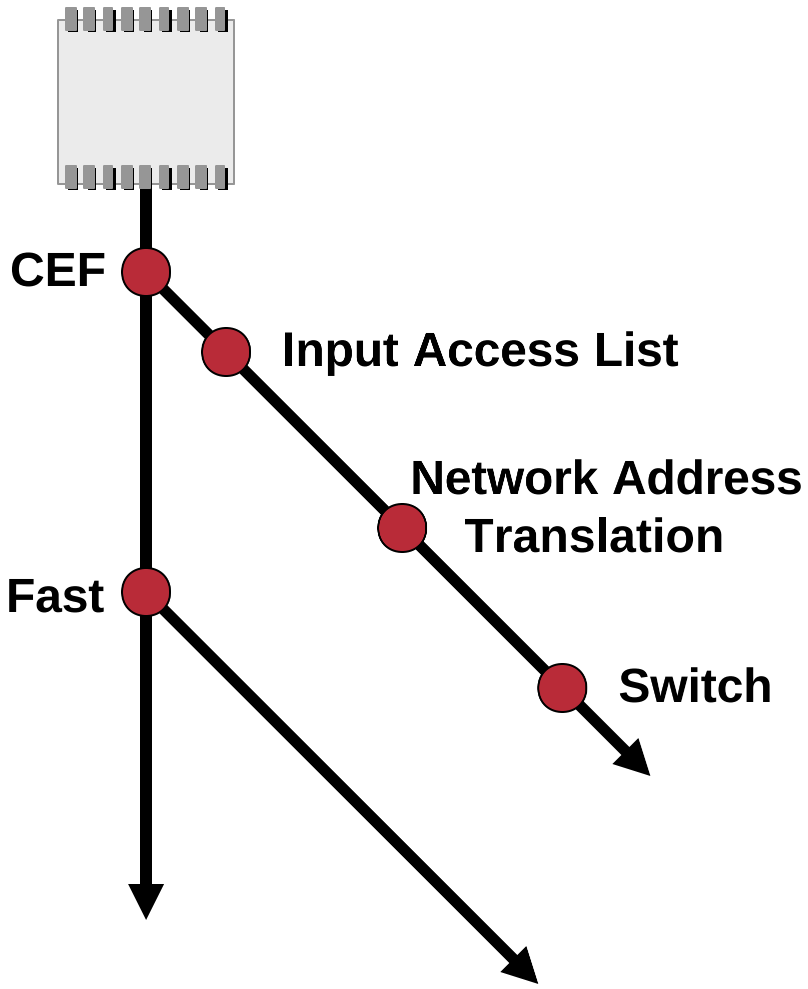
5. FIB Path is selected

6. Selected FIB Path will point to necessary entry in Adjacency Table



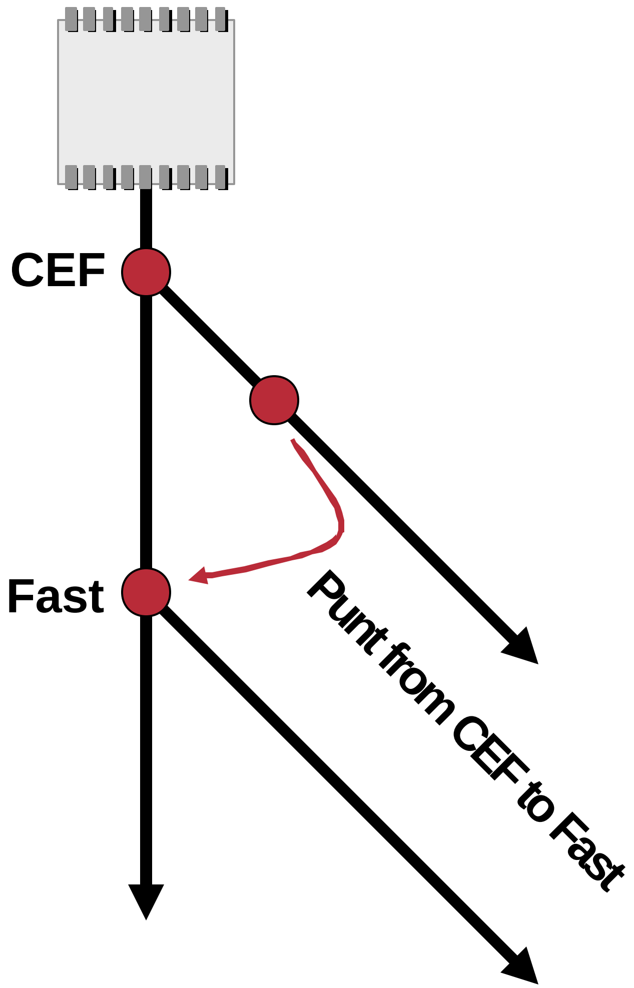
Switch During the Receive Interrupt

- Features are processed along each switching path.
- Each feature represents a function call which may fail, succeed, or just not exist.



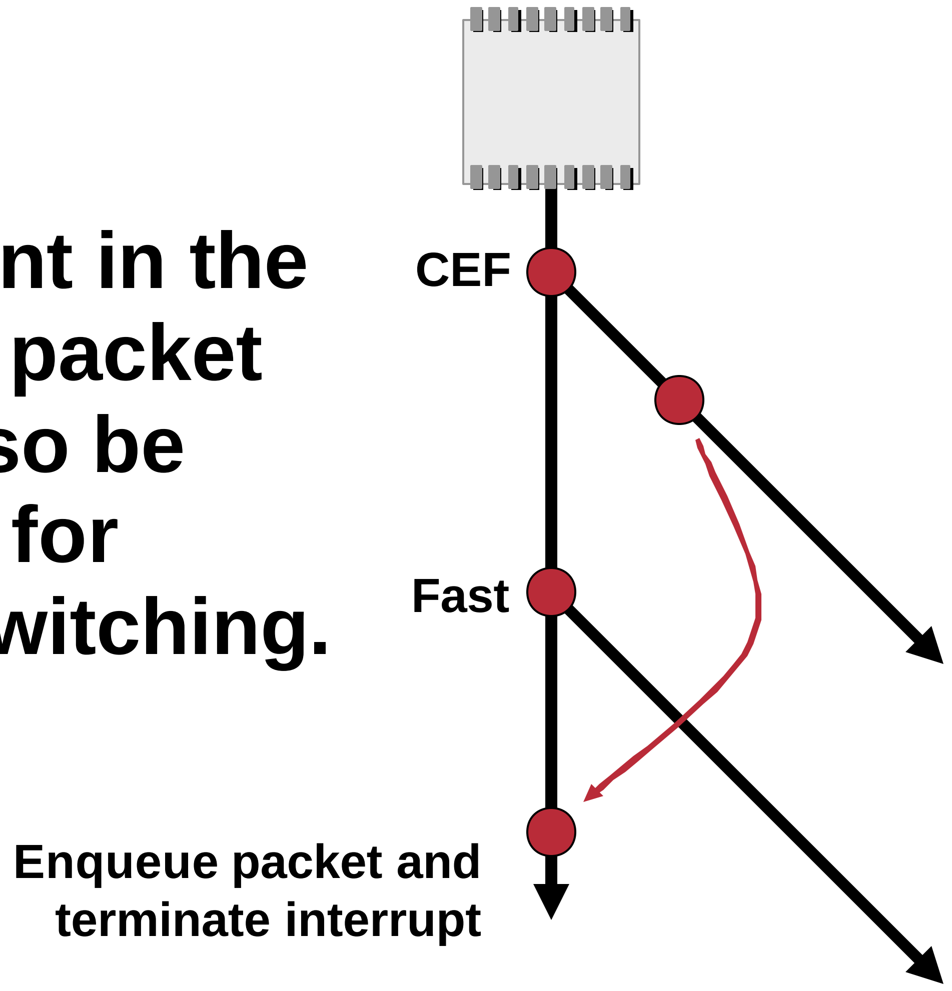
Switch During the Receive Interrupt

- At any point while the packet is being processed, it can be punted to the next slower process by allowing the processor to jump to the next pointer in the chain.



Switch During the Receive Interrupt

- **At any point in the chain, the packet may be also be enqueued for process switching.**



The CEF Mtrie

Depending on the Type of Route, a CEF Table Entry Can Be Several Different Types

- Attached
- Connected
- Receive
- Recursive

The CEF Mtrie

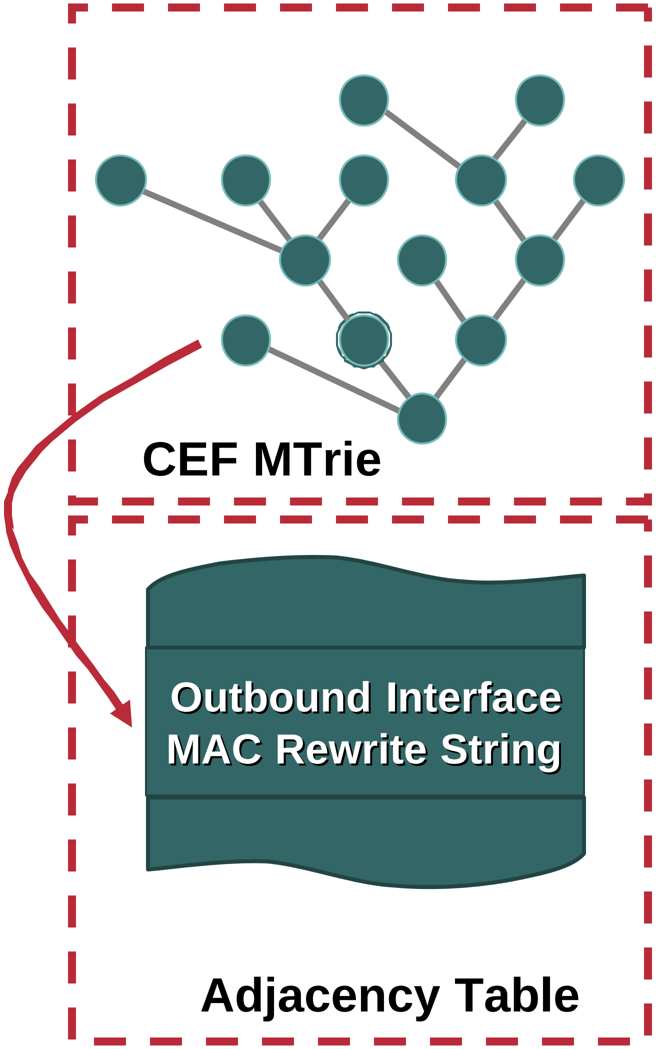
- **Attached—An “attached” mtrie entry means the destination is attached to the router**
- **Connected—A “connected” entry is the result of an ip address being configured on an interface**
- **An entry may be both Attached and Connected**

The CEF Mtrie

- **Receive—Indicates packets that are destined to the router and do not need to be switched to another interface**
- **Recursive—References another node to find the next-hop information**

The Adjacency Table

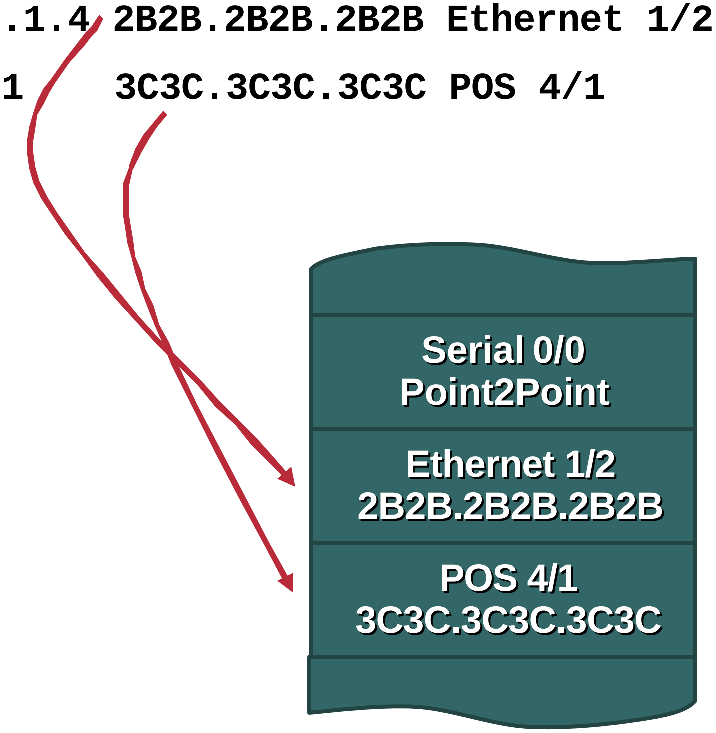
- **The Mtrie is used to look up the next-hop for a prefix**
- **The final node encountered in the Mtrie during a prefix lookup includes a pointer to the correct next-hop in the adjacency table**



