Tutorial Part II Outline: Current technologies: Protocols

- InfiniBand
- Ethernet based data centers
  - IETF TRILL
  - IEEE 802.1aq: Shortest Path Bridging
- Data center bridging
  - Make Ethernet switched networks more reliable
  - To enable support for InfiniBand transport protocol
  - Low latency
- RoCE and iWARP
Scientific community

- Department of Energy (DOE) funds scientific research in basic sciences
  - High energy physics
  - Basic energy sciences
  - Biological and Environmental Research
  - Fusion Energy Sciences
  - Nuclear Physics
  - Advanced Scientific Computing Research (ASCR)
DOE labs and facilities

- DOE national labs
  - Oak Ridge National Lab (ORNL)
  - Argonne National Lab (ANL)
  - Lawrence Berkeley National Laboratory (LBNL)
  - National Energy Research Scientific Computing Center (NERSC)
  - Brookhaven National Lab (BNL)
  - and other labs ...

- Supercomputing facilities
  - ANL
  - NERSC
  - ORNL
  - and others ...

- ESnet: 100 Gb/s IP-routed, WDM and dark-fiber network
Energy Sciences Network (ESnet)
Magellan: DOE Cloud Computing @NERSC

720 nodes, 5760 cores in 9 Scalable Units (SUs) ➔ 61.9 Teraflops
SU = IBM iDataplex rack with 640 Intel Nehalem cores

InfiniBand
QDR IB Fabric

18 Login/network nodes
Login
Network
HPSS (15PB)
Load Balancer
100-G Router
ANI
NERSC Global Filesystem
1 Petabyte with GPFS
14 I/O nodes
I/O

10G Ethernet

FibreChannel
8G FC

Internet
Three network technologies with a single data center

• InfiniBand: Inter-Process Communications (IPC)
  – IB rates (4x): QDR: 32 Gbps; FDR: 56 Gbs; EDR: 100 Gbps
• FibreChannel: Storage
• Ethernet: Wide-area network access
Why converge to a single technology?

• Several benefits:
  – less space, less heat, less electricity, less knowledge, less maintenance.

• All of these benefits amount to one major benefit
  – less cost

Mikkel Hagen, UNH IOL, Data Center Bridging Tutorial
InfiniBand clusters

- NCAR (National Center for Atmospheric Research) Wyoming SuperComputing Center (NWSC) - Yellowstone
  - http://nwsc.ucar.edu/

- NERSC Carver
  - http://www.nersc.gov/users/computational-systems

- New Mexico consortium: NSF funded – Kodiak, Denali
  - https://www.nmc-probe.org/wiki/Machines

- Texas Advanced Computing Center (TACC) – Stampede
  - http://www.tacc.utexas.edu/stampede
NWSC InfiniBand cluster

Yellowstone network

251 A-groups
14 B-groups
9 ORCAs

John Dennis, NCAR
InfiniBand

- Low-latency data-center networking technology
- Switched fabric
- Uses Remote Direct Memory Access (RDMA)
- Kernel bypass leads to low latency for operations within the servers

*InfiniBand™ Architecture Specification Volume 1
Release 1.2.1, Nov. 2007, InfiniBand™ Trade Assoc.*
Networking features

- Architecture
- Addressing
- Packet format
- Link-layer: error control? Flow control?
- Network-layer: IPv6
- Transport layer
  - API and services (difference between verbs/RDMA and TCP/IP socket)
  - Flow control, error control, congestion control
- Subnet manager & Routing
InfiniBand network architecture

- End nodes have Channel Adapters (host and target for I/O)
- Switch: forwards packets on destination local ID within local route header
- Router: “Global route header” ≡ IPv6 header
InfiniBand addressing (unicast)

- **Local Identifier (LID):** 16-bit identifier assigned by subnet manager – local to subnet
- **Global Identifier (GID):** 16-byte IPv6 address (with IB restrictions): GID prefix + GUID
- **Global Unique Identifier (GUID):** EUI-64 (created from 6-byte MAC address configured by manufacturer)
  - Interface ID in IPv6 terminology
- **GID prefix:** 64 bits
  - subnet prefix (≤ 64 bits) + filler bits

InfiniBand packet format

• Local Routing Header:
  – Basic fields: Destination and Source Local ID, packet length, next hdr
  – Other features: Virtual lane, service level

• Global Routing Header: IPv6 header

• Unlike in the Ethernet world, where IP header is present in all packets irrespective of whether the packet is intra- or inter-subnet, in IB, IP header is only needed if inter-subnet
InfiniBand link layer

• Virtual lane and service level features
  – Buffering per VL
  – Flow control per VL
  – Subnet manager configures number of VLs per link

• Flow control: per virtual lane

• Error control:
  – Error detection: CRC, buffer overrun, switch routing errors, etc.
  – Error recovery: None

• Multipathing
  – Base LID and a 3-bit LID Mask Control (LMC)
  – Subnet manager assigns a base ID and up to $2^{\text{LMC}}$ sequential LIDs to each endport
Multipath example

- Subnet manager may configure multiple (disjoint) paths through the subnet for these various LIDs.

