What You Make Possible
Session Goal

- To provide you with a thorough understanding of the Cisco Nexus™ 7000 switching architecture, supervisor, fabric, and I/O module design, packet flows, and key forwarding engine functions
- This session will not examine NX-OS software architecture or other Nexus platform architectures

Related sessions:
- BRKDCT-2204 Nexus 7000/5000/2000/1000v Deployment Case Studies
- BRKIPM-3062 Nexus Multicast Design Best Practices
- BRKDCT-2121 VDC Design and Implementation
- BRKDCT-2048 Deploying Virtual Port Channel in NX-OS
- BRKARC-3472 NX-OS Routing & Layer 3 Switching
- TECDCT-3297 Operating and Deploying NX-OS
- BRKCRS-3144 Troubleshooting Cisco Nexus 7000 Series Switches
- LTRCRT-5205 Configuring Nexus 7000 Virtualization Lab
- LTRDCT-1142 FabricPath Deployment in the Data Center Lab
What Is Nexus 7000?

Data-center class Ethernet switch designed to deliver high-availability, system scale, usability, investment protection

I/O Modules

Chassis

Supervisor Engine

Fabrics
Agenda

- Chassis Architecture
- Supervisor Engine and I/O Module Architecture
- Forwarding Engine Architecture
- Fabric Architecture
- I/O Module Queuing
- Layer 2 Forwarding
- IP Forwarding
- IP Multicast Forwarding
- Classification
- NetFlow
- Conclusion
Key Chassis Components

- **Common components:**
  - Supervisor Engines
  - I/O Modules
  - Power Supplies

- **Chassis-specific components:**
  - Fabric Modules
  - Fan Trays
Agenda

- Chassis Architecture
- Supervisor Engine and I/O Module Architecture
- Forwarding Engine Architecture
- Fabric Architecture
- I/O Module Queuing
- Layer 2 Forwarding
- IP Forwarding
- IP Multicast Forwarding
- Classification
- NetFlow
- Conclusion
Supervisor Engine 1

- Performs control plane and management functions
  - Dual-core 1.66GHz x86 processor with 8GB DRAM
  - 2MB NVRAM, 2GB internal bootdisk, compact flash slots, USB
- Console, aux, and out-of-band management interfaces
- Interfaces with I/O modules via 1G switched EOBC
- Houses dedicated central arbiter ASIC
  - Controls access to fabric bandwidth via dedicated arbitration path to I/O modules
Nexus 7000 I/O Module Families

M Series and F Series

- M family – L2/L3/L4 with large forwarding tables and rich feature set

- F family – High performance, low latency, low power with streamlined feature set
8-Port 10GE M1 I/O Module
N7K-M108X2-12L

- 8-port 10G with X2 transceivers
- 80G full-duplex fabric connectivity
- Two integrated forwarding engines (120Mpps)
  - Support for “XL” forwarding tables (licensed feature)
- Distributed L3 multicast replication
- 802.1AE LinkSec

Supported in NX-OS release 5.0(2a) and later
8-Port 10G XL M1 I/O Module Architecture
N7K-M108X2-12L

Front Panel Ports

1  2  3  4  5  6  7  8

10G MAC 10G MAC 10G MAC 10G MAC 10G MAC 10G MAC 10G MAC 10G MAC

Replication Engine

Forwarding Engine

Replication Engine

Replication Engine

EOBC

To Central Arbiter

To Fabric Modules

10G MAC

10G MAC

10G MAC

10G MAC

10G MAC

10G MAC

10G MAC

10G MAC

8-Port 10G XL M1 I/O Module Architecture
N7K-M108X2-12L
32-Port 10GE M1 I/O Modules

N7K-M132XP-12, N7K-M132XP-12L

- 32-port 10G with SFP+ transceivers
- 80G full-duplex fabric connectivity
- Integrated 60Mpps forwarding engine
  - XL forwarding engine on “L” version
- Oversubscription option for higher density (up to 4:1)
- Supports Nexus 2000 (FEX) connections
- Distributed L3 multicast replication
- LISP support
- 802.1AE LinkSec
Shared vs. Dedicated Mode

Shared mode
- Four interfaces in port group share 10G bandwidth

Dedicated mode
- First interface in port group gets 10G bandwidth
- Other three interfaces in port group disabled

"Port group"— group of contiguous even or odd ports that share 10G of bandwidth (e.g., ports 1,3,5,7)
### 32-Port 10G M1 I/O Module Architecture

**N7K-M132XP-12, N7K-M132XP-12L**

**Front Panel Ports**

- Front Panel Ports: 1 through 32

**Components**

- **Forwarding Engine**
- **Replication Engine**
- **VOQs**
- **10G MAC**
- **4:1 Mux + Linksec**
- **EOBC**
- **CPU**
- **LC**

**Connections**

- Connections to Fabric Modules
- Connections to VOQs
- Connections to Replication Engine
- Connections to 10G MAC
- Connections to 4:1 Mux + Linksec