The Adjacency Table

```
router#show arp
Address       Hardware Addr   Interface
192.168.1.4 2B2B.2B2B.2B2B Ethernet 1/2
10.1.1.1      3C3C.3C3C.3C3C POS 4/1
```

- **The ARP Cache and the Adjacency Table are directly related**
- **The adjacency table doesn’t contain any information about networks; it only contains information about next hops**



The Adjacency Table

- Allows next-hops to change without changing the mtrie
- A change in next-hop just requires the final mtrie node’s pointer to the adjacency table to be updated
- Routing table changes also don’t impact the adjacency table

The Adjacency Table

- Update the FIB when changes in the routing table occur
- Update the adjacency table when changes in connected adjacencies occur

Adjacency Table Entries

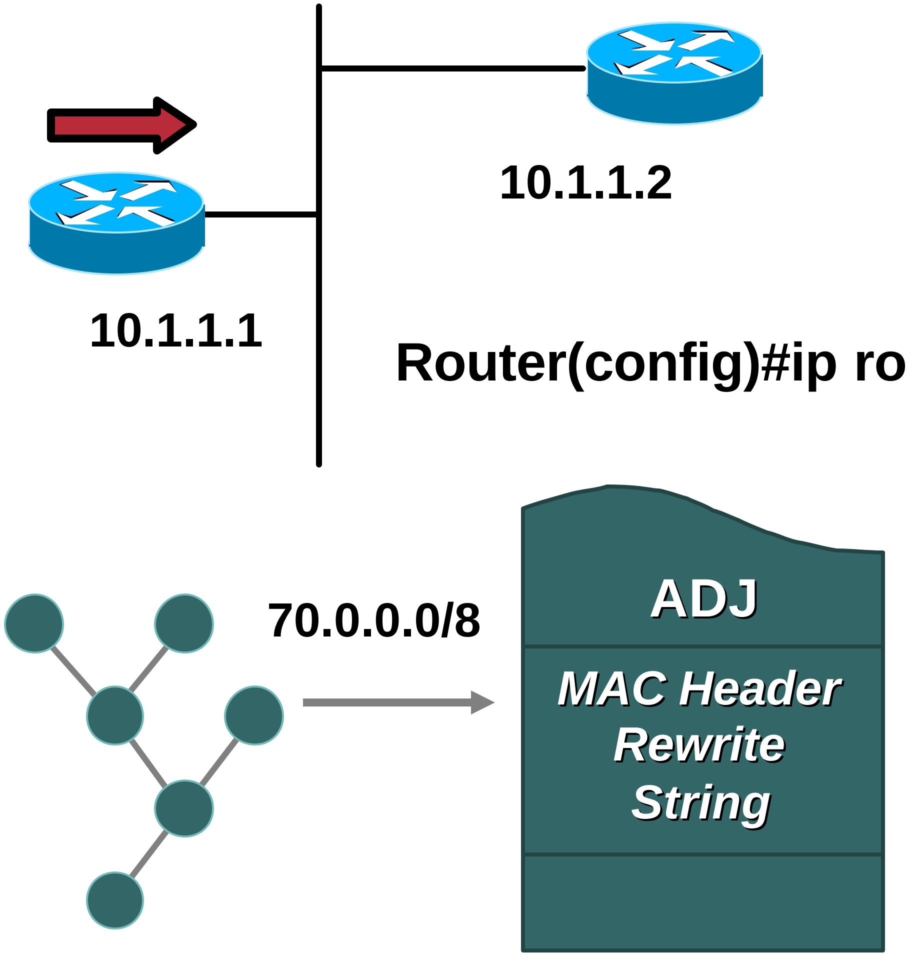
- Auto adjacencies
- Punt Adjacencies
- Glean Adjacency
- Drop Adjacencies
- Discard Adjacencies
- Null Adjacencies
- Cached Adjacencies

Adjacency Table Entries (Auto)

- **Auto Adjacencies—The most common type of adjacency; include all the information needed to rewrite the packet header and place the packet in the proper interfaces output queue**

Adjacency Table Entries (Auto)

Cisco.com



Router(config)#ip route 70.0.0.0 255.0.0.0 10.1.1.2

Adjacency Table Entries

- **Punt Adjacencies—A punt adjacency indicates that the packet should be switched by the next slower switching scheme**

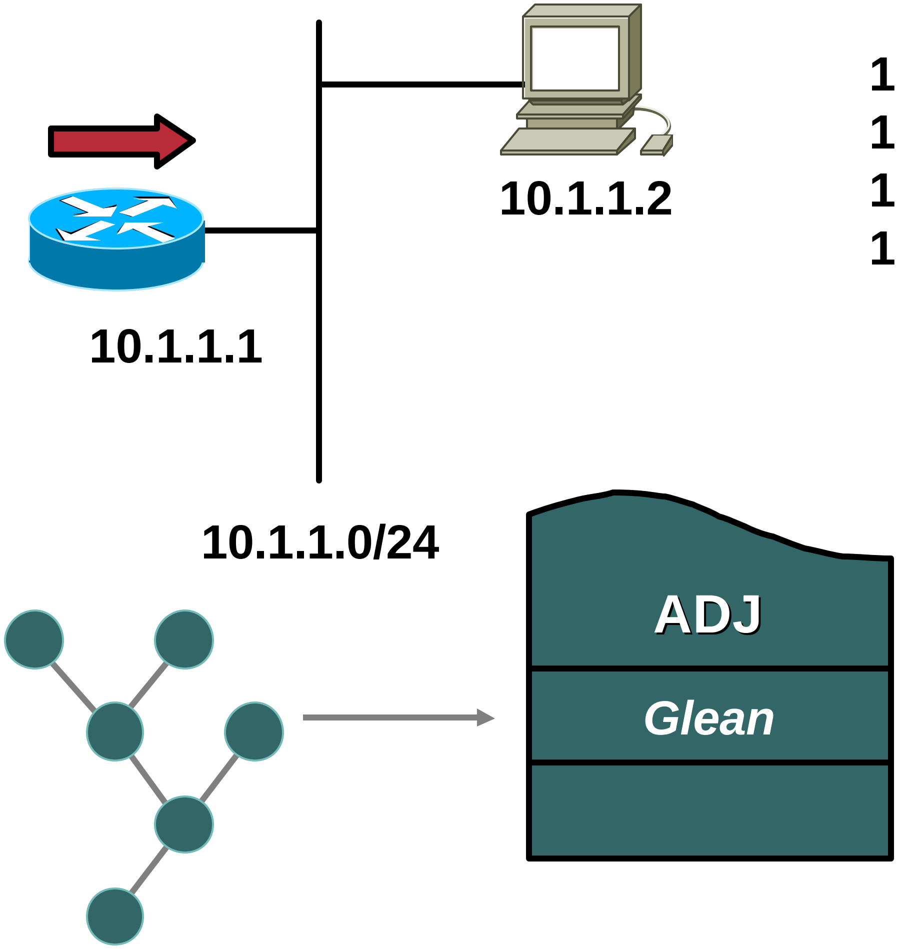
Adjacency Table Entries (Glean)

- **Glean Adjacency—Only one per router; indicates that the destination is attached to the router but the layer two information has not been acquired; results in an ARP request when a packet is switched to this destination**

Adjacency Table Entries (Glean)

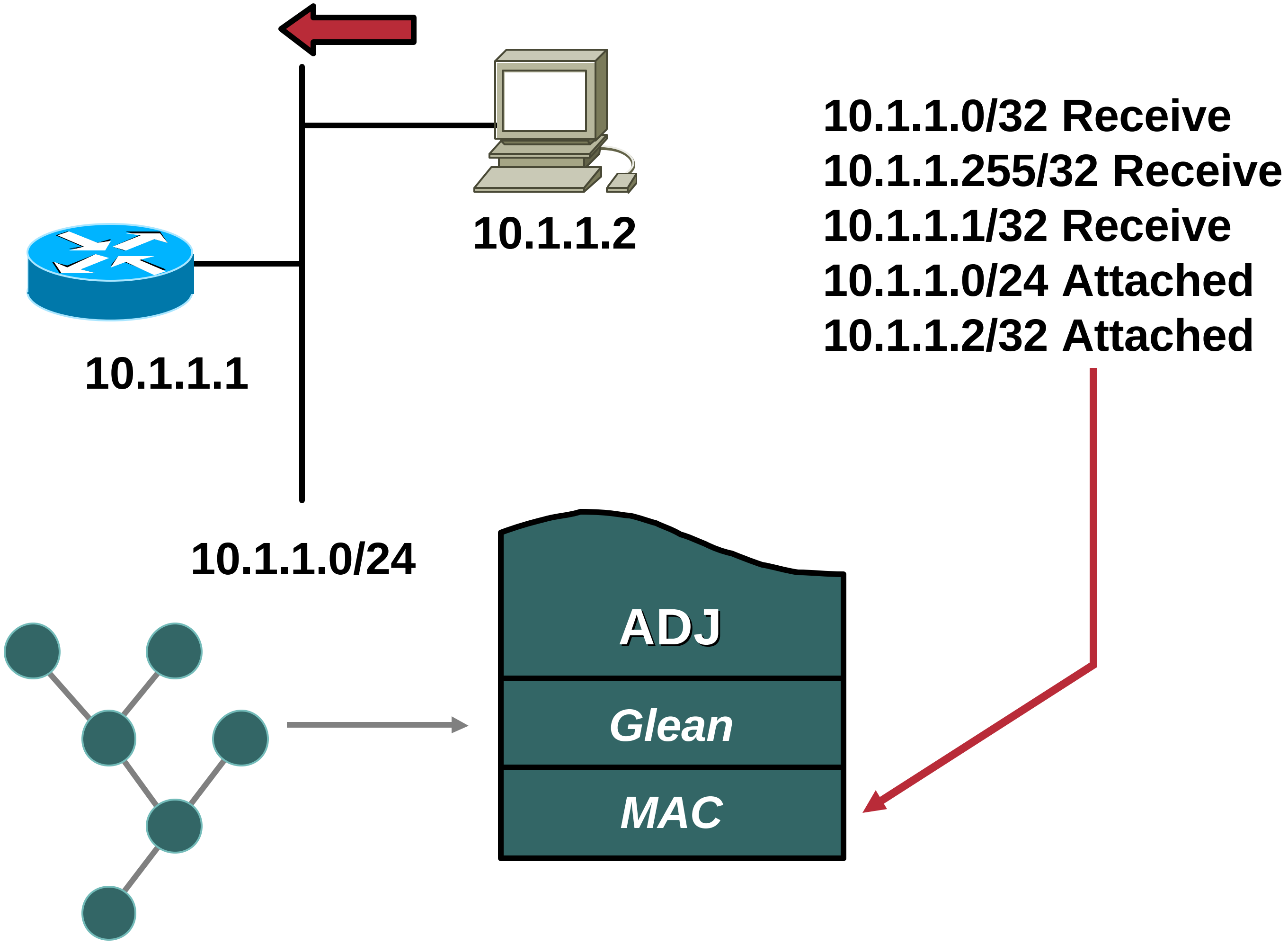
```
Router#sh ip interface brief
Interface        IP-Address    OK? Method   Status   Protocol
Ethernet0/0      20.0.0.1      YES  manual   up       up

Router#sh ip cef adjacency glean
Prefix           Next Hop           Interface
20.0.0.0/8       attached           Ethernet0/0
```