Four paths exist between channel adapters A and C. CA A is assigned a Base LID 4, LMC = 2. This translates to CA A being assigned LIDs: \{4, 5, 6, 7\}. CA C is assigned Base LID 8, LMC = 2. This translates into CA C being assigned LIDs: \{8, 9, 10, 11\}.

*Figure 43  Multipath Identification*
IB link-layer flow control

- Credit based flow control (Kung and Morris, IEEE Network 95)
- Flow control packet – receiver indicates Fwd_Cnt (all packets forwarded)
- Receiver provides sender buffer allocation & forwarded packet count
- Transmitter maintains count of all transmitted packets (Tx_Cnt)
- Crd_Bal = Buf_Alloc - (Tx_Cnt - Fwd_Cnt): determines how many packets to send
Networking features

- Architecture
- Addressing
- Packet format
- Link-layer: error control? Flow control?
- Network-layer: IPv6
  - Transport layer
    - API and services (difference between verbs/RDMA and TCP/IP socket)
    - Flow control, error control, congestion control
- Subnet manager & Routing
InfiniBand transport layer

- **API and services: verbs interface (analogy: TCP sockets)**
  - **Channel semantics: SEND/RECEIVE**
    - sender pushes the data, but it is the receiver that determines where to write the data
    - message does not describe where in receiver’s memory space the message content should be written
  - **Memory semantics: READ/WRITE**
    - initiating side directly reads/writes the virtual address space of the remote node
    - remote node communicates location of the buffer, and is NOT involved with the actual data transfer
  - **Use both:** channel semantics used to send a disk write or read command; then memory semantics used for actual transfer; then channel semantics used to push an I/O completion message back
Queues

- **Queue Pair (QP):** Send and Receive work queues
- **Completion queues (CQ)**
- **Transport service types:**
  - reliable connection
  - reliable datagram
  - unreliable datagram
  - unreliable connection
- **Connection:** QPs are “connected”
  - unreliable connection: multiple messages can be exchanged + no retries
  - unreliable datagram: only single-packet messages + no retries
  - reliable connection: a QP is associated with only one other QP
  - reliable datagram: a QP may communicate with multiple QPs (not limited to single packets)
    - how is reliable multicast handled? ACK implosion problem
Base transport header

- **OpCode**: identifies function
- **Partition key**
- **Destination**
- **Packet sequence number**

### Extended transport headers (ETH)
- Reliable Datagram (RDETH)
- Datagram (DETH)
- RDMA (RETH)
- Atomic (AtomicETH)
- Acknowledgment (AETH)
- Immediate (IMMDT)
Transport functions ↔ transport services

Table 36 Transport Functions Supported for Specific Transport Services

<table>
<thead>
<tr>
<th>Transport Function</th>
<th>Reliable Connection</th>
<th>Unreliable Connection</th>
<th>Reliable Datagram</th>
<th>Unreliable Datagram</th>
<th>Raw Datagram</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEND</td>
<td>supported</td>
<td>supported</td>
<td>supported</td>
<td>supported</td>
<td>not applicable</td>
</tr>
<tr>
<td>RESYNC</td>
<td>not supported</td>
<td>not supported</td>
<td>supported</td>
<td>not supported</td>
<td>not supported</td>
</tr>
<tr>
<td>RDMA WRITE</td>
<td>supported</td>
<td>supported</td>
<td>supported</td>
<td>not supported</td>
<td>not applicable</td>
</tr>
<tr>
<td>RDMA READ</td>
<td>supported</td>
<td>not supported</td>
<td>supported</td>
<td>not supported</td>
<td>not applicable</td>
</tr>
<tr>
<td>ATOMIC Operations</td>
<td>optional support</td>
<td>not supported</td>
<td>optional support</td>
<td>not supported</td>
<td>not applicable</td>
</tr>
</tbody>
</table>