**Layout**

- Front Panel Ports are numbered 1 to 32
- VOQs are connected to Replication Engine and Forwarding Engine
- Replication Engine connects to 10G MAC and 4:1 Mux + Linksec
- 10G MAC connects to Front Panel Ports

---

**© 2012 Cisco and/or its affiliates. All rights reserved.**

**Cisco Public**
32-Port 1G/10GE F1 I/O Module
N7K-F132XP-15

- 32-port 1G/10G with SFP/SFP+ transceivers
- 230G full-duplex fabric connectivity (320G local switching)
- System-on-chip (SoC)* forwarding engine design
  - 16 independent SoC ASICs
- Layer 2 forwarding with L3/L4 services (ACL/QoS)
- FabricPath-capable
- FCoE-capable

* sometimes called "switch-on-chip"
32-Port 1G/10G F1 I/O Module Architecture

N7K-F132XP-15
48-Port 1G/10GE F2 I/O Module
N7K-F248XP-25

- 48-port 1G/10G with SFP/SFP+ transceivers
- 480G full-duplex fabric connectivity
- System-on-chip (SoC)* forwarding engine design
  - 12 independent SoC ASICs
- Layer 2/Layer 3 forwarding with L3/L4 services (ACL/QoS)
- Supports Nexus 2000 (FEX) connections
- FabricPath-capable
- FCoE-ready

* sometimes called “switch-on-chip”
48-Port 1G/10G F2 I/O Module Architecture
N7K-F248XP-25

Front Panel Ports

To Fabric Modules

To Central Arbiters

Fabric 2

EOBC

LC CPU

4 X 10G SoC

1
2

3
4

5
6

7
8

9
10

11
12

13
14

15
16

17
18

19
20

21
22

23
24

25
26

27
28

29
30

31
32

33
34

35
36

37
38

39
40

41
42

43
44

45
46

47
48

Arbitration Aggregator
F2-Only VDC

- F2 modules do **not** interoperate with other Nexus 7000 modules
- Must deploy in an “F2 only” VDC
- Can be default VDC, or any other VDC
  - Use the `limit-resource module-type f2` VDC configuration command
- System with only F2 modules and empty configuration boots with F2-only default VDC automatically

Communication between F2-only VDC and M1/F1 VDC must be through external connection

M1/F1 modules can exist in same **chassis** as F2 modules, but **not** in the same VDC
Agenda

- Chassis Architecture
- Supervisor Engine and I/O Module Architecture
- Forwarding Engine Architecture
- Fabric Architecture
- I/O Module Queuing
- Layer 2 Forwarding
- IP Forwarding
- IP Multicast Forwarding
- Classification
- NetFlow
- Conclusion
M1 Forwarding Engine Hardware

- Hardware forwarding engine(s) integrated on every I/O module
- 60Mpps per forwarding engine Layer 2 bridging with hardware MAC learning
- 60Mpps per forwarding engine Layer 3 IPv4 and 30Mpps Layer 3 IPv6 unicast
- Layer 3 IPv4 and IPv6 multicast support (SM, SSM, bidir)
- MPLS

- OTV
- IGMP snooping
- RACL/VACL/PACL
- QoS remarking and policing policies
- Policy-based routing (PBR)
- Unicast RPF check and IP source guard
- Ingress and egress NetFlow (full and sampled)

<table>
<thead>
<tr>
<th>Hardware Table</th>
<th>M1 Modules</th>
<th>M1-XL Modules without License</th>
<th>M1-XL Modules with License</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIB TCAM</td>
<td>128K</td>
<td>128K</td>
<td>900K</td>
</tr>
<tr>
<td>Classification TCAM (ACL/QoS)</td>
<td>64K</td>
<td>64K</td>
<td>128K</td>
</tr>
<tr>
<td>MAC Address Table</td>
<td>128K</td>
<td>128K</td>
<td>128K</td>
</tr>
<tr>
<td>NetFlow Table</td>
<td>512K</td>
<td>512K</td>
<td>512K</td>
</tr>
</tbody>
</table>
M1 Forwarding Engine Architecture

Ingress pipeline:
- Ingress policing
- Ingress ACL and QoS classification
- Unicast RPF check
- Ingress MAC table lookups
- IGMP snooping lookups
- IGMP snooping redirection

Layer 2 Engine:
- Egress MAC lookups
- IGMP snooping lookups

Layer 3 Engine:
- Egress NetFlow collection
- Egress policing

Packet Headers from I/O Module Replication Engine:
- FIB TCAM and adjacency table lookups for Layer 3 forwarding
- ECMP hashing
- Multicast RPF check

Final lookup result to I/O Module Replication Engine:
- Ingress NetFlow collection
- Ingress NetFlow collection
F2 Forwarding Engine Hardware

- Each SoC forwarding engine services 4 front-panel 10G ports (12 SoCs per module)
- 60Mpps per SoC Layer 2 bridging with hardware MAC learning
- 60Mpps per forwarding engine Layer 3 IPv4/IPv6 unicast
- Layer 3 IPv4 and IPv6 multicast support (SM, SSM)
- IGMP snooping