Adjacency Table Entries (Glean)



10.1.1.0/32 Receive
10.1.1.255/32 Receive
10.1.1.1/32 Receive
10.1.1.0/24 Attached

Adjacency Table Entries (Glean)



10.1.1.0/32 Receive
10.1.1.255/32 Receive
10.1.1.1/32 Receive
10.1.1.0/24 Attached
10.1.1.2/32 Attached

Adjacency Table Entries

- **Drop Adjacency—Indicates the packet should be dropped**

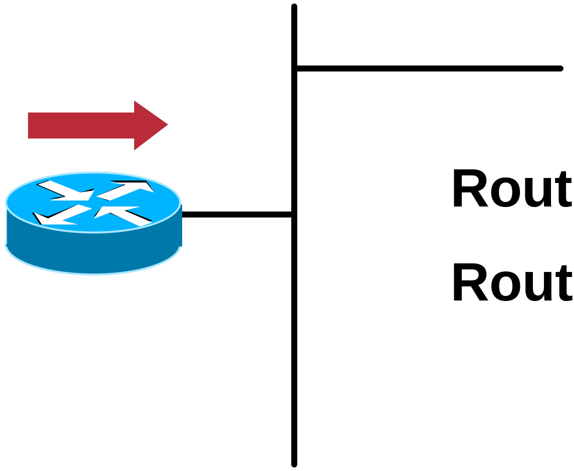
Adjacency Table Entries (Drop)

```
Router#sh ip cef adjacency drop
Prefix          Next Hop    Interface
224.0.0.0/4     drop
```

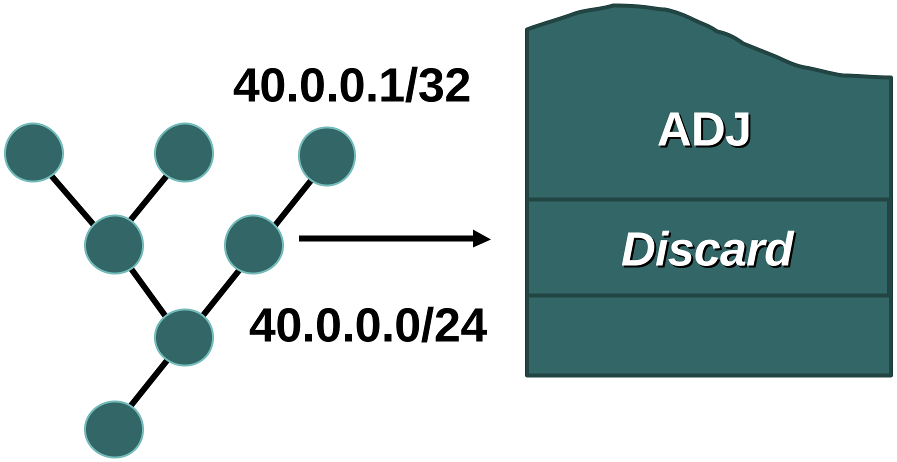
Adjacency Table Entries

- **Discard Adjacency—Indicates destinations which are part of a loopback's subnet, but are not the actual ip address configured on the interface**

Adjacency Table Entries (Discard)



```
Router(config)#int loop0
Router(config-if)#ip addr 40.0.0.1 255.255.255.0
```



```
Router#sh ip cef 40.0.0.2
40.0.0.0/24, version 3, attached, connected
0 packets, 0 bytes
  via Loopback0, 0 dependencies
    valid discard adjacency
```

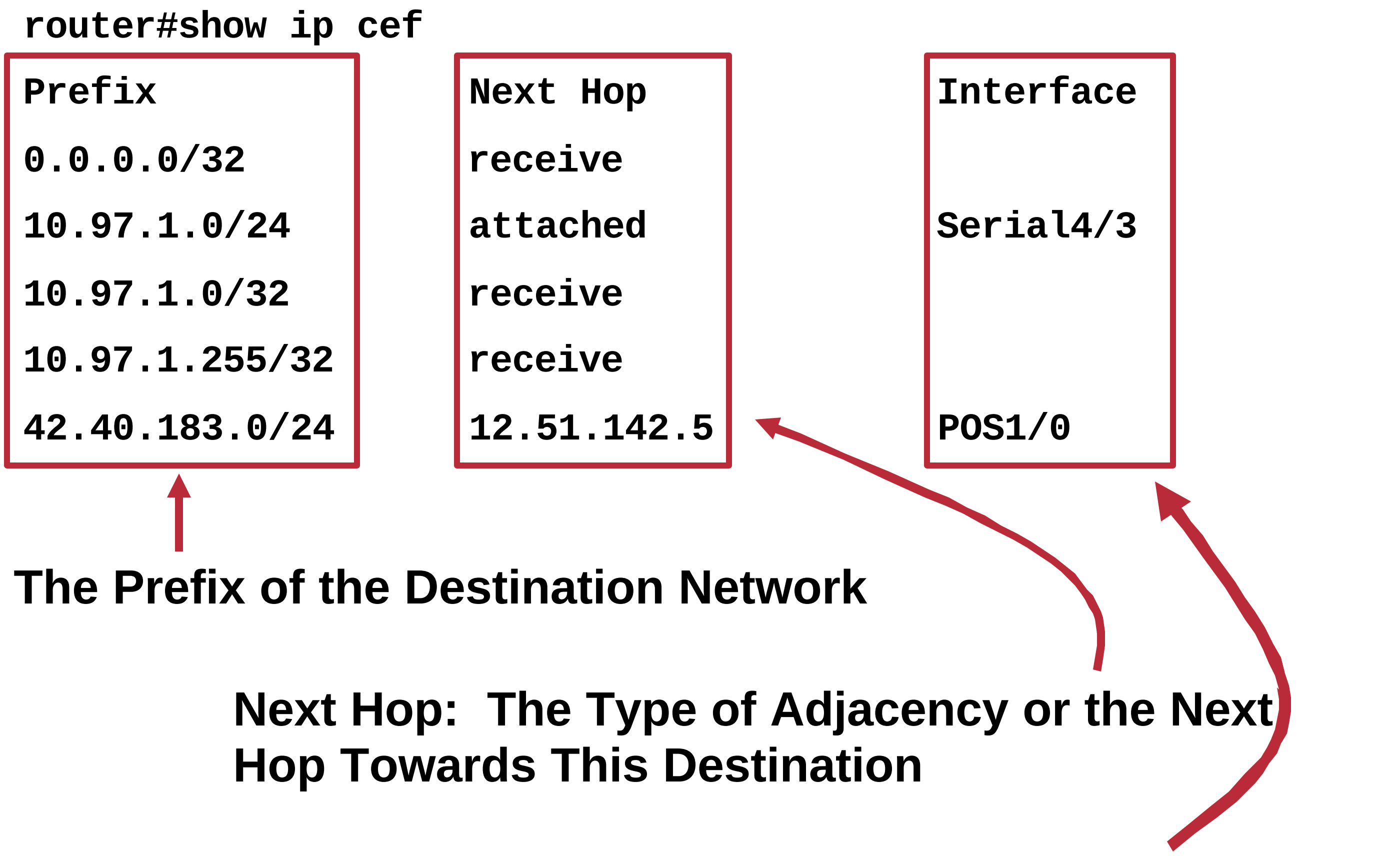
Adjacency Table Entries

- **Null Adjacency—Indicates the packet should be switched to a Null interface on the router**

Adjacency Table Entries (Null)

```
Router(config)#ip route 60.0.0.0 255.0.0.0 null0
Router#sh ip cef adjacency null
Prefix           Next Hop          Interface
60.0.0.0/8       attached          Null0
```

CEF Show Commands



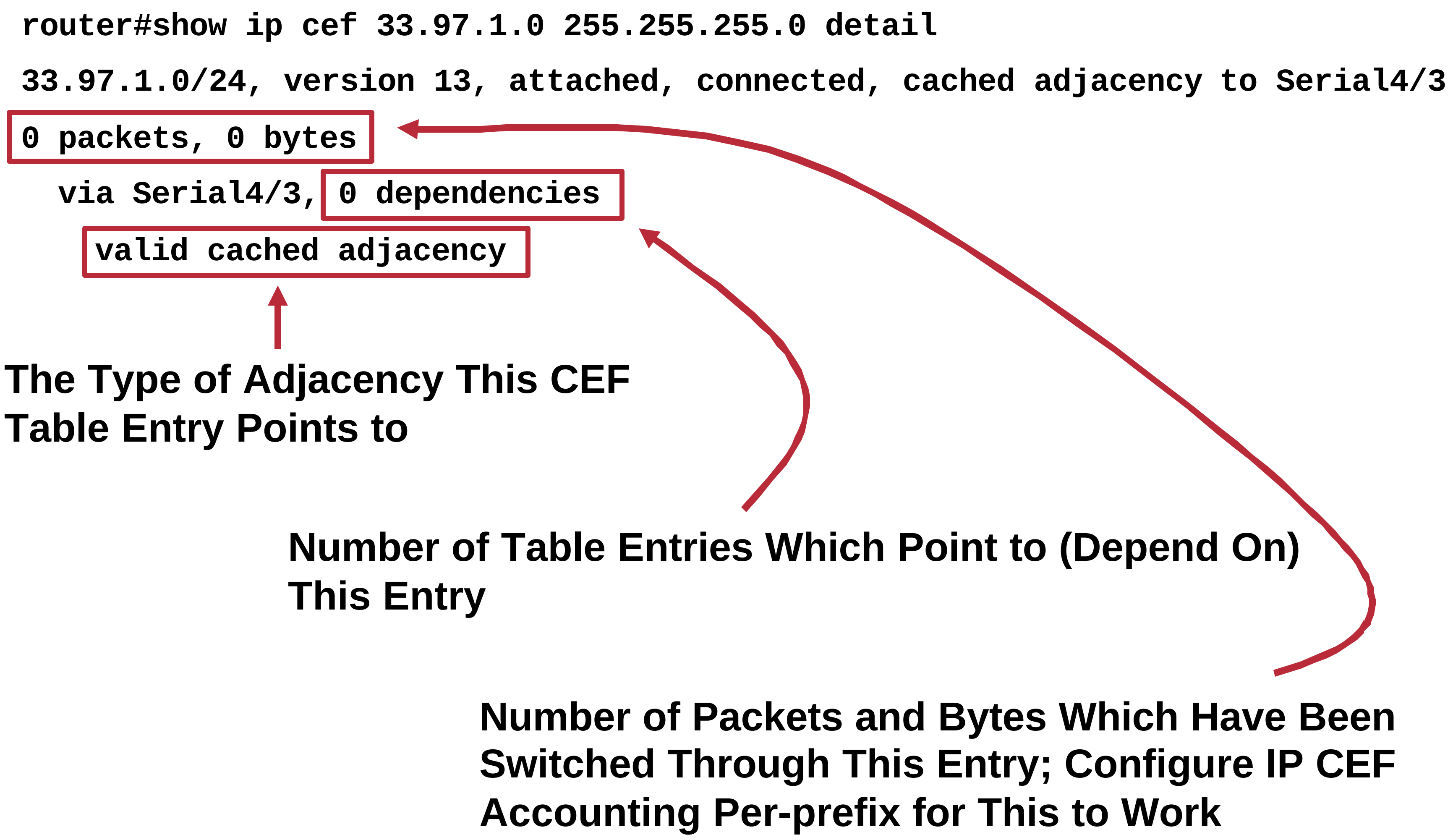
```
router#show ip cef
Prefix            Next Hop         Interface
0.0.0.0/32        receive
10.97.1.0/24      attached         Serial4/3
10.97.1.0/32      receive
10.97.1.255/32    receive
42.40.183.0/24    12.51.142.5      POS1/0
```