- ATOMIC operation: execute a 64-bit operation at a specified address on a remote node; atomically read, modify and write with a guarantee that operations on this address by other QPs on the same CA do not occur.
- RESYNC: zero-length SEND-only reliable datagram – forces responder to reset the expected packet sequence number.
How RDMA differs from TCP/IP

- “zero copy” – data transferred directly from virtual memory on one node to virtual memory on another node

- “kernel bypass” – no operating system involvement during data transfers

- asynchronous operation – threads not blocked during I/O transfers

Robert D. Russell, UNH
TCP/IP transfer

**Client Setup**
- User App
- Kernel Stack
- CA
- Wire

**Transfer**
- User App
- Kernel Stack
- CA
- Wire

**Server Setup**
- User App
- Kernel Stack
- CA
- Wire

- Blue lines: control information
- Red lines: user data
- Green lines: control and data

Robert D. Russell, UNH
RDMA transfer

Client

User App

Kernel Stack

CA

Wire

setup

rdma_connect

rdma_post_send

transfer

data

red lines: user data

Server

User App

Kernel Stack

CA

Wire

setup

rdma_bind

rdma_listen

rdma_accept

transfer

data

rdma_post_recv

blue lines: control information

green lines: control and data

Robert D. Russell, UNH
TCP RECV()
RDMA RECV()

USER

allocate
register

recv()

parallel
activity

poll_cq()
access

CHANNEL ADAPTER

recv queue

metadata

completion queue

status

data packets

control

WIRE

ACK

Robert D. Russell, UNH
Kernel Bypass

- User interacts directly with CA queues
- Queue Pair from program to CA
  - work request – data structure describing data transfer
  - send queue – post work requests to CA that send data
  - recv queue – post work requests to CA that receive data
- Completion queues from CA to program
  - work completion – data structure describing transfer status
  - Can have separate send and receive completion queues
  - Can have one queue for both send and receive completions

Robert D. Russell, UNH
ping-pong measurements

❖ Client
  – round-trip-time 15.7 microseconds
  – user CPU time 100% of elapsed time
  – kernel CPU time 0% of elapsed time

❖ Server
  – round-trip time 15.7 microseconds
  – user CPU time 100% of elapsed time
  – kernel CPU time 0% of elapsed time

❖ InfiniBand QDR 4x through a switch

Robert D. Russell, UNH
How to reduce 100% CPU usage

- **Cause is “busy polling” to wait for completions**
  - in tight loop on `ibv_poll_cq()`
  - burns CPU since most calls find nothing

- **Why is “busy polling” used at all?**
  - simple to write such a loop
  - gives very fast response to a completion
  - (i.e., gives low latency)

Robert D. Russell, UNH
“busy polling” to get completions

1. start loop

2. `ibv_poll_cq()` to get any completion in queue

3. exit loop if a completion is found

4. end loop
How to eliminate “busy polling”

- Cannot make `ibv_poll_cq()` block
  - no flag parameter
  - no timeout parameter
- Must replace busy loop with “wait – wakeup”
- Solution is a “wait-for-event” mechanism
  - `ibv_req_notify_cq()` - tell CA to send an “event” when next WC enters CQ
  - `ibv_get_cq_event()` - blocks until gets “event”
  - `ibv_ack_cq_event()` - acknowledges “event”
ping-pong measurements with wait

Client
- round-trip-time 21.1 microseconds – up 34%
- user CPU time 9.0% of elapsed time
- kernel CPU time 9.1% of elapsed time
- total CPU time 18% of elapsed time – down 82%

Server
- round-trip time 21.1 microseconds – up 34%
- user CPU time 14.5% of elapsed time
- kernel CPU time 6.5% of elapsed time
- total CPU time 21% of elapsed time – down 79%

Robert D. Russell, UNH
IB transport-layer Error, Flow, & Congestion control

• Error control
  – For reliable service types, QP detects one or more missing packets and sends NAK message indicating next expected sequence number
  – When originator receives ACK, it creates a CQE on the completion queue and retires the WQE from the send queue
  – A QP can have multiple outstanding messages at any one time (not Stop-and-Wait)

• End-to-end (message) flow control:
  – end-to-end flow credits carried in AETH Syndrome field from responder to requester

• Congestion control: Explicit congestion notification
  – Forward direction to notify destination for packets transmitted on to a congested VL
  – Destination notifies source
  – Source reduces rate of packet injection
Networking features

