Module 2: Ethernet Switching and Layer 2

Part 1: The Conceptual Lecture (The Why)

The Fundamental Problem

Imagine you have 10 computers in an office that need to communicate with each other. You could run a cable from every computer to every other computer, but that would require $\frac{n(n-1)}{2}$ cables - that's 45 cables for just 10 computers! This is completely impractical.

The Solution: Ethernet switching creates a shared communication medium where all devices connect to a central point (the switch) using just one cable each. The switch intelligently forwards messages between devices, like a postal sorting office that knows exactly where to send each letter.

What is Ethernet and Layer 2?

The networking world uses a 7-layer model (OSI Model) to organize how data moves:

```
Layer 7: Application    (Your email program)
Layer 6: Presentation   (Data formatting)
Layer 5: Session        (Conversations)
Layer 4: Transport      (TCP/UDP - reliable delivery)
Layer 3: Network        (IP addresses - routing between networks)
Layer 2: Data Link      (Ethernet - local delivery) <-- We are here!
Layer 1: Physical       (Cables and signals)
```

Layer 2 (Data Link) is responsible for moving data between directly connected devices. Think of it as the local delivery service in your neighborhood - it doesn't care about the city or country, just getting packages to the right house on your street.

How Ethernet Works

Every Ethernet device has a unique identifier called a **MAC address** (Media Access Control). It's like a serial number burned into the hardware:

```
Example MAC address: 00:1B:63:84:45:E6
(6 bytes, usually shown in hexadecimal)
```

An Ethernet frame (the Layer 2 "envelope") looks like this:

```
+----------------+----------------+------+----------+-----+
| Destination    | Source         | Type | Payload  | FCS |
| MAC Address    | MAC Address    |      | (Data)   |     |
| (6 bytes)      | (6 bytes)      | (2B) | (46-1500)| (4B)|
+----------------+----------------+------+----------+-----+
```

The Learning and Forwarding Process

A switch starts with zero knowledge about the network. It learns by observing:

1. **Learning Phase**: When a frame arrives on port 1 from MAC address A, the switch thinks: "Device A must be reachable through port 1!" It stores this in its MAC address table:

```
MAC Address Table:
+-----------------+--------+
| MAC Address     | Port   |
+-----------------+--------+
| 00:1B:63:84:45:A | Port 1 |
+-----------------+--------+
```

2. **Forwarding Decision**: When the switch receives a frame:
   - **Known destination**: Forward only to the specific port
   - **Unknown destination**: Flood to all ports except the incoming port (like shouting "Is anyone named Bob here?")
   - **Broadcast destination** (FF:FF:FF:FF:FF:FF): Send to all ports

Visualizing the Process

```
Initial State: Empty MAC table

     Switch
   +---------+
P1 |         | P2
---+         +---
   |         |
P3 |         | P4
---+         +---
   +---------+

Step 1: PC-A (on Port 1) sends to PC-B
Frame: [Dst: MAC-B | Src: MAC-A | Data]

Switch learns: MAC-A is on Port 1
Since MAC-B is unknown → Flood to P2, P3, P4

Step 2: PC-B (on Port 3) replies to PC-A
Frame: [Dst: MAC-A | Src: MAC-B | Data]

Switch learns: MAC-B is on Port 3
Since MAC-A is known (Port 1) → Forward only to P1

Final MAC Table:
+--------+--------+
| MAC-A  | Port 1 |
| MAC-B  | Port 3 |
+--------+--------+
```

Layer 2 Firewall Filters

Just like a security guard can check IDs at a building entrance, a Layer 2 firewall filter can inspect frames and decide whether to allow or block them based on:

- Source/Destination MAC addresses
- Ethernet type
- VLAN tags
- Incoming interface

Part 2: The Junos CLI Masterclass (The How)

Understanding the Junos Hierarchy

In Junos, Ethernet switching configuration lives under several hierarchies:

```
[edit interfaces]           # Physical interface settings
[edit vlans]               # VLAN definitions
[edit protocols]           # Layer 2 protocols
[edit firewall family ethernet-switching]  # L2 filters
```

Basic Switch Configuration

Let's build a simple 4-port switch configuration step by step:

Step 1: Configure Physical Interfaces

```
set interfaces ge-0/0/0 unit 0 family ethernet-switching
set interfaces ge-0/0/1 unit 0 family ethernet-switching
set interfaces ge-0/0/2 unit 0 family ethernet-switching
set interfaces ge-0/0/3 unit 0 family ethernet-switching
```

Why: This tells Junos these interfaces will participate in Layer 2 switching, not Layer 3 routing.

Step 2: Create a VLAN

```
set vlans OFFICE-LAN vlan-id 100
set vlans OFFICE-LAN interface ge-0/0/0.0
set vlans OFFICE-LAN interface ge-0/0/1.0
set vlans OFFICE-LAN interface ge-0/0/2.0
set vlans OFFICE-LAN interface ge-0/0/3.0
```

Why: VLANs create logical separation. All interfaces in the same VLAN can communicate at Layer 2.

Step 3: Configure MAC Address Learning (usually automatic)

```
set vlans OFFICE-LAN mac-table-size 4096
set vlans OFFICE-LAN mac-table-aging-time 300
```

Why:

- `mac-table-size`: Maximum entries to learn (default is usually sufficient)
- `mac-table-aging-time`: Seconds before forgetting unused MACs (default 300)

Step 4: Create a Layer 2 Firewall Filter

```
# Define the filter
set firewall family ethernet-switching filter BLOCK-GUEST term 1 from source-address 00:11:22:33:44:55/48
set firewall family ethernet-switching filter BLOCK-GUEST term 1 then discard
set firewall family ethernet-switching filter BLOCK-GUEST term 2 then accept

# Apply to an interface
set interfaces ge-0/0/0 unit 0 family ethernet-switching filter input BLOCK-GUEST
```

Why: This blocks all traffic from MAC addresses starting with 00:11:22 (perhaps a range assigned to guest devices).

Complete Reference Configuration

```
## Physical Interfaces
interfaces {
    ge-0/0/0 {
        unit 0 {
            family ethernet-switching {
                filter {
                    input BLOCK-GUEST;
                }
            }
        }
    }
    ge-0/0/1 {
        unit 0 {
            family ethernet-switching;
        }
    }
    ge-0/0/2 {
        unit 0 {
            family ethernet-switching;
        }
    }
    ge-0/0/3 {
        unit 0 {
            family ethernet-switching;
        }
    }
}

## VLAN Configuration
vlans {
    OFFICE-LAN {
        vlan-id 100;
```

```
        interface {
            ge-0/0/0.0;
            ge-0/0/1.0;
            ge-0/0/2.0;
            ge-0/0/3.0;
        }
        mac-table-aging-time 300;
    }
}

## Layer 2 Firewall Filter
firewall {
    family ethernet-switching {
        filter BLOCK-GUEST {
            term 1 {
                from {
                    source-address {
                        00:11:22:33:44:55/48;
                    }
                }
                then {
                    discard;
                }
            }
            term 2 {
                then {
                    accept;
                }
            }
        }
    }
}
```

Part 3: Verification & Troubleshooting (The What-If)

Essential Verification Commands

1. Check MAC Address Table

```
user@switch> show ethernet-switching table

MAC flags (S - static MAC, D - dynamic MAC, L - locally learned, P - Persistent static
           SE - statistics enabled, NM - non configured MAC, R - remote PE MAC, O - ovsdb MAC)

Ethernet switching table : 3 entries, 3 learned
Routing instance : default-switch
   Vlan                MAC                 MAC      Logical                Active
   name                address             flags    interface              source
   OFFICE-LAN          00:1b:63:84:45:e6   D        ge-0/0/0.0
   OFFICE-LAN          00:1b:63:84:45:e7   D        ge-0/0/1.0
   OFFICE-LAN          00:1b:63:84:45:e8   D        ge-0/0/2.0
```

What to look for:

- Entries are being learned (D = Dynamic)
- MACs appear on expected interfaces
- Table isn't full or showing errors

2. Check Interface Status

```
user@switch> show interfaces ge-0/0/0 extensive | match "Physical|Description|packets"
Physical interface: ge-0/0/0, Enabled, Physical link is Up
  Description: Connection to Office Printer
    Input packets : 18282937
    Output packets: 8371625
```

3. Check VLAN Configuration

```
user@switch> show vlans OFFICE-LAN

Routing instance        VLAN name            Tag          Interfaces
default-switch          OFFICE-LAN           100
                                                         ge-0/0/0.0*
                                                         ge-0/0/1.0*
                                                         ge-0/0/2.0*
                                                         ge-0/0/3.0*
```

Common Troubleshooting Scenarios

Scenario 1: Device Cannot Communicate

Symptom: PC on ge-0/0/0 cannot reach server on ge-0/0/2

Diagnostic Commands:

```
user@switch> show ethernet-switching table interface ge-0/0/0
(Check if source MAC is learned)

user@switch> show ethernet-switching table interface ge-0/0/2
(Check if destination MAC is learned)

user@switch> show firewall log
(Check if filter is blocking)
```

Cause: MAC address matches the firewall filter **Solution**:

```
delete firewall family ethernet-switching filter BLOCK-GUEST
# OR modify the filter to be more specific
set firewall family ethernet-switching filter BLOCK-GUEST term 1 from source-address 00:11:22:00:00:00/24
```

Scenario 2: MAC Table Fills Up

Symptom: New devices cannot communicate, logs show "MAC limit reached"

Diagnostic Commands:

```
user@switch> show ethernet-switching table summary
Global MAC count: 4096 (Limit: 4096)   <-- Table is full!

user@switch> show ethernet-switching mac-learning-log
```

Cause: MAC table size limit reached **Solution**:

```
set vlans OFFICE-LAN mac-table-size 8192
# OR reduce aging time to clear old entries faster
set vlans OFFICE-LAN mac-table-aging-time 60
```

Scenario 3: Loops Causing Broadcast Storm

Symptom: Network extremely slow, interface counters growing rapidly

Diagnostic Commands:

```
user@switch> show interfaces ge-0/0/0 extensive | match "bps|errors"
  Input rate     : 948582104 bps (125847 pps)   <-- Suspiciously high!
  Output rate    : 948582104 bps (125847 pps)
```

```
user@switch> show ethernet-switching table | count
Count: 4096 lines   <-- MAC table thrashing
```

Cause: Layer 2 loop (will be solved by STP in Module 6) **Temporary Solution**:

```
# Disable one of the redundant links
set interfaces ge-0/0/3 disable
commit
```

Scenario 4: Filter Not Working

Symptom: Filter configured but unwanted traffic still passing

Diagnostic Commands:

```
user@switch> show firewall filter BLOCK-GUEST

user@switch> show interfaces ge-0/0/0 extensive | match filter
  Filters: Input: BLOCK-GUEST-WRONG-NAME   <-- Wrong filter name!
```

Cause: Filter not applied or typo in name **Solution**:

```
delete interfaces ge-0/0/0 unit 0 family ethernet-switching filter
set interfaces ge-0/0/0 unit 0 family ethernet-switching filter input BLOCK-GUEST
commit
```

This foundation in Ethernet switching and Layer 2 operations forms the basis for all advanced switching features we'll explore in subsequent modules. Understanding how frames move through a switch and how MAC addresses are learned is crucial for troubleshooting any Layer 2 network issue.

Module 3: Virtual LANs and IRBs

Part 1: The Conceptual Lecture (The Why)

The Fundamental Problem VLANs Solve

Imagine a company with 3 departments sharing one floor: Sales, Engineering, and HR. If everyone connects to the same switch, several problems arise:

1. **Security**: Sales can see Engineering's prototype designs on the network
2. **Broadcast Storms**: HR's printer announcement reaches 300 computers instead of just 20
3. **Performance**: All departments compete for the same bandwidth
4. **Flexibility**: Moving someone to a different department requires rewiring

The Solution: Virtual LANs (VLANs) create multiple logical networks on the same physical infrastructure, like creating invisible walls in an open office.

What is a VLAN?

A VLAN is a logical grouping of devices that can communicate at Layer 2, regardless of their physical location. Think of it as creating multiple virtual switches inside one physical switch.

```
Without VLANs:                   With VLANs:
+-------------------+            +-------------------+
|                   |            | VLAN 10: Sales    |
|  One Big Network  |            +-------------------+
|   (All Devices)   |     →      | VLAN 20: Eng.     |
|                   |            +-------------------+
|                   |            | VLAN 30: HR       |
+-------------------+            +-------------------+
```

How VLAN Tagging Works

VLAN information is carried in a 4-byte "tag" inserted into the Ethernet frame:

```
Standard Ethernet Frame:
+-------------+-------------+------+---------+-----+
| Dst MAC     | Src MAC     | Type | Payload | FCS |
+-------------+-------------+------+---------+-----+

802.1Q Tagged Frame:
+-------------+-------------+----------+------+---------+-----+
| Dst MAC     | Src MAC     | 802.1Q   | Type | Payload | FCS |
|             |             | Tag(4B)  |      |         |     |
+-------------+-------------+----------+------+---------+-----+

802.1Q Tag Structure:
+-------+-----+--------+
| TPID  | PCP | VID    |
| (16b) | (3b)| (12b)  |
+-------+-----+--------+
 0x8100   QoS   VLAN ID (1-4094)
```

The magic happens at $2^{12} = 4096$ possible VLAN IDs (0 and 4095 are reserved).

Port Types in VLAN World

1. **Access Port**: Connects to end devices (PCs, printers)
    - Accepts untagged frames
    - Adds VLAN tag on ingress
    - Removes VLAN tag on egress
    - Belongs to exactly ONE VLAN
2. **Trunk Port**: Connects switches together
    - Carries multiple VLANs
    - Accepts and sends tagged frames
    - Can have one "native" VLAN (untagged)

```
Access vs Trunk Visualization:

PC ----[untagged]----> Access Port (VLAN 10) ----[tagged:10]----> Switch Core
                              ↓
                     (Switch adds tag 10)

Switch A ----[tagged:10,20,30]----> Trunk Port <----[tagged:10,20,30]---- Switch B
                                         ↓
                              (Switch preserves all tags)
```

MVRP: Automatic VLAN Administration

Problem: Manually configuring VLANs on 50 switches is error-prone and time-consuming.

Solution: Multiple VLAN Registration Protocol (MVRP) automatically propagates VLAN information between switches, like a game of "telephone" that actually works!