Prefix: The Prefix of the Destination Network

Next Hop: The Type of Adjacency or the Next Hop Towards This Destination

Interface: The Interface Out Which to Send Traffic for This Destination

CEF Show Commands



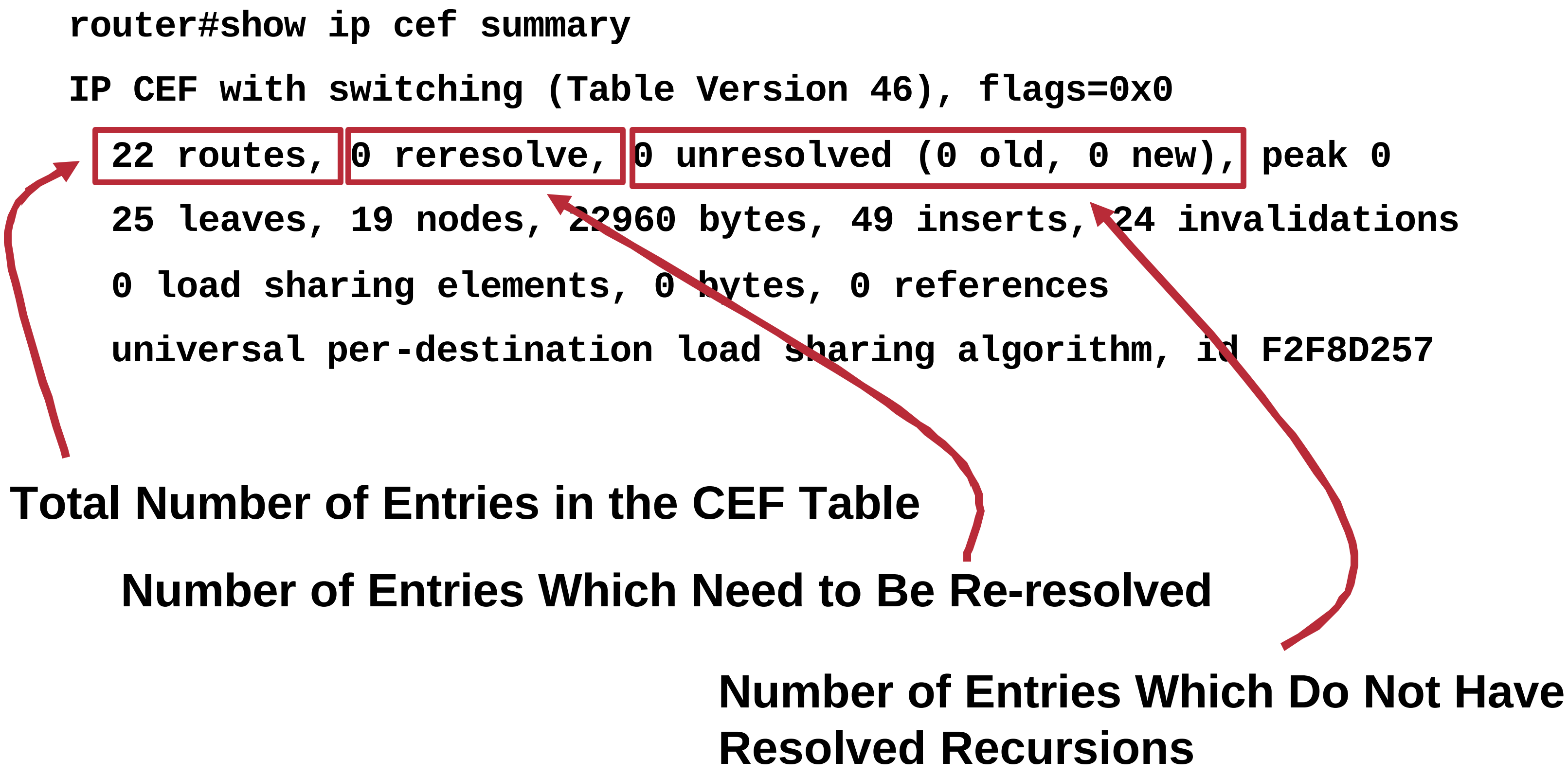
```
router#show ip cef 33.97.1.0 255.255.255.0 detail
33.97.1.0/24, version 13, attached, connected, cached adjacency to Serial4/3
0 packets, 0 bytes
  via Serial4/3, 0 dependencies
    valid cached adjacency
```

The Type of Adjacency This CEF Table Entry Points to

Number of Table Entries Which Point to (Depend On) This Entry

Number of Packets and Bytes Which Have Been Switched Through This Entry; Configure IP CEF Accounting Per-prefix for This to Work

CEF Show Commands



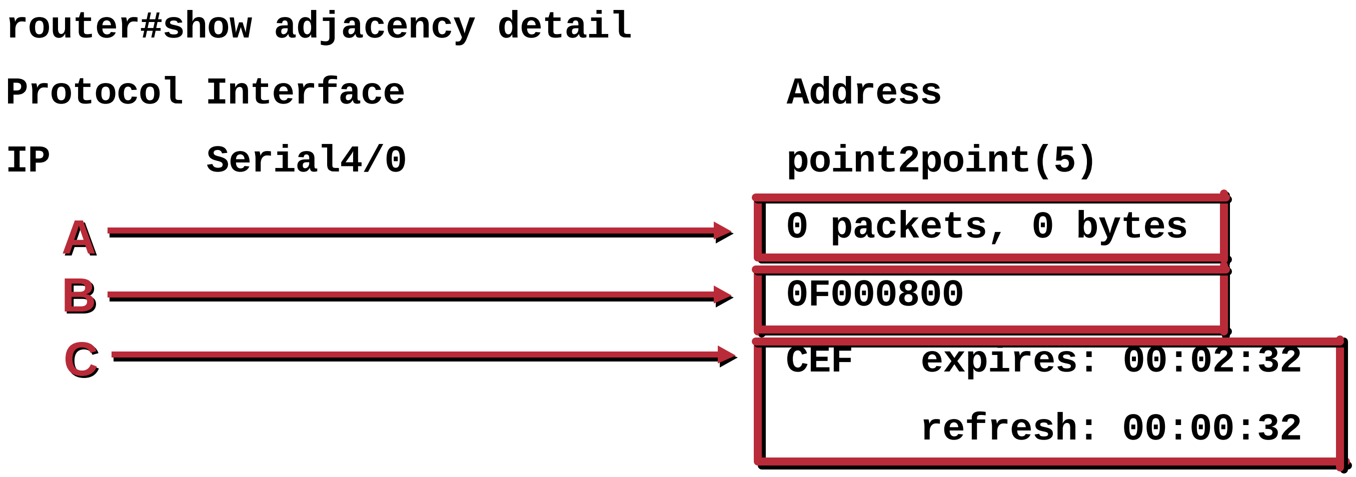
```
router#show ip cef summary
IP CEF with switching (Table Version 46), flags=0x0
  22 routes, 0 reresolve, 0 unresolved (0 old, 0 new), peak 0
  25 leaves, 19 nodes, 22960 bytes, 49 inserts, 24 invalidations
  0 load sharing elements, 0 bytes, 0 references
  universal per-destination load sharing algorithm, id F2F8D257
```

Total Number of Entries in the CEF Table

Number of Entries Which Need to Be Re-resolved

Number of Entries Which Do Not Have Resolved Recursions

CEF Show Commands



```
router#show adjacency detail
Protocol Interface                 Address
IP       Serial4/0                 point2point(5)
                                   0 packets, 0 bytes
                                   0F000800
                                   CEF    expires: 00:02:32
                                          refresh: 00:00:32
```

A: Packets and Bytes Switched Through This Adjacency

B: MAC Header Rewrite String

C: When This Entry Will Be Refreshed; In This Case, All Point2Points Are Refreshed Every Minute

CEF Show Commands

```
router#show int ethernet1/0 stat
Ethernet1/0
          Switching path    Pkts In   Chars In   Pkts Out  Chars Out
               Processor     977121   70149655     578014   56457133
             Route cache          0          0          0          0
                   Total     977121   70149655     578014   56457133
```

Route cache Includes CEF Switched Packets

Router Architectures & Parallel Express Forwarding

Introduction

- Routers have to deal in three “Planes” of operation:-

  The “*Control*” Plane

    Building and maintaining data structures such as “forwarding tables”

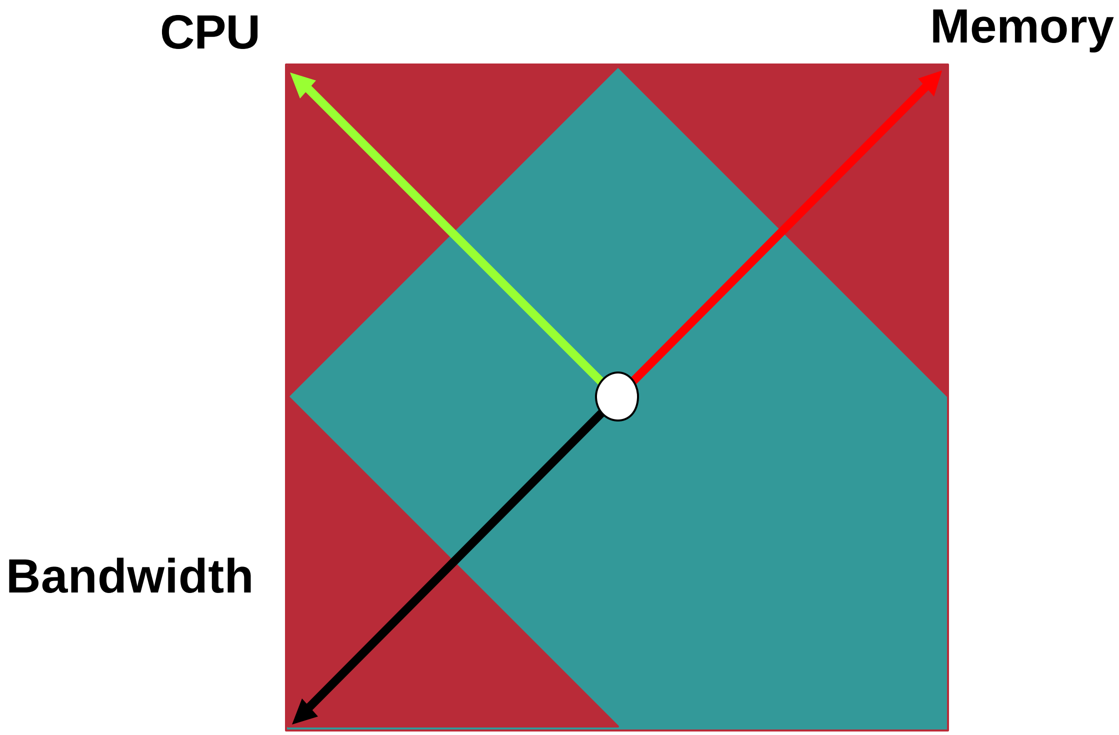
  The “*Management*” Plane

    Dealing with configuration files, gathering and providing statistics, providing and responding to control protocol messages

  The “*Data*” Plane

    Switching of packets, manipulation of packet (header and content), packet delivery scheduling (queuing)

Introduction – Consumable Resources



When any all or all of the resources are exhausted, inconsistent behavior will be observed

Routers Operationally

- Maintain/manipulate routing information
  - Listen for updates/update neighbors
- Classify packets for manipulation/queuing/permit-deny, etc.
  - Compare packets to classification lists and perform control
- Perform Layer 3 switching
  - Create outbound Layer 2 encapsulation
  - Layer 3 checksum
  - TTL/hop count update
- Management/billing (statistics)
  - Interface statistics—NetFlow export
  - Telnet, SNMP, ping, trace route, HTTP