- RACL/VACL/PACL
- QoS remarking and policing policies
- Policy-based routing (PBR)
- Unicast RPF check and IP source guard
- FabricPath forwarding
- Ingress sampled NetFlow (future)
- FCoE (future)

<table>
<thead>
<tr>
<th>Hardware Table</th>
<th>Per F2 SoC</th>
<th>Per F2 Module</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAC Address Table</td>
<td>16K</td>
<td>256K*</td>
</tr>
<tr>
<td>FIB TCAM</td>
<td>32K IPv4/16K IPv6</td>
<td>32K IPv4/16K IPv6</td>
</tr>
<tr>
<td>Classification TCAM (ACL/QoS)</td>
<td>16K</td>
<td>192K*</td>
</tr>
</tbody>
</table>

* Assumes specific configuration to scale SoC resources
F2 Forwarding Engine Architecture

- Ingress and egress forwarding decisions (L2/L3 lookups, ACL/QoS, etc.)
- "Skid buffer" – Accommodates pause reaction time
- 1G and 10G capable interface MAC
- Four front-panel interfaces per ASIC

1G/10G MAC

Port A 1G/10G

Port B 1G/10G

Pre-Forwarding Ingress Buffer

FIB, ADJ, MAC, ACL, QoS, MET

Ingress Buffer (VOQ)

Port C 1G/10G

Port D 1G/10G

Egress Buffer

Egress fabric receive buffer

Forwarding tables

Ingress Buffer

To Fabric

Virtual output queues

From Fabric

4 X 10G SoC

To/From Central Arbiter

Forwarding Engine

1G/10G Forwarding tables

Ingress and egress forwarding decisions (L2/L3 lookups, ACL/QoS, etc.)

"Skid buffer" – Accommodates pause reaction time

1G and 10G capable interface MAC

Four front-panel interfaces per ASIC
F1 Forwarding Engine Hardware

- Each SoC forwarding engine services 2 front-panel 10G ports (16 SoCs per module)
- 30Mpps per SoC Layer 2 bridging with hardware MAC learning
- IGMP snooping
- VACL/PACL
- QoS remarking policies
- FabricPath forwarding
- FCoE

<table>
<thead>
<tr>
<th>Hardware Table</th>
<th>Per F1 SoC</th>
<th>Per F1 Module</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAC Address Table</td>
<td>16K</td>
<td>256K*</td>
</tr>
<tr>
<td>Classification TCAM (ACL/QoS)</td>
<td>1K in/1K out</td>
<td>16K in/16K out*</td>
</tr>
</tbody>
</table>

* Assumes specific configuration to scale SoC resources
Agenda

- Chassis Architecture
- Supervisor Engine and I/O Module Architecture
- Forwarding Engine Architecture
- Fabric Architecture
- I/O Module Queuing
- Layer 2 Forwarding
- IP Forwarding
- IP Multicast Forwarding
- Classification
- NetFlow
- Conclusion
Crossbar Switch Fabric Modules

- Two fabric generations available – Fabric 1 and Fabric 2

<table>
<thead>
<tr>
<th>Fabric</th>
<th>Per-fabric module bandwidth</th>
<th>Total bandwidth with 5 fabric modules</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fabric 1</td>
<td>46Gbps per slot</td>
<td>230Gbps per slot</td>
</tr>
<tr>
<td>Fabric 2</td>
<td>110Gbps per slot</td>
<td>550Gbps per slot</td>
</tr>
</tbody>
</table>

- Each installed fabric increases available per-payload slot bandwidth
- Different I/O modules leverage different amount of fabric bandwidth
- All I/O modules compatible with both Fabric 1 and Fabric 2
- Access to fabric bandwidth controlled using QoS-aware central arbitration with VOQ
Multistage Crossbar

Nexus 7000 implements 3-stage crossbar switch fabric

- Stages 1 and 3 on I/O modules
- Stage 2 on fabric modules

2nd stage

- 2 x 23Gbps (Fab1) or 2 x 55Gbps (Fab2) per slot per fabric module

1st stage

Up to 230Gbps (Fab1) or 550Gbps (Fab2) per I/O module with 5 fabric modules installed

3rd stage

20 x 23Gbps (Fab1) or 20 x 55Gbps (Fab2) channels per fabric module
I/O Module Capacity – Fabric 1

230Gbps per slot bandwidth

One fabric
- Any port can pass traffic to any other port in system

Two fabrics
- 80G M1 module has full bandwidth

Five fabrics
- 230G F1 module has maximum bandwidth
- 480G F2 module limited to 230G per slot
I/O Module Capacity – Fabric 2

550Gbps per slot bandwidth

One fabric
- **Any port** can pass traffic to any other port in system

Two fabrics
- 80G M1 module has full bandwidth

Five fabrics
- 230G F1 module has maximum bandwidth
- 480G F2 module has maximum bandwidth

Fab2 does **NOT** make Fab1-based modules faster!!