MVRP Process:

1. Admin creates VLAN 100 on Switch A
2. Switch A announces: "I have VLAN 100!"
3. Switch B hears this and thinks: "I should create VLAN 100 too!"
4. Switch B announces: "I also have VLAN 100!"
5. Process continues throughout the network

IRB: Integrated Routing and Bridging

The New Problem: VLANs isolate Layer 2 traffic perfectly... too perfectly! Now Sales (VLAN 10) cannot communicate with Engineering (VLAN 20) at all, even when they need to share files.

The Solution: IRB interfaces act as a gateway between VLANs, providing Layer 3 (IP) routing between Layer 2 domains.

```
Without IRB:                      With IRB:
VLAN 10 ←X→ VLAN 20              VLAN 10 ←→ IRB.10
(Cannot communicate)                  ↓       ↑
                                   Router
                                      ↓       ↑
                                 VLAN 20 ←→ IRB.20
                                 (Can route between VLANs)
```

Think of IRB as installing a door between rooms - the walls (VLANs) still exist, but now there's a controlled way to pass between them.

Part 2: The Junos CLI Masterclass (The How)

Understanding VLAN Configuration in Junos

Junos uses two main configuration styles for VLANs:

1. **Enterprise style** (ELS): Newer, used on EX/QFX switches
2. **Service Provider style**: Used on MX routers (our focus for JNCIE-SP)

Basic VLAN Configuration

Step 1: Create VLANs

```
set vlans SALES vlan-id 10
set vlans ENGINEERING vlan-id 20
set vlans HR vlan-id 30
```

Why: Each VLAN needs a name (for humans) and ID (for the protocol).

Step 2: Configure Access Ports

```
# Sales PC on port ge-0/0/1
set interfaces ge-0/0/1 unit 0 family ethernet-switching vlan members SALES

# Engineering Workstation on port ge-0/0/2
set interfaces ge-0/0/2 unit 0 family ethernet-switching vlan members ENGINEERING

# HR Printer on port ge-0/0/3
set interfaces ge-0/0/3 unit 0 family ethernet-switching vlan members HR
```

Why: Access ports need exactly one VLAN membership.

Step 3: Configure Trunk Ports

```
# Trunk to another switch on ge-0/0/10
set interfaces ge-0/0/10 unit 0 family ethernet-switching interface-mode trunk
set interfaces ge-0/0/10 unit 0 family ethernet-switching vlan members [ SALES ENGINEERING HR ]

# Alternative: Allow all VLANs
set interfaces ge-0/0/10 unit 0 family ethernet-switching vlan members all
```

Why: Trunk ports need to know which VLANs to accept/forward.

MVRP Configuration

Step 1: Enable MVRP Globally

```
set protocols mvrp
```

Step 2: Configure MVRP on Trunk Interfaces

```
set protocols mvrp interface ge-0/0/10
set protocols mvrp interface ge-0/0/11

# Optional: Set registration mode
set protocols mvrp interface ge-0/0/10 registration-mode normal
```

Registration modes:

- `normal`: Learn and advertise VLANs (default)
- `restricted`: Only register VLANs that exist locally
- `forbidden`: Do not register any VLANs

Step 3: Configure MVRP Timers (optional)

```
set protocols mvrp join-timer 200
set protocols mvrp leave-timer 600
set protocols mvrp leaveall-timer 10000
```

Why: Fine-tune how quickly MVRP reacts to changes (milliseconds).

IRB Configuration

Step 1: Create IRB Interfaces

```
set interfaces irb unit 10 family inet address 192.168.10.1/24
set interfaces irb unit 10 description "SALES VLAN Gateway"

set interfaces irb unit 20 family inet address 192.168.20.1/24
set interfaces irb unit 20 description "ENGINEERING VLAN Gateway"

set interfaces irb unit 30 family inet address 192.168.30.1/24
set interfaces irb unit 30 description "HR VLAN Gateway"
```

Why: Each IRB unit typically matches the VLAN ID for clarity.

Step 2: Bind IRB to VLANs

```
set vlans SALES l3-interface irb.10
set vlans ENGINEERING l3-interface irb.20
set vlans HR l3-interface irb.30
```

Why: This creates the Layer 2 to Layer 3 binding.

Step 3: Enable Routing (if needed)

```
set routing-options router-id 10.0.0.1
set protocols ospf area 0.0.0.0 interface irb.10
set protocols ospf area 0.0.0.0 interface irb.20
set protocols ospf area 0.0.0.0 interface irb.30
```

Complete Reference Configuration

```
## Interface Configuration
interfaces {
    ## Access Ports
    ge-0/0/1 {
        description "Sales PC";
        unit 0 {
            family ethernet-switching {
                vlan {
                    members SALES;
                }
            }
        }
    }
```

```
    ge-0/0/2 {
        description "Engineering Workstation";
        unit 0 {
            family ethernet-switching {
                vlan {
                    members ENGINEERING;
                }
            }
        }
    }
    ge-0/0/3 {
        description "HR Printer";
        unit 0 {
            family ethernet-switching {
                vlan {
                    members HR;
                }
            }
        }
    }

    ## Trunk Ports
    ge-0/0/10 {
        description "Trunk to Core Switch";
        unit 0 {
            family ethernet-switching {
                interface-mode trunk;
                vlan {
                    members [ SALES ENGINEERING HR ];
                }
            }
        }
    }

    ## IRB Interfaces
    irb {
        unit 10 {
            description "SALES VLAN Gateway";
            family inet {
                address 192.168.10.1/24;
            }
        }
        unit 20 {
            description "ENGINEERING VLAN Gateway";
            family inet {
                address 192.168.20.1/24;
            }
        }
        unit 30 {
            description "HR VLAN Gateway";
            family inet {
                address 192.168.30.1/24;
            }
        }
    }
}

## VLAN Configuration
vlans {
    SALES {
        description "Sales Department";
        vlan-id 10;
        l3-interface irb.10;
    }
    ENGINEERING {
        description "Engineering Department";
        vlan-id 20;
        l3-interface irb.20;
    }
    HR {
        description "Human Resources";
```

```
        vlan-id 30;
        l3-interface irb.30;
    }
}

## MVRP Configuration
protocols {
    mvrp {
        interface ge-0/0/10;
        interface ge-0/0/11;
    }
}
```

Part 3: Verification & Troubleshooting (The What-If)

Essential Verification Commands

1. Verify VLAN Configuration

```
user@switch> show vlans

Routing instance        VLAN name           Tag          Interfaces
default-switch          SALES               10
                                                        ge-0/0/1.0*
                                                        ge-0/0/10.0*
default-switch          ENGINEERING         20
                                                        ge-0/0/2.0*
                                                        ge-0/0/10.0*
default-switch          HR                  30
                                                        ge-0/0/3.0*
                                                        ge-0/0/10.0*
```

What to look for:

- Each VLAN shows correct tag (VLAN ID)
- Interfaces marked with * are active
- Both access and trunk ports appear in correct VLANs

2. Verify Trunk Port Status

```
user@switch> show ethernet-switching interfaces ge-0/0/10

Routing Instance Name    : default-switch
Logical Interface Name   : ge-0/0/10.0
Interface State          : up
Administrative State     : up
Interface Mode           : TRUNK
Vlan Members             : SALES(10) ENGINEERING(20) HR(30)
Number of MACs learned   : 47
```

3. Check MVRP Status

```
user@switch> show mvrp statistics interface ge-0/0/10

MVRP statistics for interface ge-0/0/10
    MVRP PDUs sent:            1847
    MVRP PDUs received:        2341
    MVRP Join Empty sent:      423
    MVRP Join In sent:         892
    MVRP Leave sent:           89
    MVRP registrations:        3
```

4. Verify IRB Interfaces

```
user@switch> show interfaces irb terse
Interface               Admin Link Proto    Local                 Remote
irb.10                  up    up   inet     192.168.10.1/24
irb.20                  up    up   inet     192.168.20.1/24
irb.30                  up    up   inet     192.168.30.1/24
```

Common Troubleshooting Scenarios

Scenario 1: VLAN Traffic Not Passing

Symptom: PC in VLAN 10 cannot reach another device in same VLAN

Diagnostic Commands:

```
user@switch> show vlans SALES extensive

user@switch> show ethernet-switching table vlan-name SALES

user@switch> show interfaces ge-0/0/1 extensive | match "error|drop"
```

Cause: Interface not in correct VLAN **Solution**:

```
# Verify interface membership
show configuration interfaces ge-0/0/1

# Fix if needed
set interfaces ge-0/0/1 unit 0 family ethernet-switching vlan members SALES
commit
```

Scenario 2: Trunk Not Carrying All VLANs

Symptom: Only some VLANs work across trunk link

Diagnostic Commands:

```
user@switch> show ethernet-switching interfaces ge-0/0/10 detail
Interface Mode: TRUNK
Vlan Members: SALES(10) ENGINEERING(20)    <-- HR(30) missing!

user@switch> monitor traffic interface ge-0/0/10 layer2-headers
(Look for 802.1Q tags in frames)
```

Cause: VLAN not added to trunk **Solution**:

```
set interfaces ge-0/0/10 unit 0 family ethernet-switching vlan members HR
commit
```

Scenario 3: MVRP Not Propagating VLANs

Symptom: VLANs created on one switch don't appear on neighbors

Diagnostic Commands:

```
user@switch> show mvrp dynamic-vlan-memberships
(Should show VLANs learned via MVRP)

user@switch> show log messages | match MVRP
Nov 10 14:23:11 mvrpd[1234]: MVRP_PORT_DOWN: ge-0/0/10 down
```

Cause: MVRP not enabled on interface or protocol **Solution**:

```
# Enable MVRP
set protocols mvrp
set protocols mvrp interface ge-0/0/10
commit

# Verify
show mvrp interface
```

Scenario 4: IRB Not Routing Between VLANs

Symptom: Devices cannot ping across VLANs despite IRB configuration

Diagnostic Commands:

```
user@switch> show interfaces irb.10
Logical interface irb.10 (Index 123) (SNMP ifIndex 567)
    Flags: Up SNMP-Traps 0x4004000 Encapsulation: ENET2
    Bandwidth: 1000mbps
    Routing Instance: None Bridging Domain: None
                          ↑ Problem: Not bound to VLAN!

user@switch> show route table inet.0 192.168.10.0/24
(Should show as "Direct" route via irb.10)
```

Cause: IRB not bound to VLAN or IP configuration error **Solution**:

```
# Bind IRB to VLAN
set vlans SALES l3-interface irb.10
commit

# Verify hosts have correct default gateway
# On PC: 192.168.10.1 for VLAN 10 devices
```

Scenario 5: Native VLAN Mismatch

Symptom: Untagged traffic not working on trunk

Diagnostic Commands:

```
user@switch> show configuration interfaces ge-0/0/10 | match native
native-vlan-id 99;

# Check neighbor switch - if different, that's the problem!
```

Cause: Native VLAN mismatch between switches **Solution**:

```
# Make native VLAN consistent
set interfaces ge-0/0/10 native-vlan-id 1
# Or remove native VLAN entirely
delete interfaces ge-0/0/10 native-vlan-id
commit
```

Pro Tips for VLAN Troubleshooting

1. **Always check both ends** of a trunk link - mismatches are common
2. **Use VLAN 1 carefully** - it's often the default native VLAN
3. **Document your VLAN scheme** - consistency prevents errors
4. **Test with untagged traffic first** - simpler to troubleshoot
5. **Remember VLAN 0 and 4095 are reserved** - cannot be used

This module has equipped you with the fundamental understanding of VLANs - how they segment networks, how tagging works, and how IRBs bridge the Layer 2/Layer 3 gap. These concepts form the foundation for more advanced topics like Virtual Switches and Provider Bridging in the upcoming modules.

Module 4: Virtual Switches

Part 1: The Conceptual Lecture (The Why)

The Fundamental Problem

Imagine you're a service provider with one physical MX router, and three different customers want to run their own independent networks through your infrastructure. Each customer demands:

- Complete isolation from other customers
- Their own VLAN numbering scheme (Customer A and B both want to use VLAN 10)
- Independent MAC address tables
- Separate spanning-tree domains

Traditional Solution: Buy three separate switches (expensive and wasteful!)

Modern Solution: Virtual switches create multiple, completely independent Layer 2 domains within a single physical device, like running multiple virtual machines on one server.

Understanding Routing Instances

Before diving into virtual switches, we must understand **routing instances** - the foundation of virtualization in Junos.