- Architecture
- Addressing
- Packet format
- Link-layer: error control? Flow control?
- Network-layer: IPv6
- Transport layer
  - API and services (difference between verbs/RDMA and TCP/IP socket)
  - Flow control, error control, congestion control

➢ Subnet manager & Routing
InfiniBand network architecture

Figure 8 IBA Network Components

Figure 9 IBA Subnet Components
IB Subnet manager and subnet management agents (on each CA, switch, router)

- Discovers subnet topology
- Configures each channel adapter port with a range of local IDs, GID subnet prefixes, and keys
- Configures each switch with a LID, the subnet prefix (why?), and forwarding database
  - there isn’t anything like STP or OSPF/ISIS that automatically generates the forwarding tables
  - Provides GUID to LID/GID resolution service (instead of ARP)
- SMP: Subnet management packets
  - LID Routed packets: forwarded based on the LID of the destination
  - Directed Routed packets
    - source routing
    - useful means to communicate before switches are configured
    - carry a path vector
    - switch receiving these packets: look at current hop to identify location in the path vector to determine next hop
Subnet management

- Uses SNMP like Set/Get attributes and Traps
- Attributes:
  - NodeInfo: GUID of CA, and GUID of each port
  - SwitchInfo: SL/VL, capacities of forwarding tables, etc.
  - PortInfo: LIDs
  - ForwardingTables

InfiniBand packet forwarding

- Configured by Subnet Manager
- Linear forwarding table (Table 154)
  - A port corresponding to 64 LIDs (in-sequence)
- Random forwarding table (Table 156)
  - Individual LID to port mappings

**Table 154 Port Block Element**

<table>
<thead>
<tr>
<th>Component</th>
<th>Length (bits)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port</td>
<td>8</td>
<td>Port to which packets with the LID corresponding to this entry are to be forwarded.</td>
</tr>
</tbody>
</table>

**Table 156 LID/Port Block Element**

<table>
<thead>
<tr>
<th>Component</th>
<th>Length (bits)</th>
<th>Offset (bits)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LID</td>
<td>16</td>
<td>0</td>
<td>Base LID.</td>
</tr>
<tr>
<td>Valid</td>
<td>1</td>
<td>16</td>
<td>This LID/Port pair is valid. Note that setting this parameter to 0 allows the removal of entries.</td>
</tr>
<tr>
<td>LMC</td>
<td>3</td>
<td>17</td>
<td>the LMC of this LID.</td>
</tr>
<tr>
<td>Reserved</td>
<td>4</td>
<td>20</td>
<td>Reserved</td>
</tr>
<tr>
<td>Port</td>
<td>8</td>
<td>24</td>
<td>Port to which packets with this LID/LMC corresponding to this entry are to be forwarded.</td>
</tr>
</tbody>
</table>
Routing algorithms

• A few examples:
  – Min hop algorithm
  – UPDN algorithm
  – Fat tree algorithm
  – LASH algorithm
  – DOR algorithm

• http://www.mellanox.com/related-docs/prod_software/Mellanox%20OFED%20Linux%20User%20Manual%20Manual%2001_5_3-1_0_0.pdf
## Comparison

<table>
<thead>
<tr>
<th>Feature</th>
<th>Ethernet</th>
<th>InfiniBand</th>
</tr>
</thead>
<tbody>
<tr>
<td>Addressing</td>
<td>MAC (6-byte)</td>
<td>Local ID (2-byte)</td>
</tr>
<tr>
<td>Forwarding table (created by)</td>
<td>Address learning &amp; STP for port state; (distributed)</td>
<td>Subnet manager (centralized)</td>
</tr>
<tr>
<td>IP header inclusion in all packets (even intra-subnet)</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Logical links</td>
<td>VLAN</td>
<td>Virtual lane</td>
</tr>
<tr>
<td>Source routing</td>
<td>No</td>
<td>Directed Routes</td>
</tr>
<tr>
<td>Multipath support</td>
<td>Not at Ethernet layer; MP-TCP</td>
<td>Yes</td>
</tr>
<tr>
<td>Transport-layer congestion control</td>
<td>TCP: transport (yes) New in 802.1Q</td>
<td>IB switches are involved + IBT</td>
</tr>
</tbody>
</table>
Part II Outline:
Current technologies: Practice

- InfiniBand
  - Ethernet based data centers
    - Issues with Spanning Tree Protocol
    - IETF TRILL
    - IEEE 802.1aq: Shortest Path Bridging
  - Data center bridging
    - Make Ethernet switched networks more reliable
    - To enable support for InfiniBand transport protocol
    - Low latency
  - RoCE and iWARP
Spanning tree protocol (STP)