Routers Functionally

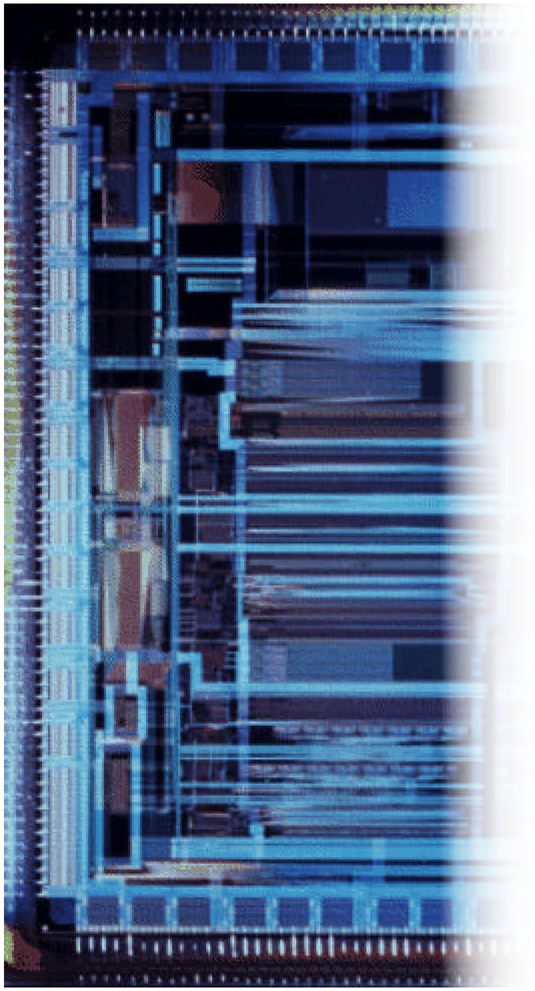
- (Attempt to) switch packets
  - Layer 3 switching based on routing information
- (Attempt to) transmit packets
  - Access outbound media
- Manipulate packets
  - Change contents of packet (CAR/NAT/compression/encryption)
- Consume packets
  - Routing protocol updates etc…/services advertisements(SAP)/ICMP/SNMP
- Generate packets
  - Routing protocol packets/SAPs/ICMP/SNMP
  - Tunnels—GRE, IPSec, DLSw etc…

Router Hardware

- Interface Processors
- The Central Processing Unit
- Memory
- The Backplane

The Central Processing Unit

- Provides horsepower for all control plane functions, such as system maintenance, building routing tables, etc.
- On some platforms, it also provides the horsepower for actually switching packets

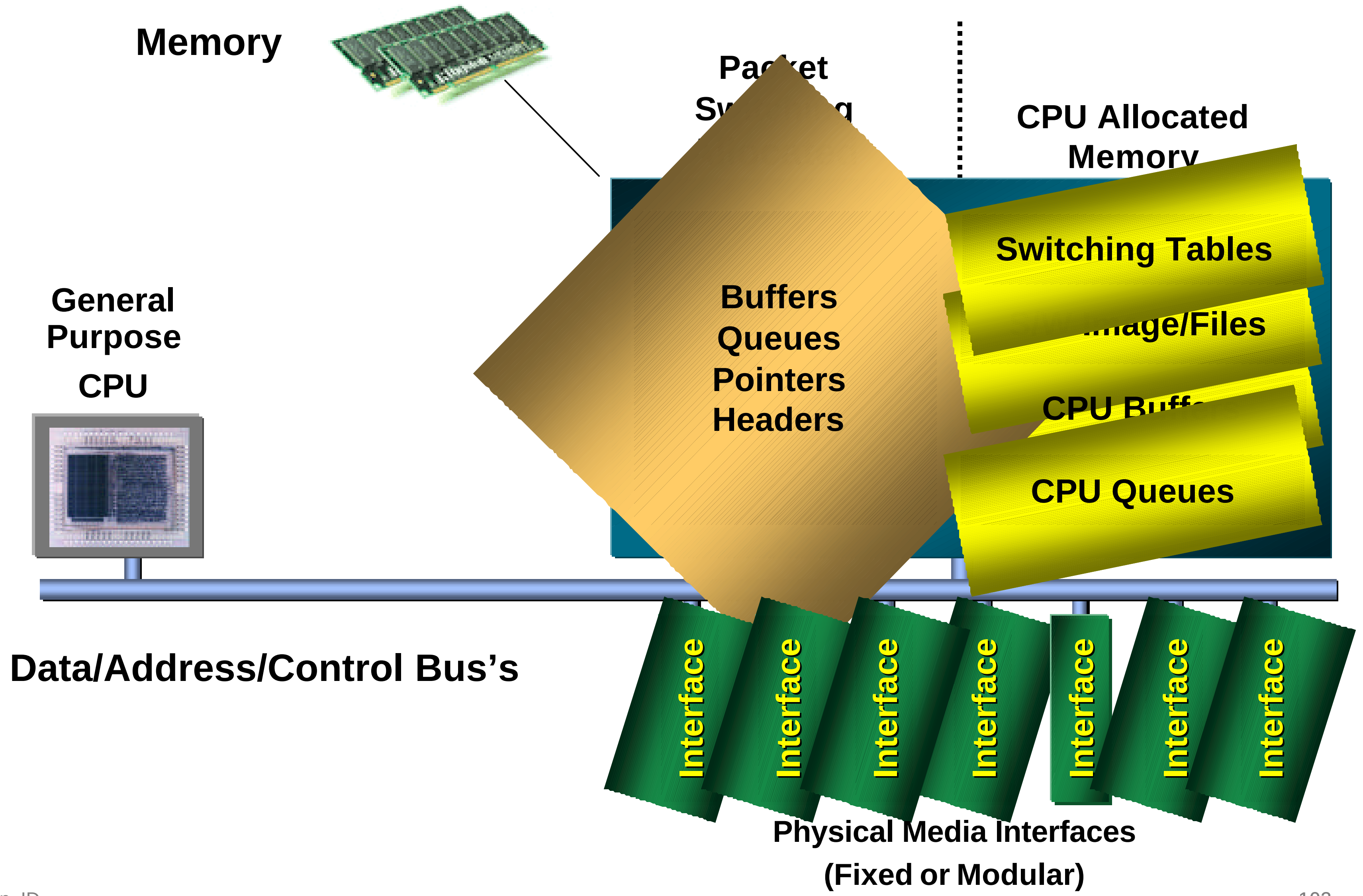


Shared Memory Architecture

- **Applicable Platforms**

  Cisco 1xxx
  Cisco 2xxx
  Cisco 3xxx
  Cisco 4xxx

Shared Memory Architecture

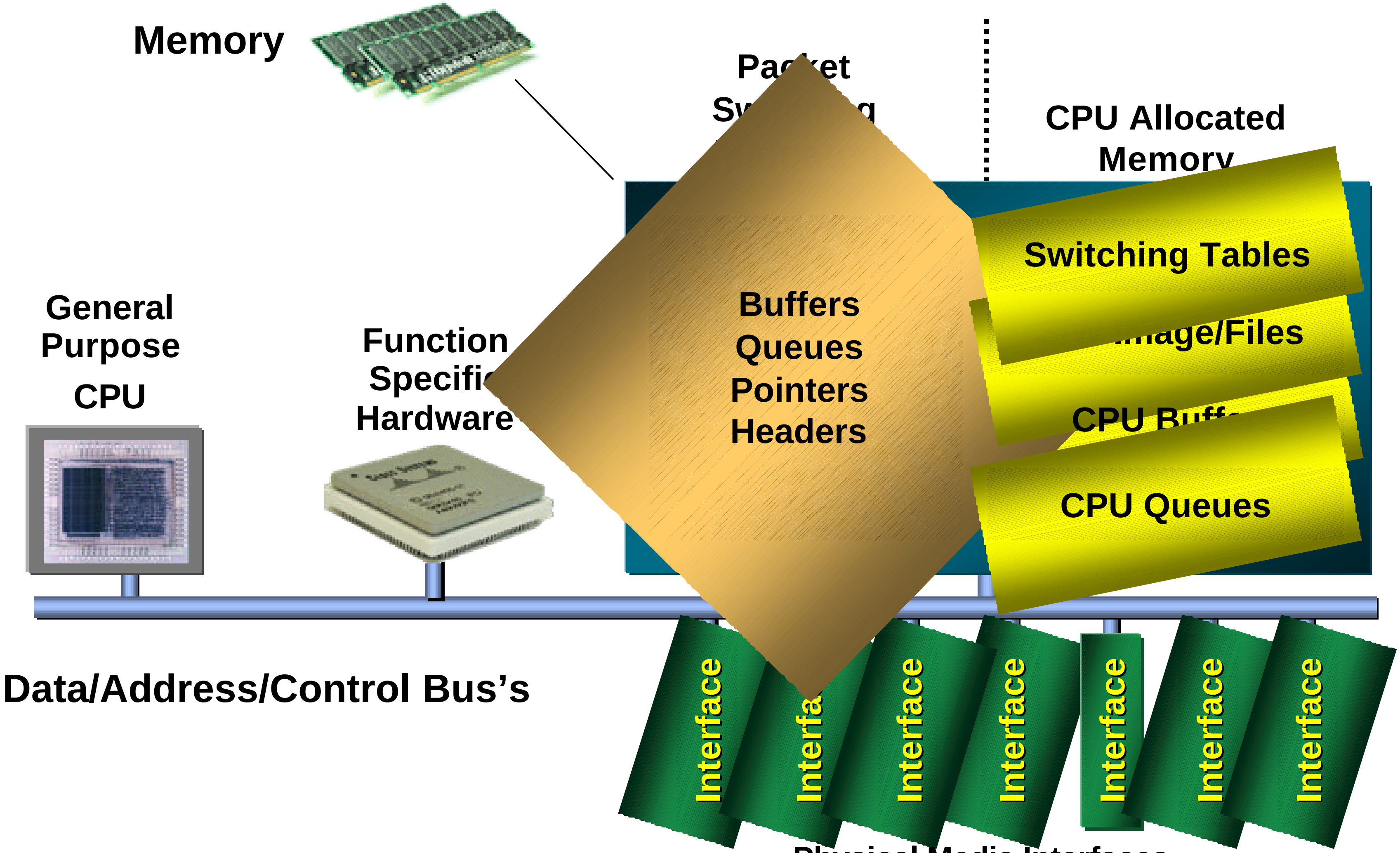


Shared Memory Architecture (Hardware “Assist”)

- **Applicable Platforms**

  Cisco 7200
  Cisco 7300
  Cisco 7400
  Cisco 10000

Shared Memory Architecture (Hardware "Assist")