A routing instance is a collection of routing tables, interfaces, and routing protocol parameters. Think of it as a separate logical router within your physical router.

```
Physical Router
├── Default Instance (inet.0, inet6.0)
├── Customer-A Instance
│   ├── customer-a.inet.0
│   └── customer-a.inet6.0
├── Customer-B Instance
│   ├── customer-b.inet.0
│   └── customer-b.inet6.0
└── Management Instance
    └── mgmt.inet.0
```

Types of Routing Instances

1. **Virtual Router (VR)**
   - Complete Layer 3 separation
   - Independent routing protocols
   - Separate routing tables
   - Like having multiple routers in one box
2. **Virtual Switch (VS)**
   - Complete Layer 2 separation
   - Independent VLAN space
   - Separate MAC tables
   - Like having multiple switches in one box
3. **VRF (Virtual Routing and Forwarding)**
   - Lite version of virtual router
   - Primarily for MPLS L3VPNs
   - Shares routing protocol processes
4. **VPLS (Virtual Private LAN Service)**
   - For Layer 2 VPNs across MPLS
   - Connects multiple sites at Layer 2

Virtual Switch Architecture

A virtual switch instance contains:

- **Bridge domains** (similar to VLANs)
- **MAC address table** (independent learning)
- **Interfaces** (logical assignments)
- **IRB interfaces** (for Layer 3 gateway functionality)

```
Physical Switch Architecture:
┌─────────────────────────────────────┐
│          Physical Device            │
├─────────────────────────────────────┤
│  Default Switch                     │
│  ├── VLAN 10: Sales                 │
│  ├── VLAN 20: Engineering           │
│  └── MAC Table (shared)             │
└─────────────────────────────────────┘

Virtual Switch Architecture:
┌─────────────────────────────────────┐
│          Physical Device            │
├─────────────────────────────────────┤
│  VS: CUSTOMER-A                     │
│  ├── BD: CUST-A-VLAN10              │
│  ├── BD: CUST-A-VLAN20              │
│  └── MAC Table A (isolated)         │
├─────────────────────────────────────┤
│  VS: CUSTOMER-B                     │
│  ├── BD: CUST-B-VLAN10              │
│  ├── BD: CUST-B-VLAN20              │
│  └── MAC Table B (isolated)         │
└─────────────────────────────────────┘
```

Bridge Domains vs VLANs

In virtual switches, **bridge domains** replace traditional VLANs:

- Each bridge domain is like a VLAN
- Can use VLAN IDs or be ID-free
- Completely isolated between virtual switches
- Support same features (learning, flooding, filtering)

Interconnecting Routing Instances

Sometimes isolated instances need to communicate. Junos provides several methods:

1. **Logical Tunnel (lt-) Interfaces**
   - Creates virtual "cable" between instances
   - Bidirectional peer (lt-0/0/0.0 ↔ lt-0/0/0.1)
   - Uses bandwidth but provides complete control
2. **RIB Groups**
   - Shares routes between routing tables
   - No traffic forwarding, just route sharing
   - Efficient for route leaking
3. **Virtual Router Interfaces**
   - Direct connection without physical resources
   - Supported on newer platforms

Logical Systems: The Ultimate Isolation

Logical systems take virtualization even further:

- Completely separate configuration files
- Independent management access
- Resource allocation (CPU, memory)
- Like having multiple physical routers

```
Physical Router
├── Logical System: CORE
│   ├── Complete Junos config
│   ├── Own users/authentication
│   └── Resource limits
├── Logical System: EDGE
│   ├── Complete Junos config
│   ├── Own users/authentication
│   └── Resource limits
```

Part 2: The Junos CLI Masterclass (The How)

Basic Virtual Switch Configuration

Step 1: Create Routing Instance

```
set routing-instances CUSTOMER-A instance-type virtual-switch
set routing-instances CUSTOMER-B instance-type virtual-switch
```

Why: Define the instance type as virtual-switch for Layer 2 functionality.

Step 2: Create Bridge Domains

```
# For Customer A
set routing-instances CUSTOMER-A bridge-domains SALES vlan-id 10
set routing-instances CUSTOMER-A bridge-domains ENGINEERING vlan-id 20

# For Customer B (can reuse same VLAN IDs!)
set routing-instances CUSTOMER-B bridge-domains RETAIL vlan-id 10
set routing-instances CUSTOMER-B bridge-domains WAREHOUSE vlan-id 20
```

Why: Bridge domains provide VLAN-like functionality within each virtual switch.

Step 3: Assign Interfaces

```
# Customer A interfaces
set interfaces ge-0/0/0 unit 0 family bridge interface-mode access
set interfaces ge-0/0/0 unit 0 family bridge vlan-id 10
set routing-instances CUSTOMER-A interface ge-0/0/0.0

set interfaces ge-0/0/1 unit 0 family bridge interface-mode access
set interfaces ge-0/0/1 unit 0 family bridge vlan-id 20
set routing-instances CUSTOMER-A interface ge-0/0/1.0

# Customer B interfaces
set interfaces ge-0/0/2 unit 0 family bridge interface-mode access
set interfaces ge-0/0/2 unit 0 family bridge vlan-id 10
set routing-instances CUSTOMER-B interface ge-0/0/2.0
```

Why: Interfaces must be configured for bridge family and assigned to instances.

Step 4: Configure Trunk Interfaces

```
# Trunk for Customer A
set interfaces ge-0/0/10 flexible-vlan-tagging
set interfaces ge-0/0/10 encapsulation flexible-ethernet-services
set interfaces ge-0/0/10 unit 100 vlan-id 100
set interfaces ge-0/0/10 unit 100 family bridge interface-mode trunk
```

```
set interfaces ge-0/0/10 unit 100 family bridge vlan-id-list [10 20]
set routing-instances CUSTOMER-A interface ge-0/0/10.100
```

Why: Flexible VLAN tagging allows multiple logical interfaces on one physical port.

Adding IRB Interfaces to Virtual Switches

```
# Create IRB interface
set interfaces irb unit 100 family inet address 192.168.10.1/24
set interfaces irb unit 100 description "Customer A Sales Gateway"

# Assign to bridge domain
set routing-instances CUSTOMER-A bridge-domains SALES routing-interface irb.100

# Assign IRB to routing instance
set routing-instances CUSTOMER-A interface irb.100
```

Interconnecting Instances with Logical Tunnels

Step 1: Create Logical Tunnel Interface

```
set interfaces lt-0/0/0 unit 0 encapsulation ethernet
set interfaces lt-0/0/0 unit 0 peer-unit 1
set interfaces lt-0/0/0 unit 0 family bridge interface-mode trunk
set interfaces lt-0/0/0 unit 0 family bridge vlan-id-list [100 200]

set interfaces lt-0/0/0 unit 1 encapsulation ethernet
set interfaces lt-0/0/0 unit 1 peer-unit 0
set interfaces lt-0/0/0 unit 1 family bridge interface-mode trunk
set interfaces lt-0/0/0 unit 1 family bridge vlan-id-list [100 200]
```

Step 2: Assign to Different Instances

```
set routing-instances CUSTOMER-A interface lt-0/0/0.0
set routing-instances SHARED-SERVICES interface lt-0/0/0.1
```

Configuring Logical Systems

Step 1: Create Logical System

```
set logical-systems CUSTOMER-A-LS
set logical-systems CUSTOMER-B-LS
```

Step 2: Assign Interfaces to Logical Systems

```
set interfaces ge-0/0/0 unit 0 logical-system CUSTOMER-A-LS
set interfaces ge-0/0/1 unit 0 logical-system CUSTOMER-A-LS
```

Step 3: Configure Within Logical System

```
set logical-systems CUSTOMER-A-LS routing-instances VS-1 instance-type virtual-switch
set logical-systems CUSTOMER-A-LS routing-instances VS-1 interface ge-0/0/0.0
set logical-systems CUSTOMER-A-LS routing-instances VS-1 bridge-domains BD-100 vlan-id 100
```

Complete Reference Configuration

```
## Interface Configuration
interfaces {
    ## Customer A Interfaces
    ge-0/0/0 {
        unit 0 {
            description "Customer A - Sales Dept";
            family bridge {
                interface-mode access;
```

```
            vlan-id 10;
        }
    }
}
ge-0/0/1 {
    unit 0 {
        description "Customer A - Engineering";
        family bridge {
            interface-mode access;
            vlan-id 20;
        }
    }
}

## Customer B Interfaces
ge-0/0/2 {
    unit 0 {
        description "Customer B - Retail";
        family bridge {
            interface-mode access;
            vlan-id 10;  ## Same VLAN ID as Customer A!
        }
    }
}

## Trunk Interface with Flexible Tagging
ge-0/0/10 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    unit 100 {
        description "Customer A Trunk";
        vlan-id 100;  ## Outer tag
        family bridge {
            interface-mode trunk;
            vlan-id-list [ 10 20 ];  ## Inner tags
        }
    }
    unit 200 {
        description "Customer B Trunk";
        vlan-id 200;  ## Different outer tag
        family bridge {
            interface-mode trunk;
            vlan-id-list [ 10 20 ];
        }
    }
}

## IRB Interfaces
irb {
    unit 100 {
        family inet {
            address 192.168.10.1/24;
        }
    }
    unit 110 {
        family inet {
            address 192.168.20.1/24;
        }
    }
    unit 200 {
        family inet {
            address 172.16.10.1/24;
        }
    }
}

## Logical Tunnel for Interconnection
lt-0/0/0 {
    unit 0 {
        encapsulation ethernet;
        peer-unit 1;
```

```
            family bridge {
                interface-mode trunk;
                vlan-id-list [ 300 400 ];
            }
        }
        unit 1 {
            encapsulation ethernet;
            peer-unit 0;
            family bridge {
                interface-mode trunk;
                vlan-id-list [ 300 400 ];
            }
        }
    }
}

## Routing Instances Configuration
routing-instances {
    CUSTOMER-A {
        instance-type virtual-switch;
        interface ge-0/0/0.0;
        interface ge-0/0/1.0;
        interface ge-0/0/10.100;
        interface irb.100;
        interface irb.110;
        interface lt-0/0/0.0;
        bridge-domains {
            SALES {
                vlan-id 10;
                routing-interface irb.100;
            }
            ENGINEERING {
                vlan-id 20;
                routing-interface irb.110;
            }
            INTERCONNECT {
                vlan-id 300;
            }
        }
    }
    CUSTOMER-B {
        instance-type virtual-switch;
        interface ge-0/0/2.0;
        interface ge-0/0/10.200;
        interface irb.200;
        bridge-domains {
            RETAIL {
                vlan-id 10;  ## Same as Customer A - no conflict!
                routing-interface irb.200;
            }
            WAREHOUSE {
                vlan-id 20;
            }
        }
    }
    SHARED-SERVICES {
        instance-type virtual-switch;
        interface lt-0/0/0.1;
        bridge-domains {
            SHARED-VLAN {
                vlan-id 300;
            }
        }
    }
}
```

Part 3: Verification & Troubleshooting (The What-If)

Essential Verification Commands

1. View Routing Instances

```
user@router> show route instance
Instance             Type
  Primary rib
CUSTOMER-A           virtual-switch
CUSTOMER-B           virtual-switch
SHARED-SERVICES      virtual-switch
```

2. Check Bridge Domains in Virtual Switch

```
user@router> show bridge domain routing-instance CUSTOMER-A

Routing instance        Bridge domain            VLAN ID     Interfaces
CUSTOMER-A              SALES                    10
                                                            ge-0/0/0.0
                                                            ge-0/0/10.100
CUSTOMER-A              ENGINEERING              20
                                                            ge-0/0/1.0
                                                            ge-0/0/10.100
```

3. View MAC Table for Specific Instance

```
user@router> show bridge mac-table routing-instance CUSTOMER-A

MAC flags (S -static MAC, D -dynamic MAC, L -locally learned, C -Control MAC
    SE -Statistics enabled, NM -Non configured MAC, R -Remote PE MAC)

Routing instance : CUSTOMER-A
 Bridging domain : SALES, VLAN : 10
   MAC                 MAC      Logical          NH     RTR
   address             flags    interface        Index  ID
   00:1a:2b:3c:4d:5e   D        ge-0/0/0.0
   00:1a:2b:3c:4d:5f   D        ge-0/0/10.100
```

4. Check Logical Tunnel Status

```
user@router> show interfaces lt-0/0/0
Physical interface: lt-0/0/0, Enabled, Physical link is Up
  Interface index: 128, SNMP ifIndex: 512
  Type: Logical-tunnel, Link-level type: Logical, MTU: Unlimited

  Logical interface lt-0/0/0.0 (Index 334) (SNMP ifIndex 613)
    Flags: Up SNMP-Traps 0x4000 Encapsulation: Ethernet
    Peer unit: 1
```

Common Troubleshooting Scenarios

Scenario 1: Traffic Not Passing Between Instances

Symptom: Customer A cannot reach shared services

Diagnostic Commands:

```
user@router> show bridge mac-table routing-instance CUSTOMER-A interface lt-0/0/0.0
(Check if MACs are learned on logical tunnel)

user@router> show interfaces lt-0/0/0.0 statistics
(Look for increasing packet counters)
```

Cause: Logical tunnel not properly configured **Solution**:

```
# Verify peer units are correct
show configuration interfaces lt-0/0/0

# Ensure both units are assigned to instances
set routing-instances CUSTOMER-A interface lt-0/0/0.0
set routing-instances SHARED-SERVICES interface lt-0/0/0.1
commit
```

Scenario 2: VLAN ID Conflict Not Isolated

Symptom: Customer A VLAN 10 traffic appearing in Customer B

Diagnostic Commands:

```
user@router> show bridge statistics routing-instance CUSTOMER-A
user@router> show bridge statistics routing-instance CUSTOMER-B

user@router> monitor traffic interface ge-0/0/2.0 layer2-headers
(Check VLAN tags in frames)
```

Cause: Interface assigned to wrong routing instance **Solution**:

```
# Check interface assignment
show configuration routing-instances | match ge-0/0/2.0

# Move to correct instance
delete routing-instances CUSTOMER-A interface ge-0/0/2.0
set routing-instances CUSTOMER-B interface ge-0/0/2.0
commit
```

Scenario 3: IRB Not Working in Virtual Switch

Symptom: Cannot ping IRB interface from devices in bridge domain

Diagnostic Commands:

```
user@router> show interfaces irb.100 routing-instance CUSTOMER-A
error: interface irb.100 not configured under routing-instance

user@router> show bridge domain SALES detail routing-instance CUSTOMER-A
Routing interface: None     <-- Problem!
```

Cause: IRB not properly bound to both routing instance and bridge domain **Solution**:

```
# Add IRB to routing instance
set routing-instances CUSTOMER-A interface irb.100

# Bind to bridge domain
set routing-instances CUSTOMER-A bridge-domains SALES routing-interface irb.100
commit
```

Scenario 4: Flexible VLAN Tagging Issues

Symptom: Double-tagged traffic not working on trunk

Diagnostic Commands:

```
user@router> show interfaces ge-0/0/10 | match "tagging|encap"
  Flexible-vlan-tagging: Disabled    <-- Problem!

user@router> show configuration interfaces ge-0/0/10.100
```

```
vlan-id 100;
## Missing inner VLAN configuration!
```

Cause: Interface not configured for flexible tagging **Solution**:

```
set interfaces ge-0/0/10 flexible-vlan-tagging
set interfaces ge-0/0/10 encapsulation flexible-ethernet-services
set interfaces ge-0/0/10 unit 100 vlan-id 100
set interfaces ge-0/0/10 unit 100 family bridge vlan-id-list [10 20]
commit
```

Scenario 5: Logical System Isolation Failure

Symptom: Can see configuration from another logical system

Diagnostic Commands:

```
user@router> set cli logical-system CUSTOMER-A-LS
Logical system: CUSTOMER-A-LS

user@CUSTOMER-A-LS> show configuration
## Should only see CUSTOMER-A config

user@CUSTOMER-A-LS> show interfaces terse | except ".32767"
## Should only see interfaces assigned to this LS
```

Cause: Interface unit not assigned to logical system **Solution**:

```
# From main system
set interfaces ge-0/0/5 unit 0 logical-system CUSTOMER-A-LS

# Or entire interface
set logical-systems CUSTOMER-A-LS interfaces ge-0/0/5 unit 0 family bridge
```

Pro Tips for Virtual Switch Management

1. **Naming Convention**: Use clear prefixes (CUST-A-VLAN10 vs just VLAN10)
2. **Document Overlaps**: Track where VLAN IDs are reused across instances
3. **Monitor Resources**: Each instance consumes memory for MAC tables
4. **Test Isolation**: Regularly verify traffic doesn't leak between instances
5. **Plan Interconnects**: Design logical tunnel topology before implementation

This module has shown how virtual switches provide complete Layer 2 isolation within a single physical device. Combined with routing instances and logical systems, you can build complex multi-tenant environments that maintain security and independence while maximizing hardware utilization. The next module will build on this with Provider Bridging for service provider deployments.