Fabric channels run at lowest common speed

1. 110Gbps/slot
2. 46Gbps/slot
3. 46Gbps/slot
4. 46Gbps/slot
5. 46Gbps/slot

**Local Fabric 2** (480G)

**Local Fabric 1** (230G)

**Local Fabric 1** (80G)
Fabric 1 to Fabric 2 Migration

- Online, non-disruptive migration of Fabric 1 to Fabric 2 supported
- Upgrade to software release supporting Fabric 2
- Remove one Fabric 1 module at a time, replace with Fabric 2 module
  - Allow new Fabric 2 module to come completely online before removing next Fabric 1 module
- Mix of Fabric 1/Fabric 2 not recommended or supported for longer than duration of the migration
  - Within 12 hours of install of first Fabric 2 module, system syslogs warning to complete migration

Arbitration, VOQ, and Crossbar Fabric

- Arbitration, VOQ, and fabric combine to provide all necessary infrastructure for packet transport inside switch

- **Central arbitration** – Controls scheduling of traffic into fabric based on fairness, priority, and bandwidth availability at egress ports

- **Virtual Output Queues (VOQs)** – Provide buffering and queuing for ingress-buffered switch architecture

- **Crossbar fabric** – Provides dedicated, high-bandwidth interconnects between ingress and egress I/O modules
Agenda

- Chassis Architecture
- Supervisor Engine and I/O Module Architecture
- Forwarding Engine Architecture
- Fabric Architecture
- I/O Module Queuing
- Layer 2 Forwarding
- IP Forwarding
- IP Multicast Forwarding
- Classification
- NetFlow
- Conclusion
Buffering, Queuing, and Scheduling

- **Buffering** – storing packets in memory
  - Needed to absorb bursts, manage congestion

- **Queuing** – buffering packets according to traffic class
  - Provides dedicated buffer for packets of different priority

- **Scheduling** – controlling the order of transmission of buffered packets
  - Ensures preferential treatment for packets of higher priority and fair treatment for packets of equal priority

Nexus 7000 uses queuing policies and network-QoS policies to define buffering, queuing, and scheduling behavior

Default queuing and network-QoS policies always in effect in absence of any user configuration
I/O Module Buffering Models

- **Buffering model varies by I/O module family**
  - **M1 modules**: hybrid model combining ingress VOQ-buffered architecture with egress port-buffered architecture
  - **F1/F2 modules**: pure ingress VOQ-buffered architecture

- **All configuration through Modular QoS CLI (MQC)**
  - Queuing parameters applied using class-maps/policy-maps/service-policies
Hybrid Ingress/Egress Buffered Model

M1 I/O Modules

- Ingress port buffer – Manages congestion in ingress forwarding/replication engines only
- Ingress VOQ buffer – Manages congestion toward egress destinations over fabric
- Egress VOQ buffer – Receives frames from fabric; also buffers multidestination frames
- Egress port buffer – Manages congestion at egress interface
Ingress Buffered Model
F1/F2 I/O Modules

- Ingress “skid” buffer – Absorbs packets in flight after external flow control asserted
- Ingress VOQ buffer – Manages congestion toward egress destinations over fabric
- Egress VOQ buffer – Receives frames from fabric; also buffers multidestination frames
Distributed Buffer Pool

- Ingress-buffered architecture implements large, distributed buffer pool to absorb congestion
- Absorbs congestion at every ingress port contributing to congestion, leveraging all per-port ingress buffer
- Excess traffic does not consume fabric bandwidth, only to be dropped at egress port
Agenda

- Chassis Architecture
- Supervisor Engine and I/O Module Architecture
- Forwarding Engine Architecture
- Fabric Architecture
- I/O Module Queuing
- Layer 2 Forwarding
- IP Forwarding
- IP Multicast Forwarding
- Classification
- NetFlow
- Conclusion
Layer 2 Forwarding

- Layer 2 forwarding – traffic steering based on destination MAC address
- Hardware MAC learning
  - CPU not directly involved in learning
- Forwarding engine(s) on each module have copy of MAC table
  - New learns communicated to other forwarding engines via hardware “flood to fabric” mechanism
  - Software process ensures continuous MAC table sync
- Spanning tree (PVRST or MST), Virtual Port Channel (VPC), or FabricPath ensures loop-free Layer 2 topology
Hardware Layer 2 Forwarding Process

- In Classic Ethernet and FabricPath edge switches, MAC table lookup drives Layer 2 forwarding
  - Source MAC and destination MAC lookups performed for each frame, based on \{VLAN,MAC\} pairs
  - Source MAC lookup drives new learns and refreshes aging timers
  - Destination MAC lookup dictates outgoing switchport (CE/FabricPath local) or destination Switch ID (FabricPath remote)

- In FabricPath core switches, Switch ID (routing) table lookup drives Layer 2 forwarding
  - Destination SID lookup dictates outgoing FabricPath interface and next hop
M1 L2 Packet Flow