- Goal: Break routing loops
- Configuration Bridge Protocol Data Units (BPDUs) are exchanged between switches
- Plug-and-play: Pre-assigned priority ID and MAC address of port 1 determine default bridge ID
- Root bridge of tree: one with smallest bridge ID
- Each bridge starts out thinking it is the root bridge
- Through BPU exchanges, tree converges, which means all switches have same view of the spanning tree
- Each bridge determines which of its ports should be root ports and which designated ports
- These ports are placed in forwarding state; rest are blocked
- Packets will not be received or forwarded on blocked ports
- Advantage: zero-configuration!
- Disadvantage: root bridge could become bottleneck; no load balancing
Example of STP
STP: Advantages/disadvantages

• Advantage:
  – Plug and Play – No configuration required

• Disadvantages:
  – Scalability issue:
    • What is the impact of MAC learning on the core?
  – Possibility of loops negatively affecting the network:
    • No hop count or time-to-live in header to drop looping packets
  – Layer 2 redundancy unexploited:
    • Blocked links created by STP
IETF TRILL (TRansparent Interconnection of Lots of Links)

• Goal
  – Design so that change can be incremental
  – With TRILL, replace any subset of bridges with RBridges (Routing Bridges)
    • still looks to IP like one giant Ethernet
    • the more bridges you replace with RBridges, better bandwidth utilization, more stability

Radia Perlman, Intel Labs, HPSR 2012
TRILL

- Basic concept: RBridges (Routing Bridges)
- Use of link-state routing mechanism between RBs
- Frame format
- How addresses are learned?
- Unknown destinations
- [Multicast not covered]
Basic TRILL concept

- RBridges find each other (perhaps with bridges in between) with link-state protocol
- Calculate paths to other RBridges
- First RBridge tunnels frames to last RBridge
- Reason for extra header:
  - Forwarding table in RBridges just size of # of RBridges
  - Layer 3-like header (hop count)
  - Small, easy to look up, addresses
Routing inside campus

- First RB encapsulates frame and sends to last RB
  - So header is “safe” (has hop count – so even if temporary loops are formed, packets will be dropped)
  - Inner R Bridges only need to know how to reach destination R Bridge
- Still need tree for unknown/multicast
  - But don’t need spanning tree protocol – compute tree(s) deterministically from the link state database

Radia Perlman, Intel Labs, HPSR 2012
Note: only one T must encap/decap
So T1 and T2 must find each other and coordinate (Designated RB)

Radia Perlman, Intel Labs, HPSR 2012
Frame format

- Outer Ethernet Header
- TRILL Header
- Inner Ethernet Header
- Ethernet Payload
- Ethernet FCS

Figure 2: An Ethernet Encapsulated TRILL Frame

- Outer header: ingress and egress RB MAC addresses
- TRILL header: for packet forwarding between ingress-egress RBs
- Inner header: original frame header
- Because TRILL nicknames are not 6-byte MAC, need outer header (compare to PBB)

IETF RFC 6325
Encapsulated Frame

(Ethernet) outer header

TRILL header

original frame

dest (nexthop)
srce (Xmitter)
Ethertype=TRILL

first RBridge
last RBridge
TTL

TRILL header specifies RBridges with 2-byte nicknames

Radia Perlman, Intel Labs, HPSR 2012
2-byte Nicknames

- Saves hdr room, faster fwd’ing
- Dynamically acquired
- Choose unused #, announce in LSP (Link State Protocol: ISIS)
- If collision, IDs and priorities break tie
- Loser chooses another nickname
- Configured nicknames higher priority

Radia Perlman, Intel Labs, HPSR 2012
Benefits offered by TRILL header

- loop mitigation through use of a hop count field
- elimination of the need for end-station VLAN and MAC address learning in transit R Bridges
- unicast forwarding tables of transit R Bridges size depends on the number of R Bridges rather than the total number of end nodes
- provision of a separate VLAN tag for forwarding traffic between R Bridges, independent of the VLAN of the native frame (inner header VLAN ID different from outer header VLAN ID)
Address learning