Module 5: Provider Bridging

Part 1: The Conceptual Lecture (The Why)

The Fundamental Problem

Imagine you're a service provider connecting multiple customer sites. Customer A has three offices using VLANs 10, 20, and 30 internally. When they send traffic through your network, you face several challenges:

1. **VLAN ID Conflicts**: Multiple customers want to use VLAN 10
2. **VLAN Exhaustion**: You only have 4094 VLAN IDs but thousands of customers
3. **Customer Transparency**: Customers expect their VLANs to work exactly as if sites were directly connected
4. **Scalability**: Your core switches would need to learn millions of customer MAC addresses

The Solution: Provider Bridging (IEEE 802.1ad) adds a second VLAN tag, creating a hierarchy where customer VLANs are encapsulated inside service provider VLANs.

Understanding VLAN Stacking (Q-in-Q)

VLAN stacking adds an outer "Service VLAN" (S-VLAN) tag while preserving the inner "Customer VLAN" (C-VLAN) tag:

```
Standard 802.1Q Frame:
+----------+----------+--------+------+---------+
| Dst MAC  | Src MAC  | C-TAG  | Type | Payload |
+----------+----------+--------+------+---------+
                       ↑
                    VLAN 10

Provider Bridging 802.1ad Frame:
+----------+----------+--------+--------+------+---------+
| Dst MAC  | Src MAC  | S-TAG  | C-TAG  | Type | Payload |
+----------+----------+--------+--------+------+---------+
                       ↑        ↑
                   VLAN 1000  VLAN 10
                   (Provider) (Customer)

S-TAG Structure:
+--------+-----+------+
| TPID   | PCP | VID  |
| 0x88a8 | 3b  | 12b  |
+--------+-----+------+
(Different from C-TAG TPID: 0x8100)
```

IEEE VLAN Stacking Models

1. Port-Based Service Interface

- Simplest model
- All customer traffic gets same S-VLAN
- No C-VLAN inspection
- "Whatever comes in port X gets S-VLAN 1000"

```
Customer A, Port 1 → S-VLAN 1000
├── C-VLAN 10 → [1000][10]
├── C-VLAN 20 → [1000][20]
└── Untagged  → [1000][none]
```

2. C-VLAN Based Service Interface

- Maps specific C-VLANs to S-VLANs
- More granular control
- Can split customer VLANs across different services

```
Customer A, Port 1:
├── C-VLAN 10 → S-VLAN 1000 → [1000][10]
├── C-VLAN 20 → S-VLAN 2000 → [2000][20]
└── C-VLAN 30 → S-VLAN 3000 → [3000][30]
```

3. Selective Q-in-Q

- Mix of both models
- Some VLANs get stacked, others pass through
- Used for managed services

Provider Bridging Components

1. Customer Edge (CE)

- Customer's equipment
- Sends standard 802.1Q frames
- Unaware of provider network

2. Provider Edge (PE)

- Service provider's edge device
- Adds/removes S-VLAN tags
- Maintains customer MAC table

3. Provider Core (P)

- Only looks at S-VLAN tags
- Doesn't learn customer MACs
- Scales to millions of customers

```
Network Topology:
CE-A1 --- PE1 ---     --- PE2 --- CE-A2
VLAN 10      \   P-Core  /         VLAN 10
              \         /
               PE3----PE4
                 |      |
              CE-A3  CE-B1
              VLAN 10 VLAN 10

PE1 adds S-VLAN 1000 for Customer A
PE3 adds S-VLAN 2000 for Customer B
Core only sees S-VLANs, not customer VLANs
```

MAC Address Learning in Provider Networks

Provider bridging uses hierarchical MAC learning:

1. **Customer MAC Learning**: PE devices learn customer MACs per S-VLAN
2. **Provider MAC Learning**: P devices only learn PE MAC addresses
3. **MAC Limiting**: Protect against MAC floods from customers

```
PE1 MAC Table:
S-VLAN 1000:
├── 00:11:11:11:11:11 (CE-A1) → Local
├── 00:22:22:22:22:22 (CE-A2) → Via PE2
└── 00:33:33:33:33:33 (CE-A3) → Via PE3

P-Core MAC Table:
S-VLAN 1000:
├── PE1-MAC → Port 1
├── PE2-MAC → Port 2
└── PE3-MAC → Port 3
(No customer MACs!)
```

VLAN Translation vs VLAN Stacking

Sometimes you need to change VLAN IDs rather than stack them:

VLAN Translation: Changes the VLAN ID

- Customer sends VLAN 10 → Provider translates to VLAN 1000
- One tag in, one tag out

VLAN Stacking: Adds additional VLAN tag

- Customer sends VLAN 10 → Provider adds VLAN 1000 → [1000][10]
- One tag in, two tags out

Part 2: The Junos CLI Masterclass (The How)

Basic Provider Bridging Configuration

Step 1: Configure Service Provider VLANs

```
# Define S-VLANs for different customers
set vlans S-VLAN-1000 vlan-id 1000
set vlans S-VLAN-1000 description "Customer A - All Sites"

set vlans S-VLAN-2000 vlan-id 2000
set vlans S-VLAN-2000 description "Customer B - All Sites"
```

Step 2: Configure Customer-Facing Interfaces (PE)

```
# Customer A - Site 1 (Port-based Q-in-Q)
set interfaces ge-0/0/0 flexible-vlan-tagging
set interfaces ge-0/0/0 encapsulation extended-vlan-bridge
set interfaces ge-0/0/0 unit 1000 vlan-id 1000
set interfaces ge-0/0/0 unit 1000 input-vlan-map push
set interfaces ge-0/0/0 unit 1000 output-vlan-map pop
```

Why:

- `flexible-vlan-tagging` : Enables multiple VLAN operations
- `extended-vlan-bridge` : Supports Q-in-Q operations
- `input-vlan-map push` : Adds S-VLAN on ingress
- `output-vlan-map pop` : Removes S-VLAN on egress

Step 3: Configure C-VLAN Based Mapping

```
# Map specific customer VLANs to S-VLANs
set interfaces ge-0/0/1 flexible-vlan-tagging
set interfaces ge-0/0/1 encapsulation extended-vlan-bridge

# Customer VLAN 10 → S-VLAN 1000
set interfaces ge-0/0/1 unit 10 vlan-id-list 10
set interfaces ge-0/0/1 unit 10 input-vlan-map push vlan-id 1000
set interfaces ge-0/0/1 unit 10 output-vlan-map pop

# Customer VLAN 20 → S-VLAN 2000
set interfaces ge-0/0/1 unit 20 vlan-id-list 20
set interfaces ge-0/0/1 unit 20 input-vlan-map push vlan-id 2000
set interfaces ge-0/0/1 unit 20 output-vlan-map pop
```

Step 4: Configure Core-Facing Interfaces

```
# Core interfaces only handle S-VLANs
set interfaces ge-0/0/10 flexible-vlan-tagging
set interfaces ge-0/0/10 encapsulation extended-vlan-bridge
set interfaces ge-0/0/10 unit 1000 vlan-id 1000
set interfaces ge-0/0/10 unit 2000 vlan-id 2000

# Add to bridge domains
set vlans S-VLAN-1000 interface ge-0/0/10.1000
set vlans S-VLAN-2000 interface ge-0/0/10.2000
```

Advanced VLAN Operations

VLAN Translation (Swap)

```
# Customer sends VLAN 99, translate to S-VLAN 1000
set interfaces ge-0/0/2 unit 99 vlan-id 99
set interfaces ge-0/0/2 unit 99 input-vlan-map swap vlan-id 1000
set interfaces ge-0/0/2 unit 99 output-vlan-map swap vlan-id 99
```

All-in-One Q-in-Q

```
# Push S-VLAN 1000 on all customer VLANs
set interfaces ge-0/0/3 flexible-vlan-tagging
set interfaces ge-0/0/3 encapsulation extended-vlan-bridge
set interfaces ge-0/0/3 unit 0 vlan-id 1000
set interfaces ge-0/0/3 unit 0 family bridge interface-mode trunk
set interfaces ge-0/0/3 unit 0 family bridge inner-vlan-id-list 1-4094
```

VLAN Rewriting with Range

```
# Rewrite C-VLAN range 100-199 to S-VLAN 1000
set interfaces ge-0/0/4 unit 100 vlan-id-range 100-199
set interfaces ge-0/0/4 unit 100 input-vlan-map push vlan-id 1000
set interfaces ge-0/0/4 unit 100 output-vlan-map pop
```

Provider Bridging with Bridge Domains

```
# Create bridge domain for S-VLAN
set bridge-domains CUSTOMER-A vlan-id 1000
set bridge-domains CUSTOMER-A interface ge-0/0/0.1000  # Customer facing
set bridge-domains CUSTOMER-A interface ge-0/0/10.1000 # Core facing

# Enable MAC learning limits
set bridge-domains CUSTOMER-A bridge-options interface ge-0/0/0.1000 mac-limit 1000
set bridge-domains CUSTOMER-A bridge-options interface ge-0/0/0.1000 mac-limit packet-action drop
```

Complete Reference Configuration

```
## Interface Configuration
interfaces {
    ## Customer A - Site 1 (Simple Q-in-Q)
    ge-0/0/0 {
        description "Customer A - Site 1";
        flexible-vlan-tagging;
        encapsulation extended-vlan-bridge;
        unit 1000 {
            description "All Customer A VLANs";
            vlan-id 1000;
            input-vlan-map push;
            output-vlan-map pop;
            family bridge {
                interface-mode trunk;
            }
        }
    }

    ## Customer A - Site 2 (Selective Q-in-Q)
    ge-0/0/1 {
        description "Customer A - Site 2";
        flexible-vlan-tagging;
        encapsulation extended-vlan-bridge;
        unit 10 {
            description "Customer VLAN 10 → S-VLAN 1000";
            vlan-id-list 10;
            input-vlan-map push vlan-id 1000;
            output-vlan-map pop;
            family bridge;
        }
        unit 20 {
            description "Customer VLAN 20 → S-VLAN 1001";
            vlan-id-list 20;
```

```
            input-vlan-map push vlan-id 1001;
            output-vlan-map pop;
            family bridge;
        }
    }

    ## Customer B (VLAN Translation)
    ge-0/0/2 {
        description "Customer B - VLAN Translation";
        flexible-vlan-tagging;
        encapsulation extended-vlan-bridge;
        unit 99 {
            description "Translate C-VLAN 99 to S-VLAN 2000";
            vlan-id 99;
            input-vlan-map swap vlan-id 2000;
            output-vlan-map swap vlan-id 99;
            family bridge;
        }
    }

    ## Core Interfaces
    ge-0/0/10 {
        description "To Provider Core";
        flexible-vlan-tagging;
        encapsulation extended-vlan-bridge;
        unit 1000 {
            description "S-VLAN 1000 - Customer A";
            vlan-id 1000;
            family bridge;
        }
        unit 1001 {
            description "S-VLAN 1001 - Customer A VLAN 20";
            vlan-id 1001;
            family bridge;
        }
        unit 2000 {
            description "S-VLAN 2000 - Customer B";
            vlan-id 2000;
            family bridge;
        }
    }
}

## Bridge Domains (Alternative to VLANs)
bridge-domains {
    CUSTOMER-A-PRIMARY {
        description "Customer A Primary Services";
        vlan-id 1000;
        interface ge-0/0/0.1000;    ## Site 1
        interface ge-0/0/1.10;      ## Site 2 VLAN 10
        interface ge-0/0/10.1000;   ## Core
        bridge-options {
            interface ge-0/0/0.1000 {
                mac-limit 1000;
                mac-limit packet-action drop;
            }
            mac-table-size 5000;
            mac-table-aging-time 600;
        }
    }
    CUSTOMER-A-SECONDARY {
        description "Customer A Secondary Services";
        vlan-id 1001;
        interface ge-0/0/1.20;      ## Site 2 VLAN 20
        interface ge-0/0/10.1001;   ## Core
    }
    CUSTOMER-B {
        description "Customer B All Services";
        vlan-id 2000;
        interface ge-0/0/2.99;      ## Translated
        interface ge-0/0/10.2000;   ## Core
```

```
    }
}

## Or Using VLANs (older style)
vlans {
    S-VLAN-1000 {
        description "Customer A Primary";
        vlan-id 1000;
        ## Interfaces added automatically via unit configuration
    }
    S-VLAN-1001 {
        description "Customer A Secondary";
        vlan-id 1001;
    }
    S-VLAN-2000 {
        description "Customer B";
        vlan-id 2000;
    }
}
```

Part 3: Verification & Troubleshooting (The What-If)

Essential Verification Commands

1. Verify VLAN Operations

```
user@pe> show interfaces ge-0/0/0.1000 | match vlan-map
  Input-vlan-map: push, Output-vlan-map: pop

user@pe> show interfaces ge-0/0/0.1000 extensive | match "VLAN-Tag"
    VLAN-Tag [ 0x8100.1000 ]
    Input VLAN-Tag: Outer [ 0x88a8.1000 ] Inner [ 0x8100.* ]
```

2. Check Bridge Domain Status

```
user@pe> show bridge domain

Routing instance        Bridge domain            VLAN ID     Interfaces
default-switch          CUSTOMER-A-PRIMARY       1000
                                                           ge-0/0/0.1000
                                                           ge-0/0/1.10
                                                           ge-0/0/10.1000
```