1. Receive packet from wire
2. LinkSec decryption
3. Ingress port QoS
4. Submit packet headers for lookup
5. L2 SMAC/DMAC lookups
6. ACL/QoS/NetFlow lookups
7. Return result
8. VOQ arbitration and queuing
9. Credit grant for fabric access
10. Transmit to fabric
11. Receive from fabric, Return buffer credit
12. Return credit to pool
13. Egress port QoS
14. LinkSec encryption
15. Transmit packet on wire

HDR = Packet Headers  
DATA = Packet Data  
CTRL = Internal Signaling
F1/F2 L2 Packet Flow

1. Receive packet from wire
2. Ingress port QoS (VOQ)
3. Submit packet headers for lookup
4. Ingress L2 SMAC/DMAC lookups, ACL/QoS lookups
5. Return result
6. VOQ arbitration
7. Credit grant for fabric access
8. Transmit to fabric
9. Receive from fabric
10. Return credit to pool
11. Transmit packet on wire

- VOQ arbitration
- Credit grant for fabric access
- Return credit to pool
- Transmit packet on wire
- Receive packet from wire
- Submit packet headers for lookup
- Ingress L2 SMAC/DMAC lookups, ACL/QoS lookups
- Return result
- Ingress port QoS (VOQ)
- Transmit to fabric
- Receive from fabric
Agenda

- Chassis Architecture
- Supervisor Engine and I/O Module Architecture
- Forwarding Engine Architecture
- Fabric Architecture
- I/O Module Queuing
- Layer 2 Forwarding
- IP Forwarding
- IP Multicast Forwarding
- Classification
- NetFlow
- Conclusion
IP Forwarding

- Nexus 7000 decouples control plane and data plane
- Forwarding tables built on control plane using routing protocols or static configuration
  - OSPF, EIGRP, IS-IS, RIP, BGP for dynamic routing
- Tables downloaded to forwarding engine hardware for data plane forwarding
  - FIB TCAM contains IP prefixes
  - Adjacency table contains next-hop information
Hardware IP Forwarding Process

- FIB TCAM lookup based on destination prefix (longest-match)
- FIB “hit” returns adjacency, adjacency contains rewrite information (next-hop)
- Pipelined forwarding engine architecture also performs ACL, QoS, and NetFlow lookups, affecting final forwarding result
ECMP Load Sharing

- Up to 16 hardware load-sharing paths per prefix
- Use maximum-paths command in routing protocols to control number of load-sharing paths
- Load-sharing is per-IP flow
- Configure load-sharing hash options with global ip load-sharing command:
  - Source and Destination IP addresses
  - Source and Destination IP addresses plus L4 ports (default)
  - Destination IP address and L4 port
- Additional randomized number added to hash prevents polarization
  - Automatically generated or user configurable value
M1 L3 Packet Flow

1. Receive packet from wire
2. LinkSec decryption
3. Ingress port QoS
4. Submit packet headers for lookup
5. L2 ingress and egress SMAC/DMAC lookups
6. L3 FIB/ADJ lookup
7. Ingress and egress ACL/QoS/NetFlow lookups
8. VOQ arbitration and queuing
9. Credit grant for fabric access
10. Transmit to fabric
11. Receive from fabric
12. Return buffer credit
13. Egress port QoS
14. LinkSec encryption
15. Transmit packet on wire
F2 L3 Packet Flow

SoC

Fabric Module 1
Fabric ASIC

Fabric Module 2
Fabric ASIC

Fabric Module 3
Fabric ASIC

Fabric Module 4
Fabric ASIC

Fabric Module 5
Fabric ASIC

Module 1
Module 2

Supervisor Engine

Central Arbiter

- Credit grant for fabric access
- Return credit to pool

HDR = Packet Headers
DATA = Packet Data
CTRL = Internal Signaling

1. Receive packet from wire
2. Ingress port QoS (VOQ)
3. Submit packet headers for lookup
4. Transmit packet on wire
5. Return result
6. VOQ arbitration
7. Credit grant for fabric access
8. Transmit to fabric
9. Receive from fabric
10. Egress port QoS
11. Return buffer credit

- L2 ingress and egress SMAC/DMAC lookups
- L3 FIB/ADJ lookup
- Ingress and egress ACL/QoS lookups
Layer 3 Forwarding with F1 I/O Modules

- F1 modules do not natively provide Layer 3 switching
  - Cannot inter-VLAN route on their own

- However, one or more M1/M1-XL modules can provide “proxy” Layer 3 services
  - M1 forwarding engines can proxy route for F1 modules
  - Proxy L3 forwarding enabled by default in M1/F1 VDC

- Packets destined to router MAC forwarded to M1 modules for Layer 3 via internal “Router Port-Channel”
  - Selection of which port on which M1 module based on EtherChannel hash function
  - Traffic requiring L3 from F1 modules traverses the fabric, “vectoring toward” M1 ports enabled for proxy L3
  - M1 module receiving such packets programmed to perform full ingress/egress L3 lookups
Proxy L3 Forwarding – Conceptual