- RB1 that is VLAN-x forwarder learns
  - port, VLAN, and MAC addresses of end nodes on links for which it is VLAN-x forwarder from source addresses of frames received
  - Or through configuration
  - Or through Layer-2 explicit registration, e.g., 802.11 Association
- RB1 learns the VLAN and MAC addresses of distant VLAN-x end nodes, and corresponding RB to which they are connected by
  - extracting ingress RB nickname from TRILL header, AND
  - VLAN and source MAC address of the inner frame
- End-Station Address Distribution Information (ESADI) protocol
  - RB that is the appointed VLAN-x forwarder could use this protocol to announce some or all of the attached VLAN-x end nodes to other RBs

IETF RFC 6325
Unknown destinations

• If destination address is unknown at an ingress RB, it sends the packets through the spanning tree as an ordinary bridge
• Set the M-bit to 1 (for multicast/broadcast) frames
• For packets sent on links leading to other RBs, it adds a TRILL header and sets the egress RBridge ID to tree ID so that the TRILL frame header is processed by all receiving RBriges on that particular distribution tree
IEEE bridging protocols

• 802.1D (2004)
  - STP: Spanning Tree Protocol
  - RSTP: Rapid Spanning Tree Protocol (RTSP)

• 802.1Q (2011)
  - VLAN and priority support
  - VLAN classification according to link layer protocol type (802.1v)
  - MSTP: Multiple STP: One STP per non-overlapping group of VLANs (802.1s)
  - Provider bridging (802.1ad)
    • added support for a second level of VLAN tag, called a "service tag", and renamed the original 802.1Q tag a "customer tag". Also known as Q-in-Q because of the stacking of 802.1Q VLAN tags.
  - Provider Backbone Bridges (802.1ah)
    • added support for stacking of MAC addresses by providing a tag to contain the original source and destination MAC addresses. Also know as MAC-in-MAC.

Review from IETF RFC 5556
### IEEE 802.1Q Ethernet VLAN

<table>
<thead>
<tr>
<th>Dest. MAC Address</th>
<th>Source MAC Address</th>
<th>TPID</th>
<th>Type / Len</th>
<th>Data</th>
<th>FCS</th>
</tr>
</thead>
</table>

FCS: Frame Check Sequence

#### VLAN Tag

<table>
<thead>
<tr>
<th>802.1Q Tag Type</th>
<th>Priority Code Point</th>
<th>DEI</th>
<th>VLAN ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 Bytes</td>
<td>3 Bits</td>
<td>1 Bit</td>
<td>12 Bits</td>
</tr>
</tbody>
</table>

DEI: Drop Eligible Indicator
Ether type values

- **Type field values**
  - 0x0800: IP
  - 0x0806: ARP
  - 0x8808: Ethernet flow control (GbE has PAUSE)
  - 0x8870: Jumbo frames (MTU: 9000 Bytes instead of 1500 B)

<table>
<thead>
<tr>
<th>Tag Type</th>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer VLAN Tag</td>
<td>IEEE 802.1Q Tag Protocol EtherType (802.1QTagType)</td>
<td>81-00</td>
</tr>
<tr>
<td>Service VLAN Tag or</td>
<td>Backbone VLAN Tag or</td>
<td></td>
</tr>
<tr>
<td>Backbone Instance Tag</td>
<td>IEEE 802.1Q Service Tag EtherType (802.1QSTagType)</td>
<td>88-a8</td>
</tr>
<tr>
<td></td>
<td>IEEE 802.1Q Backbone Service Instance Tag EtherType</td>
<td>88-e7</td>
</tr>
</tbody>
</table>
Provider bridging

*Figure 14-1  802.1Q Tunnel Ports in a Service-Provider Network*

802.1Q and Q-in-Q (provider bridging)

- Frames entering the edge switch tunnel ports with 802.1Q tags are double-tagged when they enter the service-provider network, with the outer tag containing VLAN ID 30 or 40 for customer A and customer B frames, respectively.
- Inner tag contains the original customer VLAN number, for example, VLAN 100.
- Both Customers A and B can have VLAN 100 in their networks, the traffic remains segregated within the service-provider network because the outer tag is different.
- Each customer controls its own VLAN numbering space, which is independent of the VLAN numbering space used by other customers and the VLAN numbering space used by the service-provider network.
Multi-tenant applications (relates carrier Ethernet PB, PBB apply to datacenters)

• As large enterprises continue to evolve, many have become very similar to network service providers/carriers. The enterprise IT organization is the “service provider” for its internal customers.
• With the need to support these complex multi-tenant environments comes the added cost and complexity of operating a “carrier-class” network.
• SPB is the technology that will help satisfy all aspects of the multi-tenant customer. The technology evolved from similar protocols used by carriers and service providers. SPB has been enhanced to add “enterprise friendly” features to give it the best of both worlds, carrier robustness / scalability and applicability with enterprise-class features and interoperability.
Provider Bridging (PB) vs. Provider Backbone Bridging (PBB)

PB and PBB tagging

Why is PBB required?