3. Monitor Q-in-Q Traffic

```
user@pe> monitor traffic interface ge-0/0/0.1000 layer2-headers detail
15:42:31.123456 In
  Ethernet II, Src: 00:11:22:33:44:55, Dst: 00:aa:bb:cc:dd:ee
  802.1Q, S-VID: 1000, C-VID: 10
  ↑ Two VLAN tags visible
```

4. Check MAC Learning with S-VLAN Context

```
user@pe> show bridge mac-table bridge-domain CUSTOMER-A-PRIMARY

MAC flags       : D - Dynamic, S - Static, L - Local
Routing instance: default-switch
Bridging domain: CUSTOMER-A-PRIMARY, VLAN: 1000

  MAC                 MAC      Logical          NH     RTR
  address             flags    interface        Index  ID
```

```
  00:11:22:33:44:55   D        ge-0/0/0.1000         <-- Customer MAC
  00:11:22:33:44:66   D        ge-0/0/10.1000        <-- Via core
```

Common Troubleshooting Scenarios

Scenario 1: Q-in-Q Tags Not Being Added

Symptom: Customer traffic arrives but doesn't get S-VLAN tag

Diagnostic Commands:

```
user@pe> show interfaces ge-0/0/0 | match encapsulation
  Encapsulation: Ethernet-Bridge    <-- Wrong! Should be extended-vlan-bridge

user@pe> monitor traffic interface ge-0/0/0 no-resolve layer2-headers
(Shows single VLAN tag when should show two)
```

Cause: Interface not configured for extended VLAN operations **Solution**:

```
set interfaces ge-0/0/0 encapsulation extended-vlan-bridge
set interfaces ge-0/0/0 flexible-vlan-tagging
delete interfaces ge-0/0/0 unit 0 family ethernet-switching
set interfaces ge-0/0/0 unit 1000 family bridge
commit
```

Scenario 2: VLAN Translation Not Working

Symptom: Customer VLAN 99 not being translated to S-VLAN 2000

Diagnostic Commands:

```
user@pe> show configuration interfaces ge-0/0/2.99 | display inheritance
vlan-id 99;
input-vlan-map {
    swap;           <-- Missing target VLAN ID!
}

user@pe> show interfaces ge-0/0/2.99 extensive | match swap
  Input-vlan-map: swap, Output-vlan-map: swap
  Swap-by-poppush-vlan-id: 0     <-- Should be 2000
```

Cause: VLAN map missing target VLAN ID **Solution**:

```
set interfaces ge-0/0/2 unit 99 input-vlan-map swap vlan-id 2000
set interfaces ge-0/0/2 unit 99 output-vlan-map swap vlan-id 99
commit
```

Scenario 3: MAC Limit Exceeded

Symptom: New devices cannot connect, existing work fine

Diagnostic Commands:

```
user@pe> show bridge mac-table bridge-domain CUSTOMER-A-PRIMARY summary
Total MAC count: 1000 (Limit: 1000)

user@pe> show log messages | match MAC_LIMIT
Nov 20 10:15:23 pe1 l2ald[1234]: MAC_LIMIT_EXCEEDED: ge-0/0/0.1000
```

Cause: Customer exceeding MAC address limit **Solution**:

```
# Increase limit
set bridge-domains CUSTOMER-A-PRIMARY bridge-options interface ge-0/0/0.1000 mac-limit 2000

# Or configure MAC limit with just logging
set bridge-domains CUSTOMER-A-PRIMARY bridge-options interface ge-0/0/0.1000 mac-limit packet-action none
set bridge-domains CUSTOMER-A-PRIMARY bridge-options interface ge-0/0/0.1000 mac-limit packet-action-for-limit log
```

Scenario 4: Inner VLAN Range Not Working

Symptom: Only some customer VLANs work through Q-in-Q

Diagnostic Commands:

```
user@pe> show configuration interfaces ge-0/0/3.0
vlan-id 1000;
family bridge {
    interface-mode trunk;
    ## Missing inner-vlan-id-list!
}

user@pe> show bridge domain interface ge-0/0/3.0 detail
  Number of VLANs: 1    <-- Should show range
```

Cause: Inner VLAN list not configured **Solution**:

```
set interfaces ge-0/0/3 unit 0 family bridge inner-vlan-id-list 1-4094
# Or specific range:
set interfaces ge-0/0/3 unit 0 family bridge inner-vlan-id-list [10 20 30-40]
commit
```

Scenario 5: Mixed Q-in-Q and Regular VLAN

Symptom: Some traffic double-tagged, some single-tagged incorrectly

Diagnostic Commands:

```
user@pe> monitor traffic interface ge-0/0/10 layer2-headers
# Shows mix of single and double-tagged frames

user@pe> show interfaces ge-0/0/* | match "unit|vlan-id" | except "Current"
ge-0/0/4:
  Logical interface ge-0/0/4.0
    VLAN-id: None     <-- Problem: inconsistent configuration
  Logical interface ge-0/0/4.100
    VLAN-id: 1000
```

Cause: Inconsistent unit configuration on interface **Solution**:

```
# Remove unit 0 if using flexible-vlan-tagging
delete interfaces ge-0/0/4 unit 0

# Ensure all units have proper VLAN configuration
set interfaces ge-0/0/4 unit 100 vlan-id 1000
set interfaces ge-0/0/4 unit 100 input-vlan-map push
commit
```

Pro Tips for Provider Bridging

1. **Plan S-VLAN Allocation**: Reserve ranges for different services
   - 1000-1999: Customer A
   - 2000-2999: Customer B

- 9000-9099: Management

2. **Monitor MAC Tables**: Provider edge can fill up quickly

```
set bridge-domains <name> bridge-options mac-statistics
```

3. **Use Firewall Filters**: Protect against customer misbehavior

```
set firewall family bridge filter LIMIT-BROADCAST term 1 from destination-mac-address ff:ff:ff:ff:ff:ff/48
set firewall family bridge filter LIMIT-BROADCAST term 1 then policer BROADCAST-POLICER
```

4. **Document VLAN Operations**: Track what happens at each interface
   - ge-0/0/0: Push S-VLAN 1000
   - ge-0/0/1: Selective push based on C-VLAN
   - ge-0/0/2: Translate VLAN 99 to 2000
5. **Test Both Directions**: VLAN operations are directional
   - Input map: What happens to incoming frames
   - Output map: What happens to outgoing frames

This module has equipped you with comprehensive knowledge of Provider Bridging - from understanding why double-tagging solves service provider challenges to implementing complex VLAN manipulations. These skills are essential for building scalable metro Ethernet services that maintain customer isolation while efficiently utilizing provider resources.

Module 6: Spanning-Tree Protocols

Part 1: The Conceptual Lecture (The Why)

The Fundamental Problem: Layer 2 Loops

Imagine an office network where someone accidentally connects two ports of the same switch with an Ethernet cable, or where multiple switches are interconnected with redundant links for reliability. What happens?

Broadcast Storm: When a PC sends a broadcast frame (like an ARP request):

1. Switch A receives it and floods to all ports
2. Switch B receives it and floods back to Switch A
3. Switch A floods it again...
4. The frame circulates forever!

```
The Loop Problem:
    PC1
     |
[Switch A]----[Switch B]
     |            |
     +------------+
      Loop Created!

PC1 sends broadcast →
Switch A floods to B →
Switch B floods back to A →
Infinite loop! Network dies in seconds.
```

Why This Kills Networks:

- Each broadcast gets multiplied exponentially
- Switches' CPU hit 100% processing the same frame
- MAC address tables constantly change (MAC flapping)
- Network becomes completely unusable in seconds

Enter the Spanning Tree Protocol (STP)

STP (IEEE 802.1D) prevents loops by:

1. Electing one switch as the "Root Bridge" (the boss)
2. Calculating the shortest path from each switch to the root
3. Blocking redundant paths
4. Creating a loop-free "tree" topology

```
Before STP:                    After STP:
    Root                          Root
   [SW1]                         [SW1]
   /    \                        /    \
[SW2]--[SW3]     →           [SW2]   [SW3]
   \    /                        \    /
   [SW4]                         [SW4]
                                   X
                            (Link blocked by STP)
```

How STP Works: The Election Process

Step 1: Root Bridge Election

Every switch thinks it should be root and sends "BPDU" (Bridge Protocol Data Unit) messages saying "I am the root!"

Bridge ID = Priority (2 bytes) + MAC Address (6 bytes)

- Default priority: 32768
- Lowest Bridge ID wins

```
Switch A: 32768.00:11:11:11:11:11
Switch B: 32768.00:22:22:22:22:22
Switch C: 16384.00:33:33:33:33:33  ← Wins! (Lower priority)
```

Step 2: Calculate Root Path Cost

Each link has a cost (based on speed):

- 10 Mbps = 100
- 100 Mbps = 19
- 1 Gbps = 4
- 10 Gbps = 2

Switches calculate total cost to reach root:

```
        Root (Cost 0)
         [C]
    4/        \4
   [A]         [B]
Cost: 4      Cost: 4
    \19      19/
      [D]---[E]
    Cost: 23  Cost: 23
```

Step 3: Select Port Roles

Each port gets a role:

- **Root Port**: Best path to root (one per non-root switch)
- **Designated Port**: Best path from segment to root (one per segment)
- **Blocked Port**: All others (prevents loops)

Step 4: Port States

Ports transition through states:

1. **Blocking** (20 sec): Listens to BPDUs only
2. **Listening** (15 sec): Sends/receives BPDUs
3. **Learning** (15 sec): Learns MAC addresses
4. **Forwarding**: Normal operation

Total convergence time: 50 seconds! (30-50 depending on topology)

BPDU: The STP Message

BPDUs are the "hello" messages of STP:

```
BPDU Structure:
+------------------+
| Protocol ID: 0   |
| Version: 0       |
| BPDU Type        |
| Flags            |
| Root Bridge ID   |
| Root Path Cost   |
| Bridge ID        |
| Port ID          |
| Message Age      |
| Max Age (20s)    |
| Hello Time (2s)  |
| Forward Delay    |
+------------------+
```

Evolution of STP

1. STP (802.1D) - 1990

- Original protocol
- 50-second convergence
- One tree for all VLANs

2. RSTP (802.1w) - 2001

- Rapid convergence (sub-second)
- New port roles (Alternate, Backup)
- Backward compatible

3. MSTP (802.1s) - 2002

- Multiple spanning trees
- Different topology per VLAN group
- Efficient for many VLANs

4. VSTP (Vendor-specific)

- Per-VLAN spanning tree
- Cisco originated, Juniper supports
- One tree per VLAN

Why Different STP Versions?

RSTP: Solves the 50-second problem

- Immediate transition to forwarding
- Proposal/Agreement mechanism
- Edge port concept (PortFast)

MSTP: Solves the scalability problem

- Groups VLANs into "instances"

- Maximum 64 instances (vs 4094 VLANs)
- Reduces CPU/memory usage

VSTP: Solves the suboptimal forwarding problem

- Each VLAN can have different topology
- Load balancing across redundant links
- More resource intensive

Part 2: The Junos CLI Masterclass (The How)

STP Configuration in Junos

Basic STP Configuration

```
# Enable STP globally
set protocols stp

# Configure STP on interfaces
set protocols stp interface ge-0/0/0
set protocols stp interface ge-0/0/1
set protocols stp interface ge-0/0/2

# Exclude edge ports (to PCs/servers)
set protocols stp interface ge-0/0/10 edge
```

Why: Edge ports transition immediately to forwarding (no loops possible with single host).

Configure Bridge Priority

```
# Set bridge priority (lower = more likely to be root)
set protocols stp bridge-priority 16384

# For specific VLAN in VSTP
set protocols vstp vlan 100 bridge-priority 16384
```

Priority values: Must be multiples of 4096 (0, 4096, 8192... 61440)

RSTP Configuration

```
# Enable RSTP (recommended over STP)
set protocols rstp

# Configure interfaces
set protocols rstp interface ge-0/0/0
set protocols rstp interface ge-0/0/1

# Set interface cost (optional)
set protocols rstp interface ge-0/0/0 cost 2000

# Configure edge ports
set protocols rstp interface ge-0/0/10 edge
set protocols rstp interface ge-0/0/11 edge

# No-root-port (for edge interfaces)
set protocols rstp interface ge-0/0/10 no-root-port
```

MSTP Configuration

Step 1: Configure MST Region

```
# Define MST region
set protocols mstp configuration-name REGION1
set protocols mstp revision-level 1

# Map VLANs to instances
set protocols mstp msti 1 vlan 10-50
```

```
set protocols mstp msti 2 vlan 51-100
set protocols mstp msti 3 vlan 101-200
```

Why: All switches in same region must have identical configuration-name, revision, and VLAN mappings.