- From F1 perspective, Router MAC reachable through giant port-channel
- All packets destined to Router MAC forwarded through fabric toward one “member port” in that channel

```
interface vlan 10
ip address 10.1.10.1/24
!
interface vlan 20
ip address 10.1.20.1/24
```
Proxy L3 Forwarding – Actual

1. VLAN DMAC → Dest Port
   10 router_mac → internal_channel (e3/1-8, e4/1-8)

2. EtherChannel Hash Function
   hash_input (from packet) → select_member_port

3. Ingress MAC:
   VLAN DMAC → L3_lookup
   10 router_mac → L3_lookup

4. Routing:
   DIP → Next Hop
   10.1.20.100 → server_2_mac (v20)

5. Egress MAC:
   VLAN DMAC → Dest Port
   20 server_2_mac → e2/1

6. Programming of all F1 forwarding engines

7. Programming of all M1 forwarding engines

8. Can be up to 128 ports on M1 modules

9. interface vlan 10
   ip address 10.1.10.1/24

10. interface vlan 20
    ip address 10.1.20.1/24

11. 10.1.10.100 vlan 10

12. 10.1.20.100 vlan 20

13. M1

14. Fabric

15. Replication Engine

16. VOQs

17. FE

18. M1

19. Fabric

20. Replication Engine

21. VOQs

22. FE

23. Interface

24. VLAN DMAC

25. Dest Port
Agenda

- Chassis Architecture
- Supervisor Engine and I/O Module Architecture
- Forwarding Engine Architecture
- Fabric Architecture
- I/O Module Queuing
- Layer 2 Forwarding
- IP Forwarding
- IP Multicast Forwarding
- Classification
- NetFlow
- Conclusion
IP Multicast Forwarding

- Forwarding tables built on control plane using multicast protocols
  - PIM-SM, PIM-SSM, PIM-Bidir, IGMP, MLD

- Tables downloaded to:
  - Forwarding engine hardware for data plane forwarding (FIB/ADJ)
  - Replication engines for data plane packet replication (Multicast Expansion Table – MET)
IPv4 Multicast FIB TCAM Lookup

1. **Generate Lookup Key**
   - 10.1.1.10, 239.1.1.1

2. **Compare lookup key**
   - 10.1.1.12, 239.1.1.1
   - 10.1.1.10, 232.1.2.3
   - 10.4.7.10, 225.8.8.8
   - 10.1.1.10, 239.1.1.1
   - 10.6.6.10, 239.44.2.1
   - 10.4.7.10, 225.8.8.8

3. **FIB TCAM**
   - Hit in FIB returns result in FIB DRAM
   - RPF, ADJ Index
   - RPF, ADJ Index
   - RPF, ADJ Index
   - RPF, ADJ Index

4. **FIB DRAM**
   - 10.1.1.10, 239.1.1.1
   - 10.1.1.10, 232.1.2.3
   - 10.4.7.10, 225.8.8.8
   - 10.1.1.10, 239.1.1.1
   - 10.6.6.10, 239.44.2.1
   - 10.4.7.10, 225.8.8.8

5. **Hit in FIB returns result in FIB DRAM**
   - RPF, ADJ Index

6. **Replication Engine**
   - Replicate for each OIF in MET block
   - OIFs
   - OIFs

7. **Forwarding Engine**
   - MET Index
   - MET Index
   - MET Index
   - Adjacency Table

8. **Result**
   - Return lookup result

**Ingress multicast packet header**

**Generate TCAM lookup key (source and group IP address)**

**Replication for each OIF in MET block**

**Identifies multicast adjacency entry**

**MET index used to find OIFs for replication**
Egress Replication

- Distributes multicast replication load among replication engines of all I/O modules with OIFs
- Input packets get lookup on ingress forwarding engine
- For OIFs on ingress module, ingress replication engine performs the replication
- For OIFs on other modules, ingress replication engine replicates a single copy of packet into fabric for those egress modules, fabric replicates as needed
- Each egress forwarding engine performs lookup to drive replication
- Replication engine on egress module performs replication for local OIFs
M1 L3 Multicast Packet Flow

1. **Receive packet from wire**
2. **Submit packet headers for lookup**
3. **10G MAC**
4. **Submit packet headers for lookup**
5. **L2 ingress snooping lookup**
6. **L3 multicast FIB lookup + Ingress ACL/QoS/NetFlow lookups**
7. **Return MET result**
8. **Replicate for fabric delivery**
9. **Transmit multicast fabric distribution packet**
10. **VOQ queuing**
11. **Transmit to fabric**
12. **Fabric replication**
13. **Dequeue multicast distribution copy from fabric**
14. **Submit packet headers for egress lookups**
15. **LinkSec encryption**
16. **Layer 2 Engine**
17. **Egress port QoS**
18. **L2 egress snooping lookup**
19. **Transmit packet on wire**
20. **Egress port QoS**