• In PB, the service provider network has to learn customer MAC addresses. Hence it is not scalable.
• PBB solves this scalability problem with a new frame format:
  – Customer frame encapsulated in another Ethernet frame with BEB (B-MAC) addresses as source and destination
  – Core switches forward traffic based on backbone MAC (B-MAC) addresses.
  – Confines the requirement to learn customer addresses to the BEB (edge devices) of the PBB network
  – A BEB is required to learn the addresses of only those customers that it supports, and a given BCB is required to learn the addresses of only BEBs (as opposed to having to learn addresses of all of the end customer devices)
  – This greatly enhances the scalability of the solution
• Avaya white paper and Cisco 2008 paper
Another problem with PB: service instance scalability: limited to 4096 (12 bit S-VLAN ID)

- PBB frame header: 24-bit I-SID (Backbone Service Instance Identifier)
- Each customer service instance is assigned a unique I-SID value within a service provider’s network.
  - Hence, number of service instances increased from 4094 to a theoretical maximum limit of roughly 16 million ($2^{24}$).
- I-SIDs are visible to BEB (edge) only
- I-SIDs are transparent to the BCB (core)
- PBB frame header also has 12-bit backbone VLAN ID (B-VLAN).
  - Allows provider to partition its network into different broadcast domains
  - Bundle different I-SIDs into distinct B-VLANs
  - Map different B-VLANs into different spanning-tree instances
IEEE 802.1aq Shortest Path Bridging (SPB)

- SPB comes in 2 flavors:
  - SPBV (using 802.1ad aka Q-in-Q)
  - SPBM (using 802.1ah aka MAC-in-MAC encapsulation)

- An SPT Bridge using SPBV mode:
  - supports a C-VLAN or S-VLAN for a single customer
  - uses address learning

- An SPT Bridge using SPBM mode:
  - support B-VLANs in Provider Backbone Bridged Networks
  - does not use source address learning, so unicast B-MAC frames conveying customer data are never flooded throughout the B-VLAN

- Both variants use IS-IS as the link-state routing protocol to compute shortest paths between nodes (RFC 6329)
Good overview of IEEE 802.1aq in IETF RFC 6329
IEEE calls it Filtering (of broadcast traffic) databases, while IETF calls it Forwarding (explicit direction of unicast traffic)
Symmetric (forward and reverse paths) and congruent (with respect to unicast and multicast)
  - shortest path tree (SPT) for a given node is congruent with multicast distribution tree (MDT)
  - preserve packet ordering and share Operations, Administration and Maintenance (OAM) flows with forwarding path
SPBM filtering database (FDV) is computed and installed for MAC addresses (unicast and multicast)
SPMV filtering database is computed and installed for VIDs, after which MAC addresses are “learned” for unicast MAC (as in ordinary bridged networks)
Terminology (Multiple Spanning Tree)

- **MST Bridge**: A Bridge capable of supporting the common spanning tree (CST, and one or more MSTIs, and of selectively mapping frames classified in any given VLAN to the CST or a given MSTI.

- **MST Configuration Table**: A configurable table that allocates each and every possible VID to the Common Spanning Tree or a specific Multiple Spanning Tree Instance.

- **MST Region**: One or more MST Bridges with the same MST Configuration Identifiers, interconnected by and including LANs for which one of those bridges is the Designated Bridge for the CIST and which have no bridges attached that cannot receive and transmit RST BPDUs.

- **Multiple Spanning Tree (MST) Configuration Identifier**: A name for, revision level, and a summary of a given allocation of VLANs to Spanning Trees. [New ISIS parameter: 51B]

- **Multiple Spanning Tree Instance (MSTI)**: One of a number of Spanning Trees calculated by MSTP within an MST Region, to provide a simply and fully connected active topology for frames classified as belonging to a VLAN that is mapped to the MSTI by the MST Configuration Table used by the MST Bridges of that MST Region.