Step 2: Configure MSTI Priorities

```
# Set priorities per instance
set protocols mstp msti 1 bridge-priority 16384
set protocols mstp msti 2 bridge-priority 32768
set protocols mstp msti 3 bridge-priority 49152
```

Step 3: Configure Interfaces

```
# Add interfaces to MSTP
set protocols mstp interface ge-0/0/0
set protocols mstp interface ge-0/0/1

# Set per-instance interface priorities
set protocols mstp interface ge-0/0/0 msti 1 priority 128
set protocols mstp interface ge-0/0/0 msti 2 priority 240
```

VSTP Configuration

```
# Enable VSTP
set protocols vstp

# Configure per VLAN
set protocols vstp vlan 10 bridge-priority 16384
set protocols vstp vlan 20 bridge-priority 32768

# Add interfaces
set protocols vstp interface ge-0/0/0
set protocols vstp interface ge-0/0/1

# Configure VLAN-specific interface settings
set protocols vstp vlan 10 interface ge-0/0/0 priority 128
set protocols vstp vlan 10 interface ge-0/0/0 cost 2000
```

Complete Reference Configuration

```
## RSTP Configuration (Recommended Default)
protocols {
    rstp {
        bridge-priority 16384;
        max-age 20;
        hello-time 2;
        forward-delay 15;

        interface ge-0/0/0 {
            cost 2000;
            priority 128;
        }
        interface ge-0/0/1 {
            cost 2000;
            priority 128;
        }
        interface ge-0/0/2 {
            cost 2000;
            priority 240;  ## Lower priority (backup)
        }

        ## Edge ports (to end devices)
        interface ge-0/0/10 {
            edge;
            no-root-port;
        }
```

```
        interface ge-0/0/11 {
            edge;
            no-root-port;
        }

        ## Disable on management
        interface fxp0 {
            disable;
        }
    }
}

## Alternative: MSTP Configuration
protocols {
    mstp {
        configuration-name REGION1;
        revision-level 1;

        ## Instance 0 (default for unmapped VLANs)
        bridge-priority 32768;

        ## Instance 1: VLANs 10-50
        msti 1 {
            bridge-priority 16384;
            vlan 10-50;
        }

        ## Instance 2: VLANs 51-100
        msti 2 {
            bridge-priority 32768;
            vlan 51-100;
        }

        interface ge-0/0/0 {
            priority 128;

            msti 1 {
                priority 128;
                cost 2000;
            }
            msti 2 {
                priority 240;
                cost 20000;
            }
        }

        interface ge-0/0/1 {
            priority 128;
        }

        interface ge-0/0/10 {
            edge;
        }
    }
}

## Alternative: VSTP Configuration
protocols {
    vstp {
        interface all;  ## Enable on all interfaces

        ## Configure specific VLANs
        vlan 10 {
            bridge-priority 16384;
            max-age 20;
            hello-time 2;
            forward-delay 15;

            interface ge-0/0/0 {
                priority 128;
                cost 2000;
```

```
            }
            interface ge-0/0/1 {
                priority 240;
                cost 20000;
            }
        }

        vlan 20 {
            bridge-priority 32768;

            interface ge-0/0/0 {
                priority 240;  ## Backup for VLAN 20
            }
            interface ge-0/0/1 {
                priority 128;  ## Primary for VLAN 20
            }
        }

        ## Disable on specific interfaces
        interface ge-0/0/10 {
            disable;
        }
    }
}
```

Part 3: Verification & Troubleshooting (The What-If)

Essential Verification Commands

1. Check STP Status

```
user@switch> show spanning-tree bridge
STP bridge parameters
Routing instance name               : GLOBAL
Context ID                          : 0
Enabled protocol                    : RSTP
  Root ID                           : 16384.00:11:11:11:11:11
  Root cost                         : 2000
  Root port                         : ge-0/0/0
  Hello time                        : 2 seconds
  Maximum age                       : 20 seconds
  Forward delay                     : 15 seconds
  Bridge ID                         : 32768.00:22:22:22:22:22
  Local root cost                   : 0
  Number of topology changes        : 17
  Time since last topology change   : 3421 seconds
```

What to look for:

- Which protocol is running (STP/RSTP/MSTP/VSTP)
- Root bridge ID (is it expected?)
- Root port (path to root)
- Bridge ID (local switch)

2. Check Interface Status

```
user@switch> show spanning-tree interface

Spanning tree interface parameters for instance 0

Interface    Port ID    Designated         Designated         Port    State  Role
                        port ID            bridge ID          Cost
ge-0/0/0     128:513    128:513    16384.001111111111         2000  FWD    ROOT
```

```
ge-0/0/1     128:514    128:514    32768.002222222222         2000  FWD    DESG
ge-0/0/2     128:515    128:515    32768.002222222222         2000  BLK    ALT
```

Port States:

- FWD (Forwarding): Passing traffic
- BLK (Blocking): Preventing loops
- LRN (Learning): Building MAC table
- LIS (Listening): Exchanging BPDUs

Port Roles:

- ROOT: Best path to root bridge
- DESG: Designated port for segment
- ALT: Alternate path (RSTP)
- BACKUP: Backup port (RSTP)

3. Check MSTP Details

```
user@switch> show spanning-tree mstp configuration
MSTP configuration information
Context identifier   : 0
Region name          : REGION1
Revision             : 1
Configuration digest : 0x7c23a5b8f91d4e6c8a9b2f3d5e1a7b9c

Instance  Vlan
0         1-9,51
1         10-50
2         51-100
```

Common Troubleshooting Scenarios

Scenario 1: Loop Despite STP Running

Symptom: Broadcast storm occurring, network unusable

Diagnostic Commands:

```
user@switch> show spanning-tree interface | match BLK
(No output - no ports blocking!)

user@switch> show log messages | match BPDU
Nov 20 14:22:13 Loop detected on ge-0/0/3

user@switch> show interfaces ge-0/0/3 | match STP
  STP: Disabled     <-- Problem found!
```

Cause: Interface not participating in STP **Solution**:

```
set protocols rstp interface ge-0/0/3
commit

# Verify
show spanning-tree interface ge-0/0/3
```

Scenario 2: Root Bridge Flapping

Symptom: Network unstable, frequent topology changes

Diagnostic Commands:

```
user@switch> show spanning-tree bridge | match "changes|change"
  Number of topology changes        : 847    <-- Very high!
  Time since last topology change   : 12 seconds

user@switch> monitor start messages | match STP
Nov 20 14:30:15 New root: 32768.00:33:33:33:33:33
Nov 20 14:30:27 New root: 32768.00:44:44:44:44:44
```

Cause: Multiple switches with same priority competing **Solution**:

```
# Set deterministic root bridge
set protocols rstp bridge-priority 8192
commit

# On backup root bridge
set protocols rstp bridge-priority 16384
```

Scenario 3: Suboptimal Forwarding Path

Symptom: Traffic taking longer path despite shorter one available

Diagnostic Commands:

```
user@switch> show spanning-tree interface detail ge-0/0/0
[...]
Port cost                           : 20000    <-- Very high!
Port priority                       : 128

user@switch> show configuration protocols rstp interface ge-0/0/0
cost 20000;    <-- Manually set high cost
```

Cause: Interface cost manually set too high **Solution**:

```
# Use automatic cost based on speed
delete protocols rstp interface ge-0/0/0 cost

# Or set appropriate cost
set protocols rstp interface ge-0/0/0 cost 2000
commit
```

Scenario 4: MSTP Region Mismatch

Symptom: MSTP not working between switches

Diagnostic Commands:

```
user@switch1> show spanning-tree mstp configuration
Region name          : REGION1
Revision             : 1

user@switch2> show spanning-tree mstp configuration
Region name          : REGION-1    <-- Different name!
Revision             : 1
```

Cause: MSTP region parameters don't match exactly **Solution**:

```
# Make region configuration identical on all switches
set protocols mstp configuration-name REGION1
set protocols mstp revision-level 1
# Ensure VLAN-to-instance mappings also match
commit
```

Scenario 5: Edge Port Causing Issues

Symptom: PC connections take 30+ seconds to work

Diagnostic Commands:

```
user@switch> show spanning-tree interface ge-0/0/10
Interface    Port ID    State  Role
ge-0/0/10    128:521    LRN    DESG    <-- Still learning!

user@switch> show configuration protocols rstp interface ge-0/0/10
## No edge configuration
```

Cause: Edge port not configured, going through STP states **Solution**:

```
set protocols rstp interface ge-0/0/10 edge
set protocols rstp interface ge-0/0/10 no-root-port
commit

# Verify instant forwarding
show spanning-tree interface ge-0/0/10
State: FWD (should be immediate)
```

Pro Tips for STP Management

1. **Always Use RSTP or MSTP**: Original STP is too slow

   ```
   delete protocols stp
   set protocols rstp
   ```

2. **Design Your Root Bridge**: Don't let STP choose randomly
   - Primary root: Priority 8192
   - Backup root: Priority 16384
   - Others: Leave at 32768
3. **Use Edge Ports**: Speeds up host connections

   ```
   set protocols rstp interface ge-0/0/[10-20] edge
   ```

4. **Monitor Topology Changes**: Frequent changes indicate problems

   ```
   show spanning-tree statistics-information
   ```

5. **Document Expected Topology**: Draw which ports should block
   - Helps troubleshooting
   - Validates configuration
   - Identifies unauthorized changes

This module has provided comprehensive knowledge of spanning-tree protocols - from understanding why loops kill networks to implementing modern rapid protocols that converge in under a second. These skills are critical for building resilient Layer 2 networks that automatically handle failures while preventing the catastrophic loops that would otherwise destroy network functionality.

Module 7: Configuring Spanning-Tree

Part 1: The Conceptual Lecture (The Why)

Beyond Basic STP: Protection Mechanisms

While Module 6 covered how STP prevents loops, Module 7 focuses on protecting your spanning-tree topology from both accidental misconfigurations and deliberate attacks. Think of these features as security guards for your network topology.

The New Problems We Face

Problem 1: Rogue Switches

Someone plugs an unauthorized switch into your network. This switch might:

- Claim to be the root bridge (low priority)
- Cause your carefully designed topology to reconverge
- Create suboptimal paths or even loops

```
Original Topology:              After Rogue Switch:
    [Root]                          [Rogue Root]
   /      \                              |
[SW1]    [SW2]        →           [Original Root]
  |        |                          /        \
[SW3]    [SW4]                   [SW1]      [SW2]
                                    |          |
                                 [SW3]        [SW4]
```

Problem 2: Unidirectional Links

A fiber cable fails in one direction only. Switch A can hear Switch B, but B cannot hear A:

- A thinks the link is up
- B stops receiving BPDUs from A
- B unblocks alternate ports
- Loop created!

Problem 3: BPDU Loss

Configuration errors or software bugs cause BPDU loss:

- Edge port accidentally receives BPDUs
- Software filters BPDUs incorrectly
- CPU too busy to process BPDUs

BPDU Protection

BPDU Protection prevents edge ports from processing received BPDUs. It's like a "No STP Allowed" sign on ports connected to end devices.

Use Cases:

- Ports connected to PCs/Servers
- Ports where customers connect
- Any port that should NEVER see a switch

How it Works:

1. Configure port as edge with BPDU protection
2. If BPDU received → Port is disabled
3. Administrator must manually re-enable

```
Normal Edge Port:               With BPDU Protection:
PC → Edge Port → Switch         PC → Edge Port → Switch
                                         ↓
                               If BPDU received:
                               Port DISABLED!
```

Loop Protection

Loop Protection prevents ports from transitioning from blocking to forwarding when BPDUs stop arriving. It's insurance against unidirectional link failures.

The Unidirectional Link Problem:

```
Normal Operation:           Unidirectional Failure:
[A] ←BPDU→ [B]             [A] ←BPDU— [B]
    Link OK                     TX OK, RX Failed

                           B stops receiving BPDUs
                           B thinks A is dead
                           B unblocks alternate port
                           LOOP CREATED!
```

How Loop Protection Works:

- If designated port stops receiving BPDUs
- Instead of transitioning blocked ports to forwarding
- Transitions them to "loop-inconsistent" state
- Prevents loops from unidirectional failures

Root Protection

Root Protection prevents ports from becoming root ports. It enforces your topology design by saying "The root bridge is NEVER in this direction."

Use Cases:

- Customer-facing ports
- Access layer ports
- Any port facing "untrusted" networks

How it Works:

1. Configure root protection on port
2. If superior BPDU received (better root)
3. Port placed in "root-prevented" state
4. Port blocks but continues sending BPDUs

```
Without Root Protection:        With Root Protection:
Customer → Superior BPDU        Customer → Superior BPDU
    ↓                               ↓
Switch accepts new root         Switch blocks port
Topology reconverges            Topology protected
```

Advanced STP Timers and Optimization

Understanding and tuning STP timers can significantly improve network performance:

Key Timers:

1. **Hello Time** (default 2s): How often BPDUs are sent
2. **Max Age** (default 20s): How long to store BPDU info
3. **Forward Delay** (default 15s): Time in listening/learning states

Timer Relationship: Max Age must be ≥ 2 × (Hello Time + 1)

Diameter Considerations:

Network diameter = maximum switches between any two endpoints

```
Recommended timers by diameter:
Diameter 2-3: Hello=2s, MaxAge=20s, FwdDelay=15s
Diameter 4-5: Hello=2s, MaxAge=25s, FwdDelay=18s
Diameter 6-7: Hello=2s, MaxAge=30s, FwdDelay=21s
```

Per-VLAN vs Global Protection

Different STP variants handle protection differently:

RSTP/STP: Global protection

- One STP instance
- Protection applies to all VLANs

MSTP: Per-instance protection

- Protection per MST instance
- Different policies for different VLAN groups

VSTP: Per-VLAN protection

- Each VLAN has independent STP
- Maximum granularity but resource intensive

Part 2: The Junos CLI Masterclass (The How)

Configuring BPDU Protection

Global BPDU Protection

```
# Enable BPDU protection on all edge ports
set protocols rstp bpdu-block-on-edge

# Or for specific protocol
set protocols stp bpdu-block-on-edge
set protocols mstp bpdu-block-on-edge
set protocols vstp bpdu-block-on-edge
```

Per-Interface BPDU Protection

```
# Configure edge port with BPDU protection
set protocols rstp interface ge-0/0/10 edge
set protocols rstp interface ge-0/0/10 bpdu-block

# Disable specific interfaces if needed
set protocols rstp interface ge-0/0/10 disable-bpdu-block-on-edge
```

Recovery from BPDU Protection

```
# Configure automatic recovery
set protocols rstp bpdu-block disable-timeout 300

# Or manual recovery
clear ethernet-switching bpdu-error interface ge-0/0/10
```

Configuring Loop Protection

Global Loop Protection

```
# Enable loop protection globally
set protocols rstp loop-protection

# For other STP variants
set protocols mstp loop-protection
set protocols vstp loop-protection
```

Per-Interface Loop Protection

```
# Enable on specific interfaces
set protocols rstp interface ge-0/0/0 loop-protection
set protocols rstp interface ge-0/0/1 loop-protection

# Disable on specific interface if globally enabled
set protocols rstp interface ge-0/0/2 no-loop-protection
```

Configuring Root Protection

```
# Configure root protection on access ports
set protocols rstp interface ge-0/0/0 no-root-port
set protocols rstp interface ge-0/0/1 no-root-port

# For MSTP (per instance)
set protocols mstp interface ge-0/0/0 msti 1 no-root-port
set protocols mstp interface ge-0/0/0 msti 2 no-root-port

# For VSTP (per VLAN)
set protocols vstp vlan 10 interface ge-0/0/0 no-root-port
set protocols vstp vlan 20 interface ge-0/0/0 no-root-port
```

Advanced Timer Configuration

```
# Modify global timers
set protocols rstp max-age 25
set protocols rstp hello-time 2
set protocols rstp forward-delay 20

# Per-VLAN timers (VSTP)
set protocols vstp vlan 10 max-age 25
set protocols vstp vlan 10 hello-time 2
set protocols vstp vlan 10 forward-delay 20

# Verify timer relationships
show protocols rstp | display detail | match "time|age|delay"
```