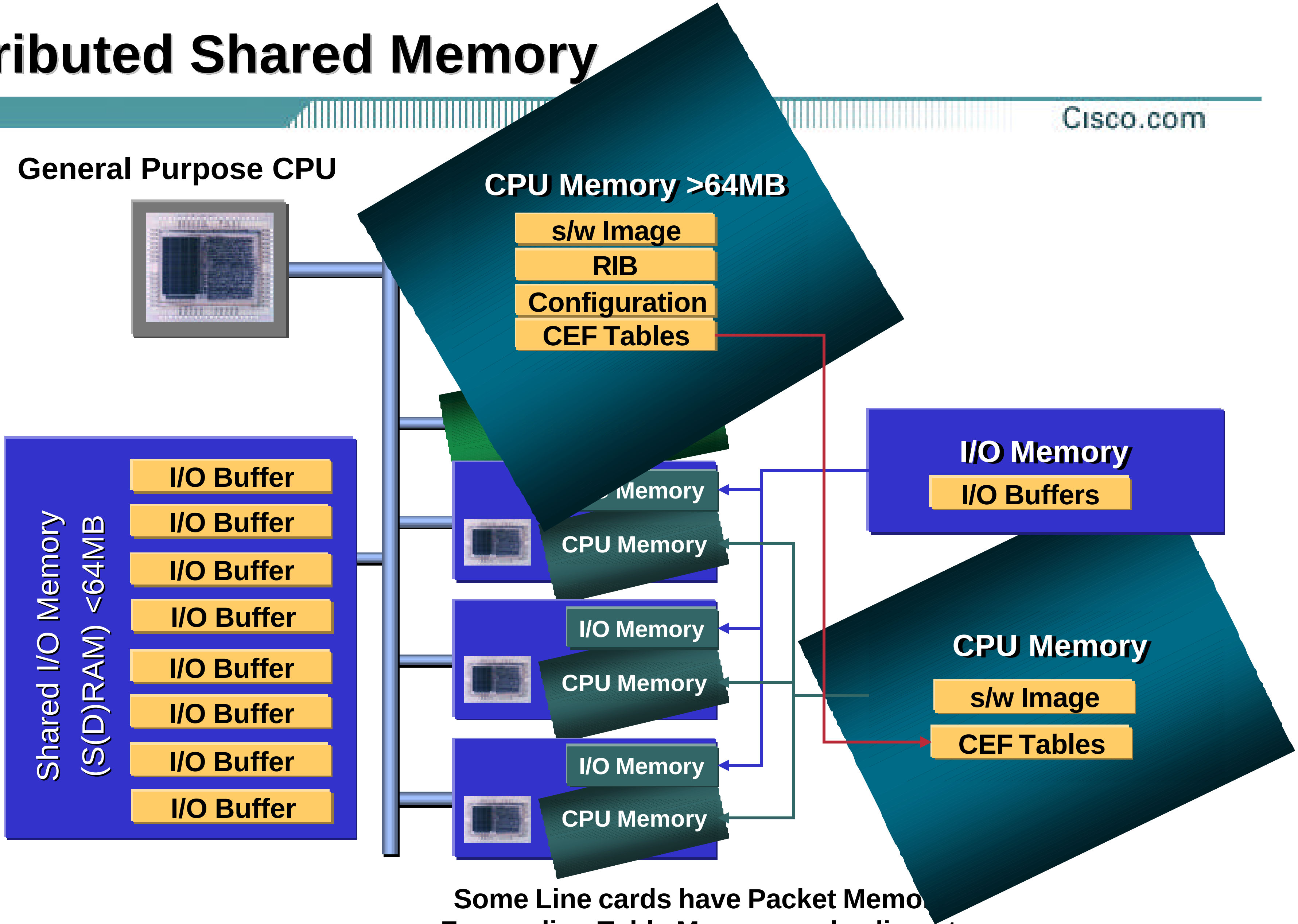
Physical Media Interfaces
(Fixed or Modular)

Distributed Shared Memory

- **Applicable Platforms**

  Cisco 7500
  Catalyst 5xxx RSM

Distributed Shared Memory



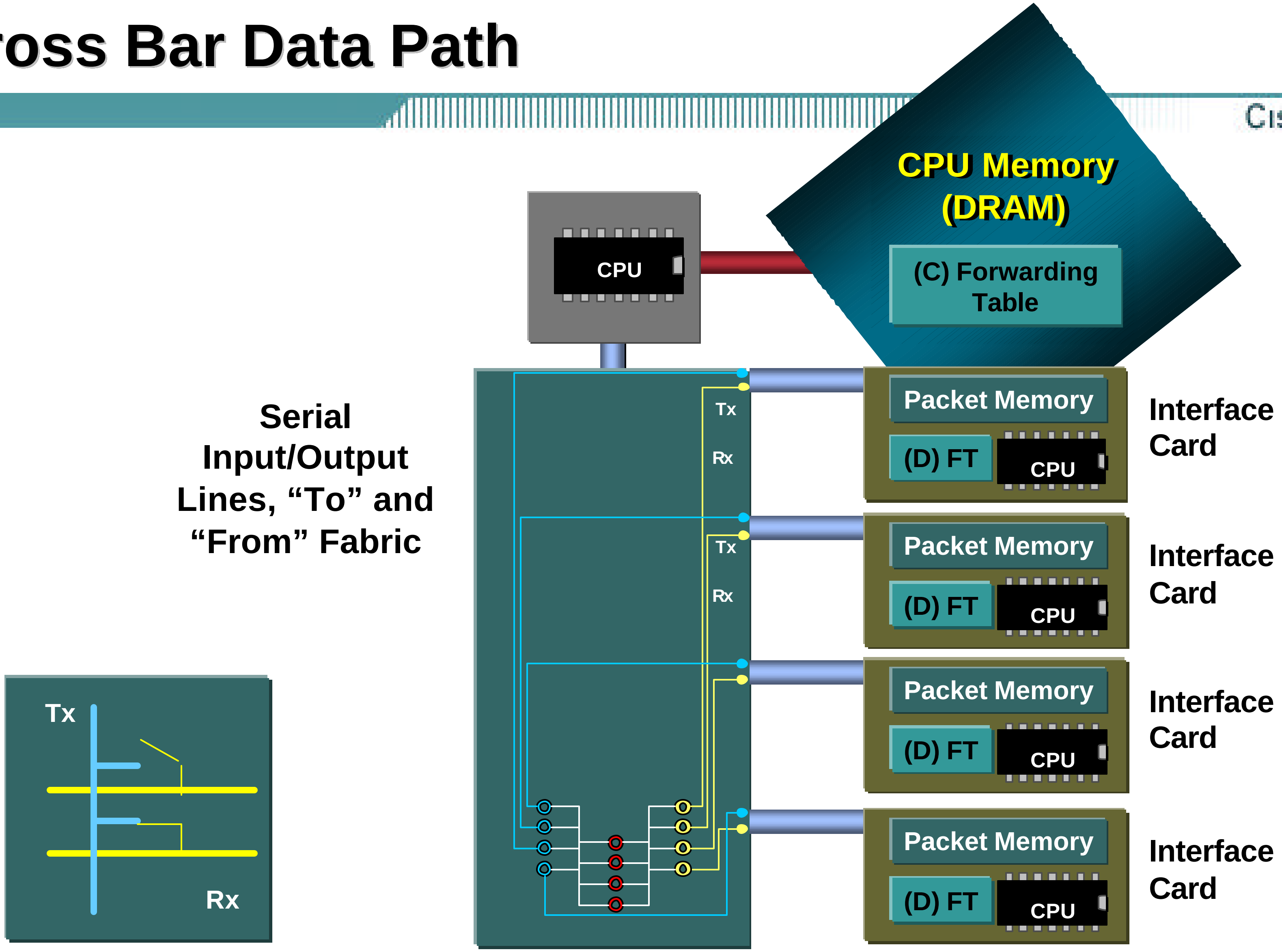
Some Line cards have Packet Memo
Forwarding Table Memory and a discrete
Switching Hardware.

Distributed Cross Bar

- **Applicable Platforms**

  Cisco 6500/7600 OSR
  Cisco 12000 (GSR)

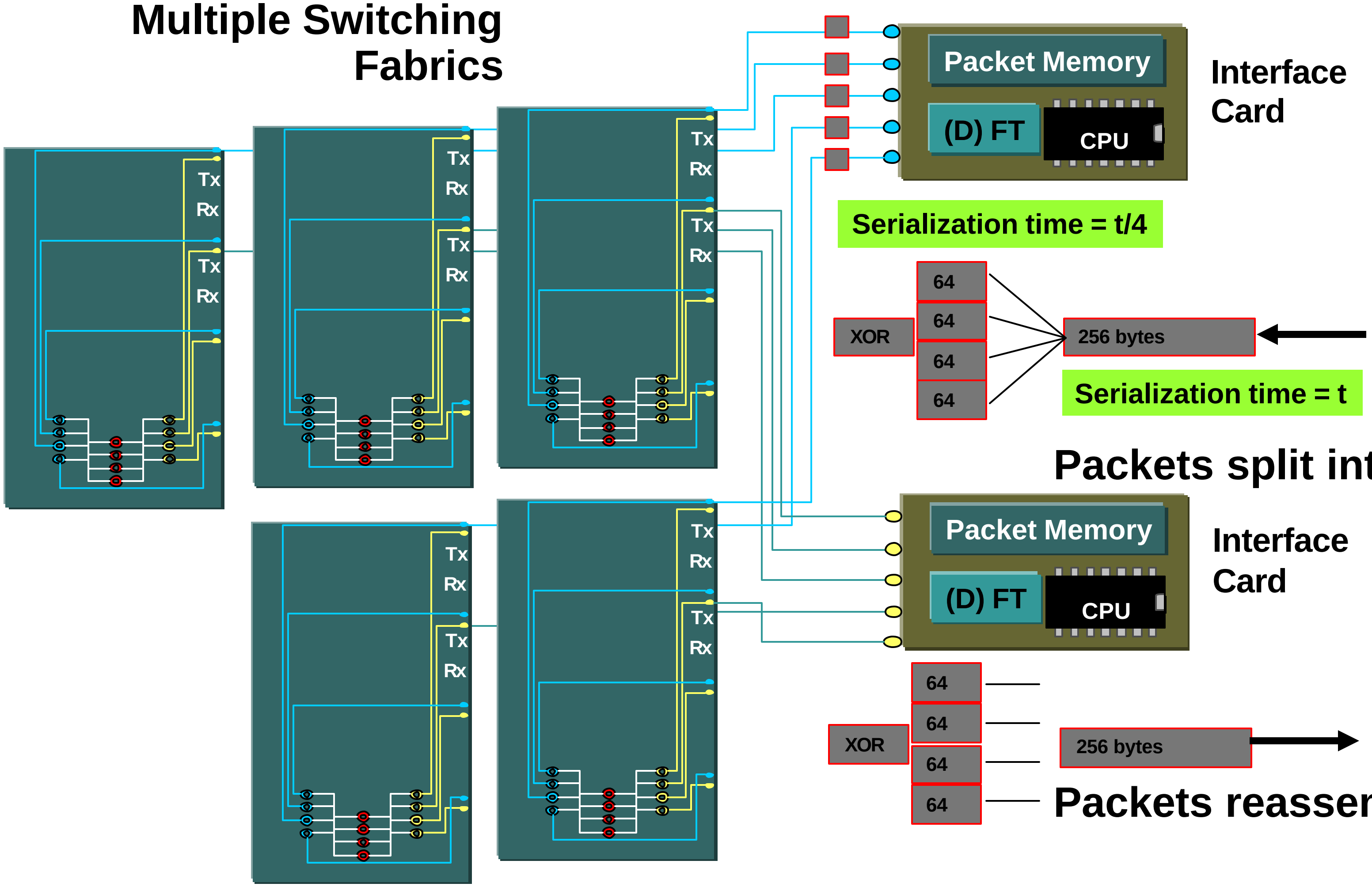
Cross Bar Data Path



ASIC X-Bar Fabric

Cross Bar Data Path (Multiple Fabrics)

Packet Slicing Allows Multiple Switching Fabrics



Interface Card

Serialization time = t/4

Serialization time = t

Packets split into cells

Interface Card

Packets reassembled

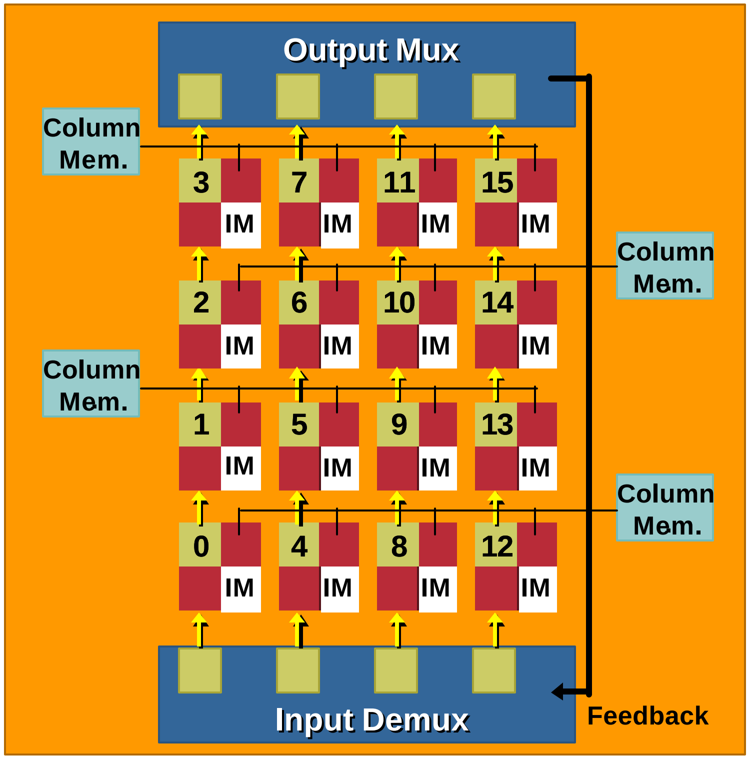
Parallel Express Forwarding

- PXF is one kind of “Function Specific Hardware”
- PXF Architecture
- PXF packet switching