**Fabric Module 1**
- Fabric ASIC
- Layer 2 Engine
- Forwarding Engine
- VOQs
- 10G MAC

**Fabric Module 2**
- Fabric ASIC
- Layer 3 Engine
- Forwarding Engine
- Layer 2 Engine
- Egress port QoS

**Fabric Module 3**
- Fabric ASIC
- Layer 3 Engine
- Forwarding Engine
- Layer 2 Engine

HDR = Packet Headers
DATA = Packet Data
F2 L3 Multicast Packet Flow

1. Receive packet from wire
2. Ingress port QoS (VOQ)
3. Submit packet headers for lookup
4. L2 ingress snooping lookup
   • L3 multicast FIB lookup
   • Ingress ACL/QoS lookups
5. Return MET result
6. Replicate for fabric delivery
7. VOQ queuing
8. Transmit multicast fabric distribution packet
9. Fabric replication
10. Receive from fabric
11. Replicate for local OIF delivery
12. Transmit packet on wire
13. Egress port QoS
14. Egress ACL/QoS lookups and L2 egress snooping lookup for each copy

F2 L3 Multicast Packet Flow

Module 1
- Fabric Module 1
- Fabric ASIC

Module 2
- Fabric Module 2
- Fabric ASIC
- e2/1
- Fabric Module 3
- Fabric ASIC

Module 3
- Fabric Module 4
- Fabric ASIC
- e3/1
- Fabric Module 5
- Fabric ASIC

HDR = Packet Headers
DATA = Packet Data
Agenda

- Chassis Architecture
- Supervisor Engine and I/O Module Architecture
- Forwarding Engine Architecture
- Fabric Architecture
- I/O Module Queuing
- Layer 2 Forwarding
- IP Forwarding
- IP Multicast Forwarding
- Classification
- NetFlow
- Conclusion
What Is Classification?

- Matching packets
  - Layer 2, Layer 3, and/or Layer 4 information
- Used to decide whether to apply a particular policy to a packet
  - Enforce security, QoS, or other policies
- Some examples:
  - Match TCP/UDP source/destination port numbers to enforce security policy
  - Match destination IP addresses to apply policy-based routing (PBR)
  - Match 5-tuple to apply marking policy
  - Match protocol-type to apply Control Plane Policing (CoPP)
  - etc.
CL TCAM Lookup – ACL

Packet header:
SIP: 10.1.1.1
DIP: 10.2.2.2
Protocol: TCP
SPORT: 33992
DPORT: 80

Generate Lookup Key

10.1.1.1 | 10.2.2.2 | tcp | 33992 | 80

SIP | DIP | Pr | SP | DP

CL TCAM

SIP | DIP | Pr | SP | DP

10.1.1.1 | 10.2.2.2 | tcp | 33992 | 80

Hit in CL TCAM returns result in CL SRAM

CL SRAM

Forwarding Engine

RESULTS

Permit
Deny
Permit
Deny
Permit

Security ACL

ip access-list example
permit ip any host 10.1.2.100
deny ip any host 10.1.68.44
deny ip any host 10.33.2.25
permit tcp any eq 22
deny tcp any eq 23
deny udp any eq 514
permit tcp any eq 80
permit udp any eq 161

Hit in CL TCAM returns result in CL SRAM
Packet header:
SIP: 10.1.1.1
DIP: 10.2.2.2
Protocol: TCP
SPORT: 33992
DPORT: 80

Generate TCAM lookup key

10.1.1.1 | 10.2.2.2 | tcp | 33992 | 80

CL TCAM

Comparisons (X = "Mask")

Hit in CL TCAM returns result in CL SRAM

CL SRAM

Results

QoS Classification ACLs
ip access-list police
permit ip any 10.3.3.0/24
permit ip any 10.4.12.0/24
ip access-list remark-dscp-32
permit udp 10.1.1.0/24 any
ip access-list remark-dscp-40
permit tcp 10.1.1.0/24 any
ip access-list remark-prec-3
permit tcp any 10.5.5.0/24 eq 23

Result affects final packet handling
Atomic Policy Programming

- Avoids packet loss during policy updates
- Enabled by default
- Atomic programming process:
  - Program new policy in free/available CL TCAM entries
  - Enable new policy by swapping the ACL label on interface
  - Free CL TCAM resources used by previous policy
Atomic Policy Programming Cont.