Complete Protection Configuration

```
## RSTP with Full Protection Suite
protocols {
    rstp {
        ## Core settings
        bridge-priority 16384;
        max-age 20;
        hello-time 2;
        forward-delay 15;

        ## Global protection
        bpdu-block-on-edge;
        loop-protection;

        ## Automatic recovery
        bpdu-block {
            disable-timeout 600;  ## 10 minutes
        }

        ## Uplink interfaces (to core)
        interface ge-0/0/0 {
            priority 128;
            cost 2000;
            loop-protection;      ## Protect against uni-directional
        }
        interface ge-0/0/1 {
            priority 128;
            cost 2000;
            loop-protection;
        }

        ## Access interfaces (to customers/users)
        interface ge-0/0/10 {
            edge;
            no-root-port;         ## Root protection
            bpdu-block;          ## BPDU protection
        }
        interface ge-0/0/11 {
            edge;
```

```
            no-root-port;
            bpdu-block;
        }

        ## Interface range configuration
        interface-range ACCESS-PORTS {
            member-range ge-0/0/12 to ge-0/0/23;
            edge;
            no-root-port;
            bpdu-block;
        }
    }
}

## Alternative: MSTP with Protection
protocols {
    mstp {
        configuration-name PROTECTED-REGION;
        revision-level 2;

        ## Global protections
        bpdu-block-on-edge;
        loop-protection;
        bpdu-block {
            disable-timeout 300;
        }

        ## Instance configuration
        msti 1 {
            bridge-priority 16384;
            vlan [ 10-50 ];
        }
        msti 2 {
            bridge-priority 32768;
            vlan [ 51-100 ];
        }

        ## Protected interfaces
        interface ge-0/0/0 {
            msti 1 {
                priority 128;
                cost 2000;
            }
            msti 2 {
                priority 240;
                cost 20000;
            }
            loop-protection;
        }

        ## Access ports with protection
        interface-range CUSTOMER-FACING {
            member-range ge-0/0/10 to ge-0/0/20;
            edge;
            bpdu-block;
            msti 1 {
                no-root-port;
            }
            msti 2 {
                no-root-port;
            }
        }
    }
}

## VSTP with Per-VLAN Protection
protocols {
    vstp {
        ## Global settings
        bpdu-block-on-edge;
        bpdu-block {
```

```
            disable-timeout 900;  ## 15 minutes
        }

        ## VLAN 10 configuration
        vlan 10 {
            bridge-priority 8192;
            loop-protection;

            interface ge-0/0/0 {
                priority 128;
            }
            interface ge-0/0/10 {
                edge;
                no-root-port;
                bpdu-block;
            }
        }

        ## VLAN 20 configuration
        vlan 20 {
            bridge-priority 16384;
            loop-protection;

            interface ge-0/0/1 {
                priority 128;
            }
            interface ge-0/0/10 {
                edge;
                no-root-port;
                bpdu-block;
            }
        }

        ## Apply to all other interfaces
        interface all {
            edge;
            bpdu-block;
        }
    }
}
```

Part 3: Verification & Troubleshooting (The What-If)

Essential Verification Commands

1. Check Protection Status

```
user@switch> show spanning-tree interface detail
[...]
Interface name : ge-0/0/10
[...]
  Edge port: Yes
  BPDU block: Enabled
  Root protection: Enabled
  Loop protection: Disabled
  State: Forwarding
```

2. View BPDU Errors

```
user@switch> show ethernet-switching bpdu-error
Interface         State      BPDU-block     Time
ge-0/0/10.0       Blocked    Enabled        2023-11-20 14:30:15 UTC
ge-0/0/11.0       Blocked    Enabled        2023-11-20 15:45:22 UTC
```

3. Check Loop Protection Status

```
user@switch> show spanning-tree interface | match loop-inconsistent
ge-0/0/2.0       128:515    LOOP-INCONSISTENT
```

4. View Root Protection Events

```
user@switch> show log messages | match ROOT_PREVENT
Nov 20 16:30:45 ROOT_PREVENTED_ON_INTERFACE: ge-0/0/10.0: Root prevented
```

Common Troubleshooting Scenarios

Scenario 1: Edge Port Disabled by BPDU Protection

Symptom: User port suddenly stops working

Diagnostic Commands:

```
user@switch> show ethernet-switching interface ge-0/0/10
State: Blocked
Blocked by: STP BPDU error

user@switch> show log messages | match "ge-0/0/10|BPDU"
Nov 20 10:15:33 l2ald[1234]: BPDU_BLOCK: ge-0/0/10.0: BPDU received on edge port
```

Cause: Someone connected a switch to edge port **Solution**:

```
# First, remove the switch!
# Then clear the error:
clear ethernet-switching bpdu-error interface ge-0/0/10

# Or wait for auto-recovery if configured
show protocols rstp bpdu-block
  Disable timeout: 600 seconds
```

Scenario 2: Loop Protection Blocking Port

Symptom: Redundant link not working after primary fails

Diagnostic Commands:

```
user@switch> show spanning-tree interface detail ge-0/0/2
State: LOOP-INCONSISTENT
Loop protection: Enabled

user@switch> show log messages | match LOOP_PROTECT
Nov 20 11:20:15 LOOP_PROTECTION_ACTIVATED: ge-0/0/2.0: No BPDUs received
```

Cause: Unidirectional link failure or BPDU loss **Solution**:

```
# Check physical connectivity
show interfaces ge-0/0/2 extensive | match "error|drop|FCS"

# Temporarily disable loop protection to test
set protocols rstp interface ge-0/0/2 no-loop-protection
commit

# If loop occurs, re-enable immediately!
rollback 1
```

Scenario 3: Root Protection Preventing Convergence

Symptom: Network won't reconverge after primary root fails

Diagnostic Commands:

```
user@switch> show spanning-tree interface
Interface    State          Role
ge-0/0/0     ROOT-PREVENTED ALT
ge-0/0/1     DOWN           DIS

user@switch> show spanning-tree bridge
Root ID    : 32768.00:11:11:11:11:11  (This switch!)
Root port  : None
```

Cause: Root protection on all potential root ports **Solution**:

```
# Remove root protection from uplink ports
delete protocols rstp interface ge-0/0/0 no-root-port
commit

# Better: Design proper root protection
# Only on access/customer-facing ports
```

Scenario 4: Timer Mismatch Issues

Symptom: Frequent topology changes, slow convergence

Diagnostic Commands:

```
user@switch1> show spanning-tree bridge | match time
  Hello time                        : 2 seconds
  Maximum age                       : 20 seconds
  Forward delay                     : 15 seconds

user@switch2> show spanning-tree bridge | match time
  Hello time                        : 1 seconds    <-- Different!
  Maximum age                       : 10 seconds   <-- Different!
  Forward delay                     : 15 seconds
```

Cause: Inconsistent timer configuration **Solution**:

```
# Standardize timers across all switches
set protocols rstp hello-time 2
set protocols rstp max-age 20
set protocols rstp forward-delay 15
commit

# Verify formula: MaxAge >= 2(Hello + 1)
# 20 >= 2(2 + 1) = 6 ✓
```

Scenario 5: BPDU Protection Too Aggressive

Symptom: Legitimate redundant links being blocked

Diagnostic Commands:

```
user@switch> show configuration protocols rstp
bpdu-block-on-edge;
interface all {      <-- Problem: Applied to ALL interfaces!
    edge;
    bpdu-block;
}
```

Cause: BPDU protection on trunk ports **Solution**:

```
# Remove global edge configuration
delete protocols rstp interface all

# Configure edge only on access ports
set protocols rstp interface-range ACCESS member-range ge-0/0/10 to ge-0/0/48
set protocols rstp interface-range ACCESS edge
set protocols rstp interface-range ACCESS bpdu-block

# Ensure trunk ports are not edge
set protocols rstp interface ge-0/0/0 no-edge
set protocols rstp interface ge-0/0/1 no-edge
commit
```

Pro Tips for STP Protection

1. **Layer Your Protection**:

   ```
   Core Ports:    Loop Protection only
   Uplinks:       Loop Protection + Aggressive timers
   Access Ports:  BPDU Protection + Root Protection + Edge
   ```

2. **Monitor Protection Events**:

   ```
   set system syslog file stp-protection any any
   set system syslog file stp-protection match "BPDU|ROOT|LOOP"
   ```

3. **Test Protection Mechanisms**:
   - Connect a switch to edge port → Should disable
   - Unplug receive fiber → Should go loop-inconsistent
   - Configure low priority on edge → Should go root-prevented
4. **Document Expected Behavior**:

   ```
   ## Port Protection Map ##
   ge-0/0/0-1:   Uplinks - Loop Protection
   ge-0/0/2-9:   Server - Edge + BPDU Block
   ge-0/0/10-48: Users - Edge + BPDU + Root Block
   ```

5. **Use Automatic Recovery Carefully**:
   - Too short: Doesn't fix underlying problem
   - Too long: Extended outage
   - Recommended: 300-900 seconds with alerting

This module has equipped you with advanced STP protection mechanisms that prevent both accidental and malicious topology disruptions. These features transform STP from a basic loop prevention protocol into a robust, secure foundation for your Layer 2 network. Combined with proper design and monitoring, these protections ensure your spanning-tree topology remains stable and predictable even in the face of misconfigurations or attacks.

Module 8: Ethernet OAM

Part 1: The Conceptual Lecture (The Why)

The Fundamental Problem

Traditional Ethernet was designed for local networks where you could physically trace cables and see link lights. But in modern carrier networks, Ethernet spans cities, countries, even continents. When a customer says "my connection is slow," how do you troubleshoot a problem that might be anywhere across thousands of kilometers of fiber and hundreds of devices?

The Challenges:

1. **No visibility**: Can't see remote link status

2. **No diagnostics**: Can't ping at Layer 2
3. **No performance data**: Can't measure loss or delay
4. **No fault isolation**: Can't determine where problem occurs

The Solution: Ethernet OAM (Operation, Administration, and Maintenance) provides tools to monitor, troubleshoot, and measure Ethernet services just like we do for IP networks.

Two Flavors of Ethernet OAM

1. Link Fault Management (LFM) - IEEE 802.3ah

- Works on single link (point-to-point)
- Monitors physical layer health
- Like checking your pulse

2. Connectivity Fault Management (CFM) - IEEE 802.1ag

- Works end-to-end across network
- Monitors service health
- Like checking entire circulatory system

```
LFM vs CFM Scope:
[CE1]--LFM--[PE1]====CFM====[PE2]--LFM--[CE2]
      Link1       Network         Link2

LFM: Monitors each individual link
CFM: Monitors complete path CE1 to CE2
```

Link Fault Management (LFM) Deep Dive

LFM runs directly on Ethernet links using special "slow protocol" frames (EtherType 0x8809).

LFM Components:

1. **OAM PDUs (Protocol Data Units)**
   - Information OAMPDU: Heartbeat messages
   - Event Notification: Report problems
   - Variable Request/Response: Read remote MIB
   - Loopback Control: Test connectivity
2. **Discovery Process**

   ```
   Step 1: Send Information OAMPDU
   Step 2: Receive remote Information OAMPDU
   Step 3: Negotiate capabilities
   Step 4: Enter "Active" mode
   Step 5: Exchange periodic heartbeats
   ```

3. **Fault Detection**
   - Link fault: Physical layer problems
   - Dying gasp: Remote device powering down
   - Critical event: Unspecified critical failure

LFM State Machine:

```
FAULT ←→ ACTIVE SEND LOCAL → SEND LOCAL REMOTE → SEND ANY
  ↓         ↓                      ↓                   ↓
Device    Can send             Can send            Full OAM
failed    local info           local+remote        active
```

Connectivity Fault Management (CFM) Deep Dive

CFM creates a hierarchical monitoring system with different "maintenance domains" at different levels of the network.

CFM Hierarchy:

```
Level 7: Customer Domain       [CE1]--------[CE2]
                                  ↑            ↑
Level 5: Provider Domain    [PE1]===========[PE2]
                               ↑                ↑
Level 3: Operator Domain   [P1]---[P2]---[P3]---[P4]
```

CFM Components:

1. **Maintenance Domain (MD)**
   - Defines scope of management
   - Has a level (0-7)
   - Contains Maintenance Associations
2. **Maintenance Association (MA)**
   - Service instance within MD
   - Usually maps to a VLAN/Service
   - Contains Maintenance End Points
3. **Maintenance End Points (MEP)**
   - Active monitoring points
   - Send/receive CFM messages
   - Can be Up or Down facing
4. **Maintenance Intermediate Points (MIP)**
   - Passive monitoring points
   - Respond to CFM messages
   - Don't initiate messages

```
MEP and MIP Placement:
[MEP]---[MIP]---[MIP]---[MEP]
 ↓                        ↓
Initiates              Initiates
CFM                    CFM


        ↑       ↑
     Responds  Responds
     Only      Only
```

CFM Protocol Messages:

1. **Continuity Check (CCM)**
   - Heartbeat messages
   - Sent continuously (3.3ms to 10 min intervals)
   - Detect connectivity failures
2. **Loopback (LBM/LBR)**
   - Like Layer 2 ping
   - Tests bidirectional connectivity
   - On-demand troubleshooting
3. **Link Trace (LTM/LTR)**
   - Like Layer 2 traceroute
   - Discovers path between MEPs
   - Shows all MIPs in path
4. **Delay Measurement (DMM/DMR)**
   - Measures round-trip delay
   - Timestamps in messages
   - Performance monitoring
5. **Loss Measurement (LMM/LMR)**
   - Counts transmitted/received frames
   - Calculates loss ratio

- SLA verification

Why Different Levels?