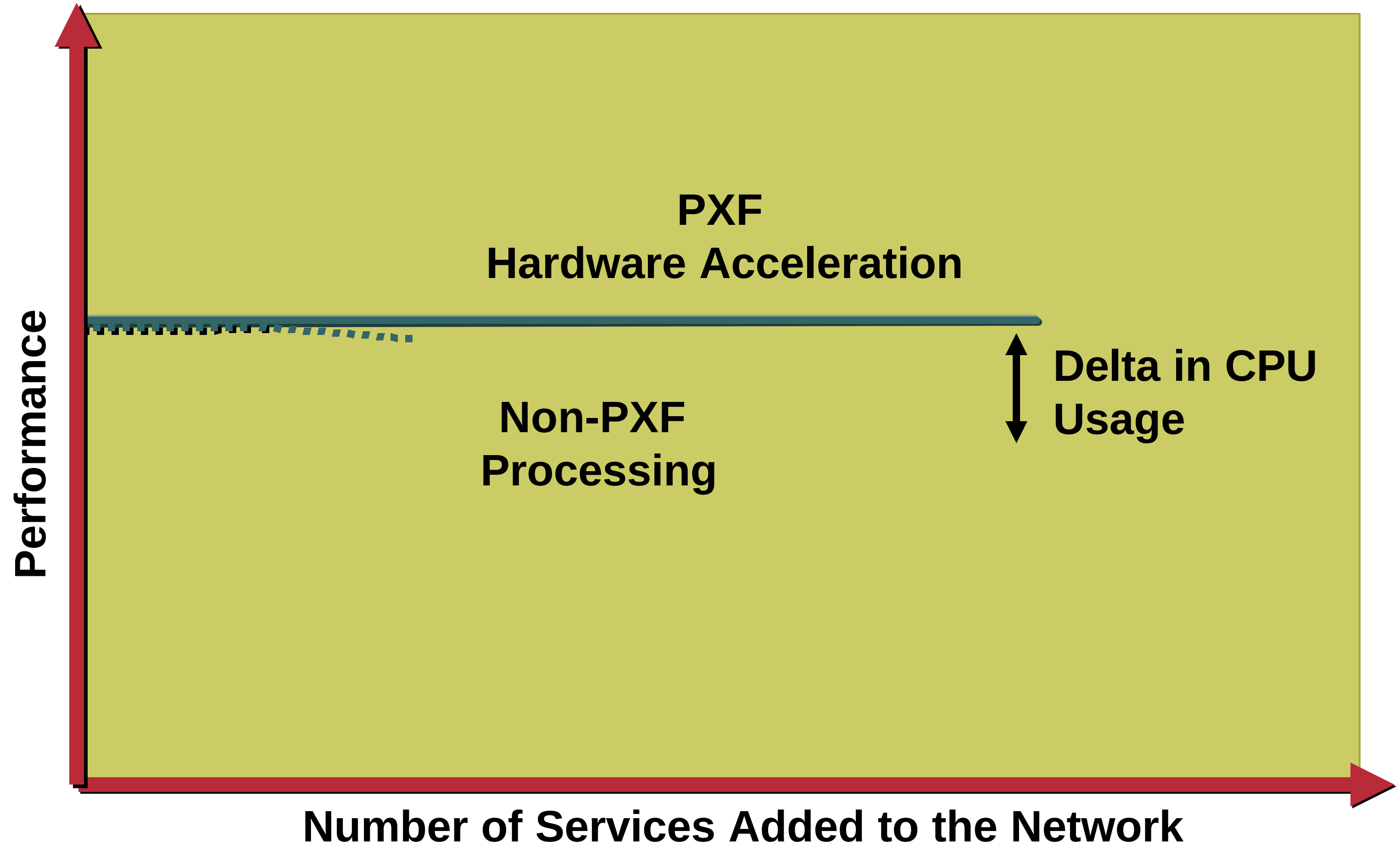
Cisco-Developed Unique Value-Add
PXF IP Services Processor

- Internally-Developed Cisco Processing Technology

US Patent 6,101,599



Benefit of PXF Acceleration

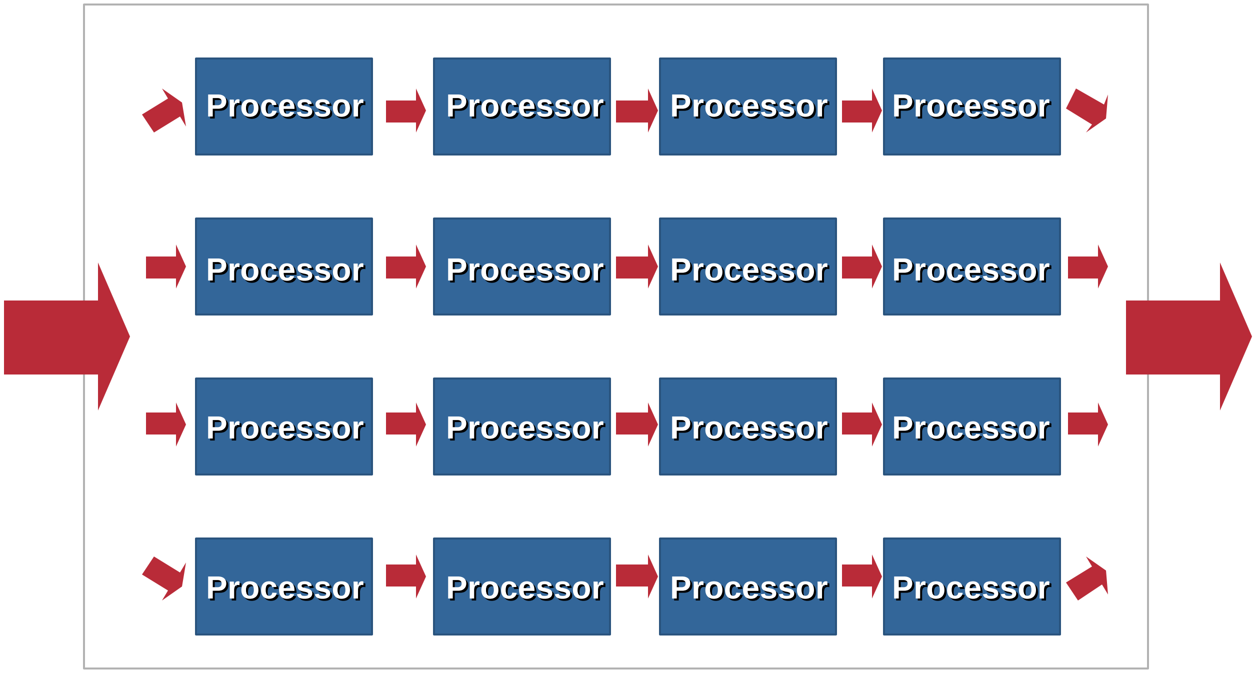


Parallel eXpress Forwarding (PXF) Engine

- New technology switching engine for high-touch L3 services with optimized throughput
- Programmable architecture to allow for future feature upgrades
- Based on custom pipelined array processor (ASIC)