IEEE 802.1Q
Routing algorithms

- IETF RFC 6329: Shortest-path default tie-breaking
  - ECT-Algorithm (Equal Cost Tree)
  - Standard ECT Algorithms
- IEEE 802.1Q: Rules for creating MST regions and MSTIs
Industry perspective

• “Industry split on data center network standards” Mar. 22, 2011

• Vendors
  – Cisco’s FabricPath for its Nexus 7000 switch: superset of TRILL
  – BrocadeOne fabric architecture: based on TRILL
  – Juniper: TRILL detractor:
    • QFabric line: proprietary way of scaling Ethernet in datacenters
  – HP is supporting both TRILL and SPB
  – Huawei is supporting both
  – Avaya and Alcatel-Lucent: supporting SPB given carrier roots
Part II Outline: Current technologies: Practice

- InfiniBand
- Ethernet based data centers
  - IETF TRILL
  - IEEE 802.1aq: Shortest Path Bridging

- Data center bridging
  - Make Ethernet switched networks more reliable
  - To enable support for InfiniBand transport protocol
  - Low latency

- RoCE and iWARP
Data center bridging

- Data Center Bridging is focused primarily on three (3) IEEE specifications:
  - IEEE 802.1Qaz – ETS & DCBX – bandwidth allocation to major traffic classes (Priority Groups); plus DCB management protocol
  - IEEE 802.1Qbb – Priority PAUSE. Selectively PAUSE traffic on link by Priority Group
  - IEEE 802.1Qau – Dynamic Congestion Notification (part of 802.1Q 2011)
IEEE 802.1Qaz

• **Enhanced transmission selection**
  – Support multiple traffic classes
  – Support priority queueing
  – Support per-traffic class bandwidth allocation (weighted fair queueing)
  – Credit based traffic shaper

• **Data Center Bridging eXchange (DCB-X) protocol**
  – Discovery of DCB capability in a peer port: for example, it can be used to determine if peer ports support PFC (Priority based Flow Control)
  – DCB feature misconfiguration detection: possible to misconfigure a feature between the peers on a link.
  – Peer configuration of DCB features: if the peer port is willing to accept configuration.
IEEE 802.1Qbb

• Priority based flow control
  – PFC allows link flow control to be performed on a per-priority basis.
  – PFC is used to inhibit transmission of data frames associated with one or more priorities for a specified period of time.
  – PFC can be enabled for some priorities on the link and disabled for others.

• 8 priority levels per port

• In a port of a Bridge or station that supports PFC, a frame of priority n is not available for transmission if that priority is paused on that port.
IEEE 802.1Qau: part of 802.1Q 2011

• Quantized Congestion Notification (QCN) algorithm
  – Congestion Point (CP) Algorithm: a congested bridge samples outgoing frames and generates a feedback message (Congestion Notification Message or CNM) to the source of the sampled frame with information about the extent of congestion at the CP.
  – Reaction Point (RP) Algorithm: a Rate Limiter (RL) associated with a source decreases its sending rate based on feedback received from the CP, and increases its rate unilaterally (without further feedback) to recover lost bandwidth and probe for extra available bandwidth.
  • See 802.1Q Section 30 for details

• Congestion Notification Tag
  – An end station may add a Congestion Notification Tag (CN-TAG) to every frame it transmits from a Congestion Controlled Flow (e.g., same src/dst MAC + priority)
  – CN-TAG contains a Flow Identifier (Flow ID) field.
  – The destination_address, Flow ID, and a portion of the frame that triggered the transmission of the CNM are the means by which a station can determine to which RP a CNM applies.
RoCE and iWARP

- Bring low-latency advantages of RDMA to Ethernet environment
  - RDMA over Converged Ethernet
    - InfiniBand transport protocol over Ethernet
    - iWARP: Internet Wide-Area RDMA Protocol
RDMA and wide-area networks

Table 1. RDMA over Ethernet solutions

<table>
<thead>
<tr>
<th>Solution</th>
<th>Protocols</th>
<th>Corresponding wide-area networking solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>RoCE</td>
<td>RDMA over InfiniBand transport protocol over Ethernet</td>
<td>Ethernet VLAN based virtual circuit</td>
</tr>
<tr>
<td>iWARP</td>
<td>RDMA over TCP/IP over Ethernet</td>
<td>IP-routed (datagram) path</td>
</tr>
</tbody>
</table>

- RoCE: Dynamic circuit services – provide wide-area VLAN/MPLS based virtual circuits (Ethernet over MPLS)
- iWARP: IP-routed service
Summary of technologies reviewed

- InfiniBand
- IETF TRILL
- IEEE 802.1aq: SPB
- Data center bridging
- RoCE and iWARP