- To support atomic programming, **software reserves 50% of available TCAM**
  - Failed to complete Verification: Tcam will be over used, please turn off atomic update
- Disable with **no platform access-list update atomic**
  - Disabling may be necessary for very large ACL configurations
  - Atomic programming attempted but not mandatory
- User can disable atomic programming and perform update non-atomically (assuming ACL fits in CL TCAM)
  - “Default” ACL result (deny by default) returned for duration of reprogramming
  - Use `[no] hardware access-list update default-result permit` to control default result
Classification Configuration Sessions

Two ways to configure ACL/QoS policies:

- Normal configuration mode (**config terminal**)
  - Configuration applied immediately line by line
  - Recommended only for small ACL/QoS configurations, or non-data-plane ACL configuration

- Session config mode (**config session**)
  - Configuration only applied after `commit` command issued
  - Recommended for large ACL/QoS configurations

- Config session mode also provides **verify** facility to “dry-run” the configuration against available system resources
  - No change to existing hardware configuration after verification (regardless of verification result)
Agenda

- Chassis Architecture
- Supervisor Engine and I/O Module Architecture
- Forwarding Engine Architecture
- Fabric Architecture
- I/O Module Queuing
- Layer 2 Forwarding
- IP Forwarding
- IP Multicast Forwarding
- Classification
- NetFlow
- Conclusion
NetFlow on Nexus 7000

- NetFlow collects flow data for packets traversing forwarding engines
- Per-interface full and sampled NetFlow provided by M1 module hardware

<table>
<thead>
<tr>
<th></th>
<th>M1→M1</th>
<th>M1→F1</th>
<th>F1→M1</th>
<th>F1→F1</th>
<th>F2→F2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bridged</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No**</td>
</tr>
<tr>
<td>Routed</td>
<td>Yes</td>
<td>Yes*</td>
<td>Yes*</td>
<td>Yes*</td>
<td>No**</td>
</tr>
</tbody>
</table>

- Each M1 module maintains independent NetFlow table
  - 512K hardware entries per forwarding engine
- Hardware NetFlow entry creation
  - CPU not involved in NetFlow entry creation/update
Full vs. Sampled NetFlow

- NetFlow configured per-direction and per-interface
  - Ingress and/or egress on per-interface basis
- Each interface can collect full or sampled flow data
- Full NetFlow: Accounts for every packet of every flow on interface, up to capacity of NetFlow table
- Sampled NetFlow: Accounts for M in N packets on interface, up to capacity of NetFlow table
Sampled NetFlow Details

- Random packet-based sampling
- M:N sampling: Out of N consecutive packets, select M consecutive packets and account only for those flows in the hardware NetFlow table
- Sampled flows aged and exported from NetFlow table normally

Advantages
- Reduces NetFlow table utilization
- Reduces CPU load on switch and collector

Disadvantages
- Some flows may not be accounted
- Collector extrapolates total traffic load based on configured sampling rate
Netflow Data Export (NDE)

- Process of exporting statistics data from network devices to a “collector”
- Allows long-term baselining, trending, and analysis of NetFlow data
- Exported data sent via UDP
- Variety of export “formats” exist
  - Exported data and format of records varies from version to version
NetFlow Data Export Process

Generate NetFlow v5 or v9 export packets

Hardware Flow Creation

Hardware Flow Creation

Hardware Flow Creation

To NetFlow Collector
via Inband
via mgmt0

Fabric ASIC
VOQs
Main CPU
Switched EOBC
Mgmt Enet

M1 Module
LC CPU
Aged Flows
Forwarding Engine
NetFlow Table

M1 Module
LC CPU
Aged Flows
Forwarding Engine
NetFlow Table

M1 Module
LC CPU
Aged Flows
Forwarding Engine
NetFlow Table
Agenda

- Chassis Architecture
- Supervisor Engine and I/O Module Architecture
- Forwarding Engine Architecture
- Fabric Architecture
- I/O Module Queuing
- Layer 2 Forwarding
- IP Forwarding
- IP Multicast Forwarding
- Classification
- NetFlow
- Conclusion
Nexus 7000 Architecture Summary

I/O Modules

Variety of front-panel interface and transceiver types with hardware-based forwarding and services, including unicast/multicast, bridging/routing, ACL/QoS classification, and NetFlow statistics

Supervisor Engine

Future-proofed chassis designs with density and airflow options

Chassis

High-bandwidth fabric to interconnect I/O modules and provide investment protection

Fabrics

Control plane protocols, system and network management
Conclusion

- You should now have a thorough understanding of the Nexus 7000 switching architecture, I/O module design, packet flows, and key forwarding engine functions…

- Any questions?
Complete Your Online Session Evaluation

- Give us your feedback and you could win fabulous prizes. Winners announced daily.
- Receive 20 Passport points for each session evaluation you complete.
- Complete your session evaluation online now (open a browser through our wireless network to access our portal) or visit one of the Internet stations throughout the Convention Center.

Don’t forget to activate your Cisco Live Virtual account for access to all session material, communities, and on-demand and live activities throughout the year. Activate your account at the Cisco booth in the World of Solutions or visit www.ciscolive.com.
Final Thoughts

- Get hands-on experience with the Walk-in Labs located in World of Solutions, booth 1042
- Come see demos of many key solutions and products in the main Cisco booth 2924
- Visit www.ciscoLive365.com after the event for updated PDFs, on-demand session videos, networking, and more!
- Follow Cisco Live! using social media:
  - Facebook: https://www.facebook.com/ciscoliveus
  - Twitter: https://twitter.com/#!/CiscoLive
  - LinkedIn Group: http://linkd.in/CiscoLI