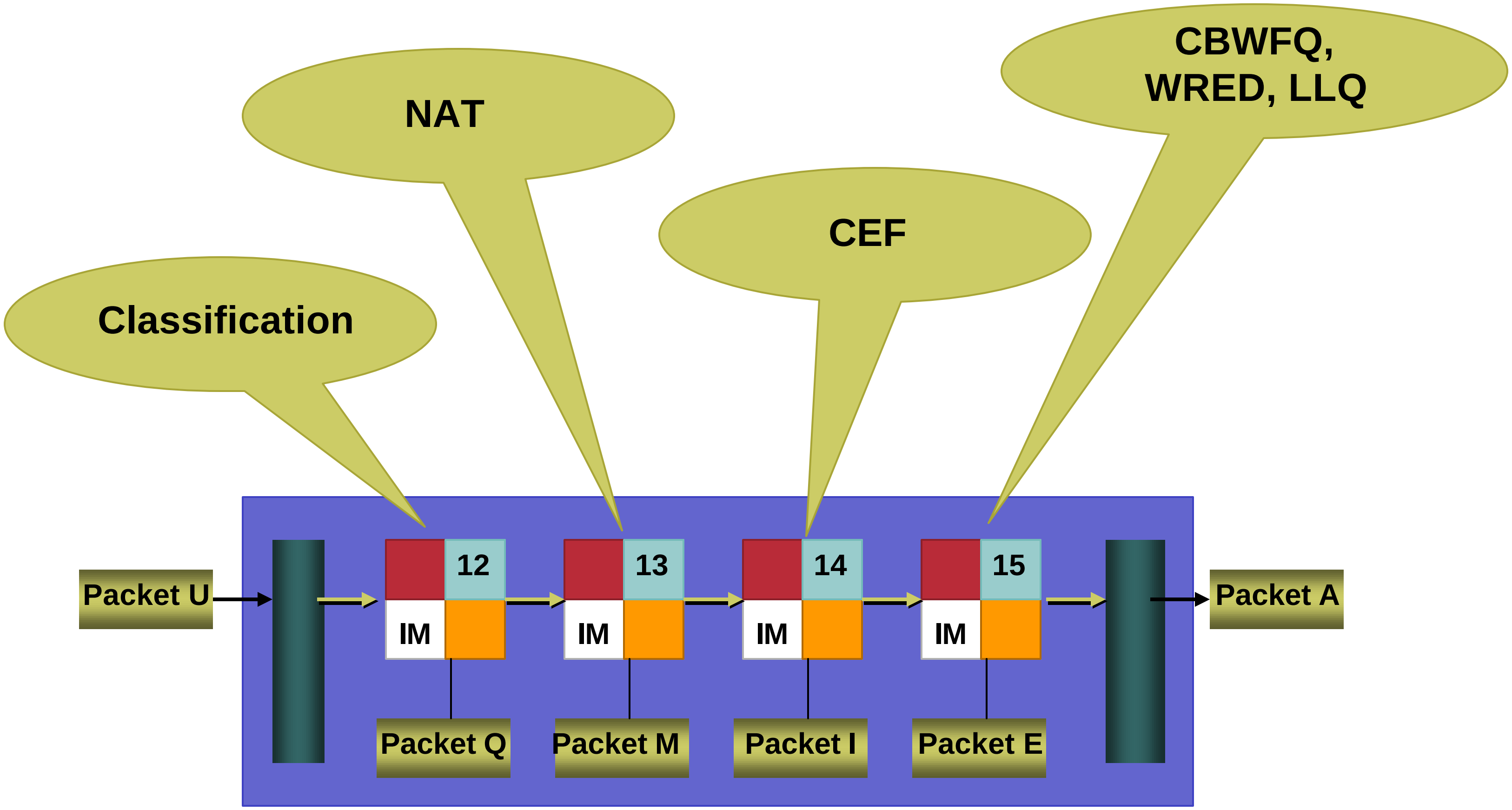
Power of Cisco Parallel Processing

PXF Network Processor



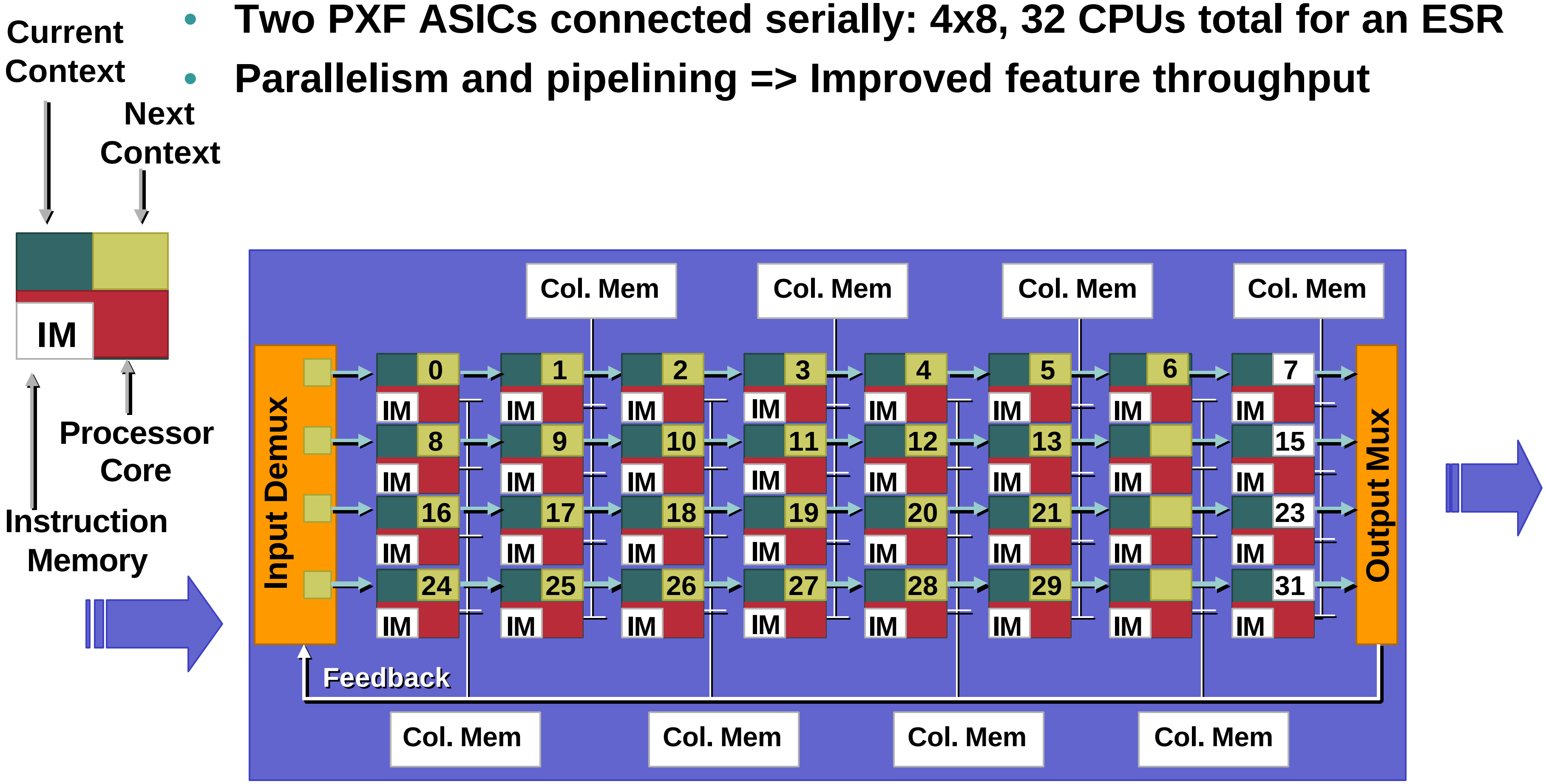
- **Matrix of separate processors**
- **Implements “assembly line” for exceptional performance**
- **“Assembly line” enables consistent throughout**
- **Little division when services are enabled/disabled**

PXF Processor Services
Example

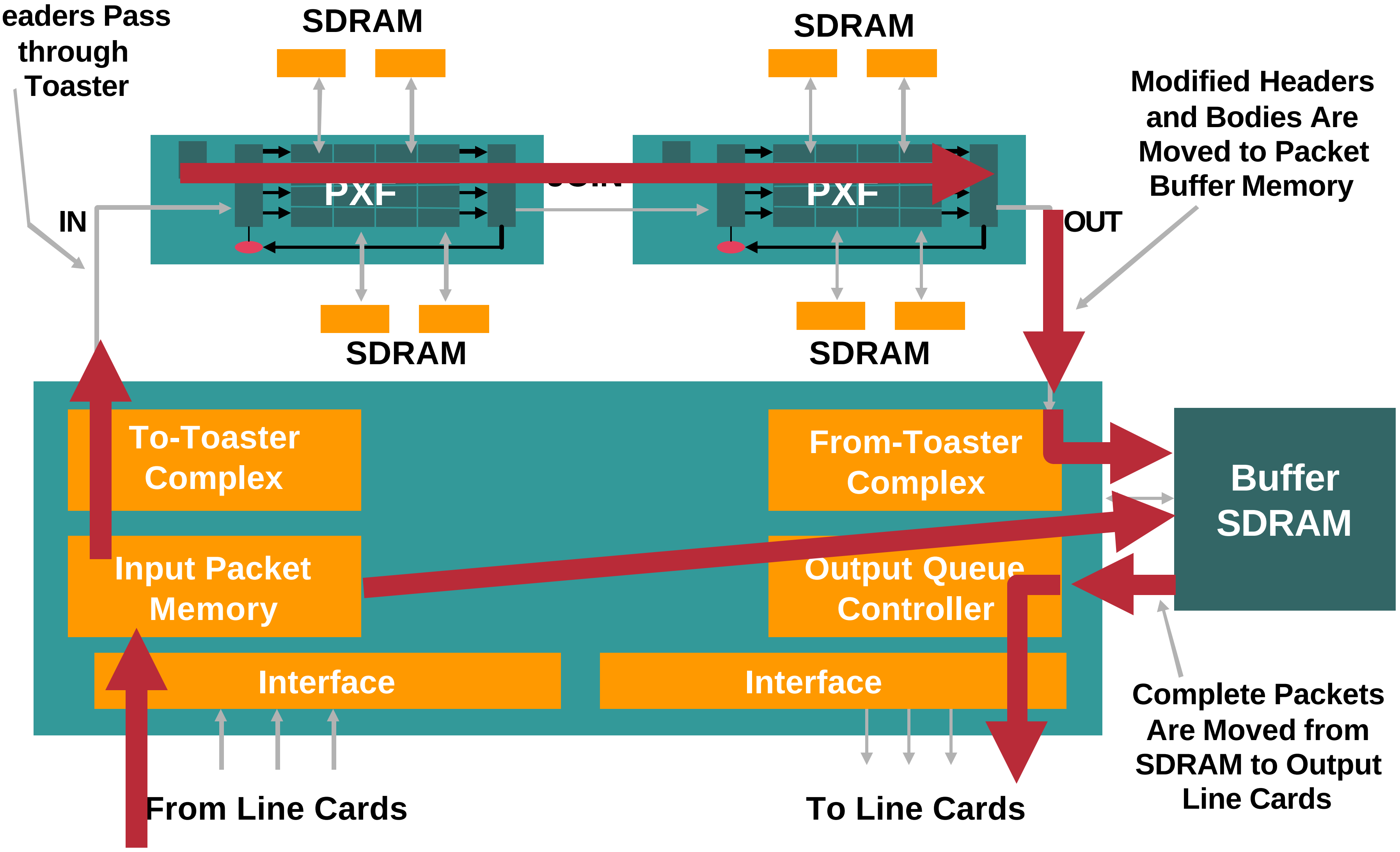


Parallel eXpress Forwarding (PXF)

- Each PXF ASIC has 16 processors arranged in 4 rows x 4 columns
- Two PXF ASICs connected serially: 4x8, 32 CPUs total for an ESR
- Parallelism and pipelining => Improved feature throughput



PXF Packet Forwarding



Headers Pass through Toaster

Modified Headers and Bodies Are Moved to Packet Buffer Memory

Complete Packets Are Moved from SDRAM to Output Line Cards

Summary

Summary

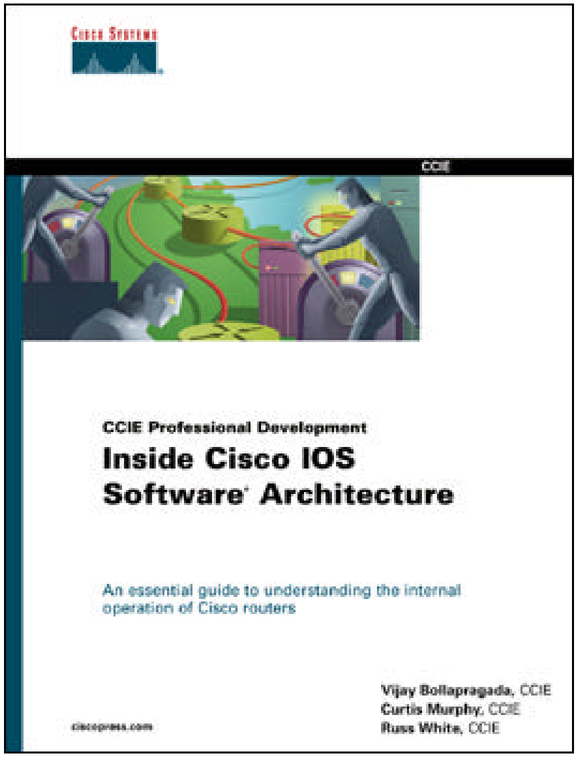
- 90 minutes is way too much to summarize in one slide, and not enough time to cover these topics!
- Routers scale based on CPU, processing hardware, memory and bandwidth
- Resource exhaustion results in dropped packets
- No one architecture has all the answers different platforms are appropriate for different roles in your network

Recommended Reading

Inside Cisco IOS Software Architecture
(CCIE Professional Development)
ISBN: 1-57870-181-3

IP Routing Fundamentals
ISBN: 1-57870-071-X

Cisco Router Configuration,
Second Edition
ISBN: 1-57870-241-0



CCIE

CCIE Professional Development

Inside Cisco IOS Software Architecture

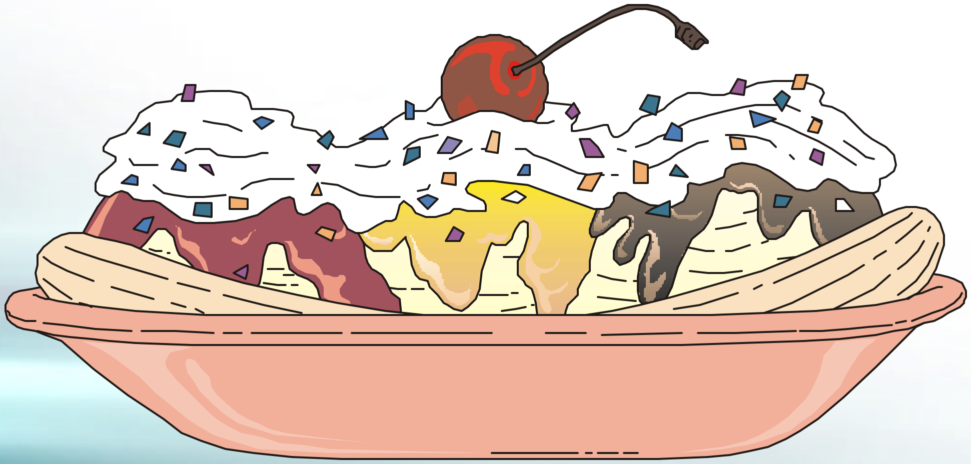
An essential guide to understanding the internal operation of Cisco routers

ciscopress.com

Vijay Bollapragada, CCIE
Curtis Murphy, CCIE
Russ White, CCIE

CEF Whitepaper:
http://www.cisco.com/warp/public/732/Tech/switching/docs/cef_ov_final.pdf

Life Is Short, Eat Dessert First!



CISCO SYSTEMS ®

EMPOWERING THE
INTERNET GENERATION