Each level monitors different segments:

```
Customer View (Level 7):
"Is my service working end-to-end?"
[CE1]====================[CE2]

Provider View (Level 5):
"Is service working across my network?"
     [PE1]============[PE2]

Operator View (Level 3):
"Are my transport links working?"
        [P1]--[P2]--[P3]
```

This hierarchy allows fault isolation:

- Level 7 fails, Level 5 works = Customer equipment problem
- Level 5 fails, Level 3 works = Provider service problem
- Level 3 fails = Transport network problem

Part 2: The Junos CLI Masterclass (The How)

Configuring Link Fault Management (LFM)

Basic LFM Configuration

```
# Enable LFM on interface
set protocols oam ethernet link-fault-management interface ge-0/0/0

# Configure LFM parameters
set protocols oam ethernet link-fault-management interface ge-0/0/0 link-discovery active
set protocols oam ethernet link-fault-management interface ge-0/0/0 pdu-interval 1000
set protocols oam ethernet link-fault-management interface ge-0/0/0 remote-loopback
```

Parameters explained:

- `link-discovery active` : Actively initiate OAM discovery
- `pdu-interval` : Milliseconds between PDUs (100-1000ms)
- `remote-loopback` : Allow remote to put us in loopback

Advanced LFM Configuration

```
# Configure event thresholds
set protocols oam ethernet link-fault-management interface ge-0/0/0 event-thresholds {
    symbol-period 10 per 1000;      # 10 errors per 1000 symbols
    frame-error 10 per 1000;        # 10 errors per 1000 frames
    frame-period 10 per 1000;       # 10 errors per 1000 frames
    frame-period-summary 10 per 1000;
}

# Configure actions on events
set protocols oam ethernet link-fault-management action-profile SHUTDOWN {
    event link-adjacency-loss;
    event link-event-rate {
        symbol-period 20;           # Errors per minute
        frame-error 15;
    }
    action syslog;
    action link-down;
}
```

```
# Apply action profile
set protocols oam ethernet link-fault-management interface ge-0/0/0 action-profile SHUTDOWN
```

Configuring Connectivity Fault Management (CFM)

Step 1: Define Maintenance Domain

```
# Create provider-level domain
set protocols oam ethernet connectivity-fault-management maintenance-domain PROVIDER {
    level 5;
    name-format character-string;
    maintenance-association CUSTOMER-A {
        continuity-check {
            interval 1s;
            hold-interval 3.5;
        }
        mep 100 {
            interface ge-0/0/0.100;
            direction down;
            auto-discovery;
        }
        mep 200 {
            interface ge-0/0/1.100;
            direction down;
            remote-mep 201;
        }
    }
}
```

Key concepts:

- `level 5`: Provider domain level
- `interval 1s`: CCM sent every second
- `hold-interval 3.5`: 3.5 × interval before declaring MEP down
- `direction down`: MEP faces customer
- `remote-mep`: Expected remote MEP IDs

Step 2: Configure Multiple Levels

```
# Customer level domain
set protocols oam ethernet connectivity-fault-management maintenance-domain CUSTOMER {
    level 7;
    name-format character-string;
    maintenance-association CE-TO-CE {
        continuity-check {
            interval 10s;
        }
        mep 1001 {
            interface ge-0/0/0.100;
            direction up;
        }
    }
}

# Operator level domain
set protocols oam ethernet connectivity-fault-management maintenance-domain OPERATOR {
    level 3;
    name-format character-string;
    maintenance-association TRANSPORT {
        continuity-check {
            interval 100ms;
        }
        mep 10 {
            interface ge-0/0/10;
            direction down;
        }
    }
}
```

Step 3: Configure MIPs

```
# Explicit MIP configuration
set protocols oam ethernet connectivity-fault-management maintenance-domain PROVIDER {
    maintenance-association CUSTOMER-A {
        mip-half-function {
            interface ge-0/0/2.100;
        }
    }
}

# Or automatic MIP creation
set protocols oam ethernet connectivity-fault-management maintenance-domain PROVIDER {
    mip-auto-discovery;
}
```

Complete OAM Configuration Example

```
## Link Fault Management
protocols {
    oam {
        ethernet {
            link-fault-management {
                ## Action profiles
                action-profile LINK-PROTECTION {
                    event {
                        link-adjacency-loss;
                        critical-event;
                        dying-gasp;
                        link-event-rate {
                            symbol-period 50;
                            frame-error 40;
                        }
                    }
                    action {
                        syslog;
                        ## Optionally bring link down
                        # link-down;
                    }
                }

                ## Interface configuration
                interface ge-0/0/0 {
                    link-discovery active;
                    pdu-interval 500;
                    remote-loopback;
                    action-profile LINK-PROTECTION;
                    event-thresholds {
                        symbol-period 10 per 1000;
                        frame-error 10 per 1000;
                    }
                }
                interface ge-0/0/1 {
                    link-discovery passive;
                    pdu-interval 1000;
                }
            }

            ## Connectivity Fault Management
            connectivity-fault-management {
                ## Performance monitoring
                performance-monitoring {
                    sla-iterator-profile DELAY-LOSS {
                        measurement-type two-way-delay;
                        cycle-time 1000;        ## 1 second
                        iteration-period 3000;   ## 3 seconds
                    }
                }

                ## Operator Domain (Level 3)
```

```
maintenance-domain OPERATOR {
    level 3;
    name-format character-string;

    maintenance-association CORE-LINKS {
        continuity-check {
            interval 100ms;
            hold-interval 3.5;
            loss-threshold 3;
        }
        mep 301 {
            interface ge-0/0/10;
            direction down;
            remote-mep 302;
        }
    }
}

## Provider Domain (Level 5)
maintenance-domain PROVIDER {
    level 5;
    name-format character-string;
    mip-auto-discovery;

    maintenance-association CUSTOMER-A-VLAN100 {
        vlan-id 100;
        continuity-check {
            interval 1s;
            hold-interval 3.5;
        }
        mep 501 {
            interface ge-0/0/0.100;
            direction down;
            auto-discovery;
            ## Performance monitoring
            sla-iterator-profile DELAY-LOSS;
        }
        mep 502 {
            interface ge-0/0/1.100;
            direction down;
            remote-mep 501;
        }
    }

    maintenance-association CUSTOMER-B-VLAN200 {
        vlan-id 200;
        continuity-check {
            interval 10s;
        }
        mep 503 {
            interface ge-0/0/2.200;
            direction down;
        }
    }
}

## Customer Domain (Level 7)
maintenance-domain CUSTOMER {
    level 7;
    name-format character-string;

    maintenance-association END-TO-END {
        continuity-check {
            interval 1m;
        }
        mep 701 {
            interface ge-0/0/0.100;
            direction up;
        }
    }
}
```

```
                ## Linktrace database
                linktrace {
                    age 65535;
                    size 2048;
                }
            }
        }
    }
}
```

Part 3: Verification & Troubleshooting (The What-If)

Essential Verification Commands

1. Check LFM Status

```
user@router> show oam ethernet link-fault-management interface ge-0/0/0
Interface: ge-0/0/0
  Status: Active, Discovery State: Send Any
  Peer address: 00:11:22:33:44:55
  Flags: Remote-Stable Remote-State-Valid Local-Stable 0x50

  Local:
    State: Active
    MUX: Forwarding, PAR: Forwarding

  Remote:
    State: Active
    MUX: Forwarding, PAR: Forwarding
    Vendor OUI: 00:90:69 (Juniper Networks)

  Statistics:
    Information OAMPDU Tx: 45123
    Information OAMPDU Rx: 45098
    Event Notification Tx: 3
    Event Notification Rx: 0
```

What to look for:

- Discovery State: "Send Any" = fully operational
- Both MUX and PAR: "Forwarding" = normal operation
- Statistics show bidirectional communication

2. Check CFM MEP Status

```
user@router> show oam ethernet connectivity-fault-management mep-database
Maintenance domain: PROVIDER, Level: 5
  Maintenance association: CUSTOMER-A-VLAN100
    Local MEP: 501, Direction: Down
      Interface: ge-0/0/0.100
      Continuity check status: Active
      Remote MEP: 502
        Status: OK
        MAC address: 00:22:33:44:55:66
        Last CCM received: 00:00:01 ago
```

3. CFM Loopback Test

```
user@router> ping ethernet maintenance-domain PROVIDER maintenance-association CUSTOMER-A-VLAN100 mep 501 remote-mep 502
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max/stddev = 1.231/1.453/1.892/0.234 ms
```

4. CFM Linktrace

```
user@router> traceroute ethernet maintenance-domain PROVIDER maintenance-association CUSTOMER-A-VLAN100 mep 501 remote-mep
502
Hop  MAC Address         Ingr-Port    Egr-Port     Action  Relay
1    00:11:11:11:11:11  ge-0/0/0.100 ge-0/0/10.100  FDB    Hit
2    00:22:22:22:22:22  ge-0/1/0.100 ge-0/1/1.100   FDB    Hit
3    00:33:33:33:33:33  ge-0/2/0.100 ge-0/2/1.100   FDB    Hit
```

Common Troubleshooting Scenarios

Scenario 1: LFM Not Establishing

Symptom: LFM shows "Discovery State: Passive Wait"

Diagnostic Commands:

```
user@router> show oam ethernet link-fault-management interface ge-0/0/0
Discovery State: Passive Wait
Remote: State: Unknown

user@router> monitor traffic interface ge-0/0/0 matching "0x8809"
(No OAMPDUs seen)
```

Cause: Remote side not configured or blocked **Solution**:

```
# Verify remote configuration
# Check if provider blocks slow protocols
set protocols oam ethernet link-fault-management interface ge-0/0/0 link-discovery passive

# Or try active mode
set protocols oam ethernet link-fault-management interface ge-0/0/0 link-discovery active
commit
```

Scenario 2: CFM Remote MEP Down

Symptom: Remote MEP showing as down

Diagnostic Commands:

```
user@router> show oam ethernet connectivity-fault-management mep-database
Remote MEP: 502
  Status: Down
  Last CCM received: 00:05:32 ago

user@router> show log messages | match CFM
Nov 21 10:15:44 CFM_REMOTE_MEP_DOWN: Domain: PROVIDER, MA: CUSTOMER-A-VLAN100, MEP: 502
```

Cause: CCMs not being received **Solution**:

```
# Check VLAN configuration
show vlans 100
show interfaces ge-0/0/0.100

# Verify CCM interval matches on both sides
show configuration protocols oam ethernet connectivity-fault-management | match interval

# Test with loopback
ping ethernet maintenance-domain PROVIDER maintenance-association CUSTOMER-A-VLAN100 mep 501 remote-mep 502
```

Scenario 3: CFM Level Conflicts

Symptom: MIPs not responding to linktrace

Diagnostic Commands:

```
user@router> traceroute ethernet maintenance-domain PROVIDER maintenance-association CUSTOMER-A-VLAN100
Hop  MAC Address        Action
1    00:11:11:11:11:11  No Response
2    00:33:33:33:33:33  FDB Hit

user@router> show log messages | match "CFM.*level"
Nov 21 11:30:15 CFM_LEVEL_CONFLICT: Received CFM PDU at wrong level
```

Cause: Overlapping maintenance domains at same level **Solution**:

```
# Ensure unique levels for overlapping domains
show configuration protocols oam ethernet connectivity-fault-management | match "level|domain"

# Adjust domain levels
set protocols oam ethernet connectivity-fault-management maintenance-domain OPERATOR level 3
set protocols oam ethernet connectivity-fault-management maintenance-domain PROVIDER level 5
set protocols oam ethernet connectivity-fault-management maintenance-domain CUSTOMER level 7
commit
```

Scenario 4: Performance Monitoring Not Working

Symptom: No delay/loss measurements appearing

Diagnostic Commands:

```
user@router> show oam ethernet connectivity-fault-management sla-iterator-statistics
No statistics available

user@router> show configuration protocols oam ethernet connectivity-fault-management maintenance-domain PROVIDER
maintenance-association CUSTOMER-A-VLAN100 mep 501
## sla-iterator-profile not configured!
```

Cause: SLA iterator profile not applied to MEP **Solution**:

```
# Create SLA profile
set protocols oam ethernet connectivity-fault-management performance-monitoring sla-iterator-profile MONITOR measurement-
type two-way-delay

# Apply to MEP
set protocols oam ethernet connectivity-fault-management maintenance-domain PROVIDER maintenance-association CUSTOMER-A-
VLAN100 mep 501 sla-iterator-profile MONITOR
commit

# Verify after few minutes
show oam ethernet connectivity-fault-management sla-iterator-statistics
```

Scenario 5: LFM Detecting Errors but Link Stays Up

Symptom: High error rates in LFM statistics

Diagnostic Commands:

```
user@router> show oam ethernet link-fault-management interface ge-0/0/0 detail
  Error Statistics:
    Symbol errors: 1523
    Frame errors: 234
    Frame period errors: 89
```

```
user@router> show configuration protocols oam ethernet link-fault-management interface ge-0/0/0
## No action-profile configured
```

Cause: No action profile to act on errors **Solution**:

```
# Create action profile
set protocols oam ethernet link-fault-management action-profile PROTECT event link-event-rate symbol-period 100
set protocols oam ethernet link-fault-management action-profile PROTECT action syslog
set protocols oam ethernet link-fault-management action-profile PROTECT action link-down

# Apply to interface
set protocols oam ethernet link-fault-management interface ge-0/0/0 action-profile PROTECT
commit
```

Pro Tips for Ethernet OAM

1. **Start Simple**:
   - LFM for point-to-point link monitoring
   - CFM for service monitoring
   - Add complexity gradually
2. **CCM Interval Planning**:

   ```
   Critical Services: 100ms-1s
   Standard Services: 1s-10s
   Non-critical: 1m-10m
   ```

3. **Use Appropriate Levels**:

   ```
   7: Customer equipment
   6: Customer service
   5: Provider service
   4: Provider test
   3: Operator transport
   2: Operator test
   1: Reserved
   0: Reserved
   ```

4. **Monitor Resource Usage**:

   ```
   show oam ethernet connectivity-fault-management statistics
   ```

5. **Combine with Other Features**:
   - Use with BFD for faster detection
   - Integrate with routing protocols
   - Trigger SNMP traps for NMS

This module has provided comprehensive coverage of Ethernet OAM, from basic link monitoring with LFM to sophisticated service assurance with CFM. These tools transform Ethernet from a simple LAN technology into a carrier-grade service platform with full visibility, fault detection, and performance monitoring capabilities essential for service provider networks.

Module 9: Configuring OAM

Part 1: The Conceptual Lecture (The Why)

From Theory to Practice: Deploying OAM in Production

Module 8 introduced OAM concepts. Module 9 focuses on real-world deployment scenarios and advanced configurations that service providers use daily.

OAM Deployment Strategies

Strategy 1: Layered OAM Architecture

Service providers deploy OAM in layers, each serving a specific purpose:

```
Customer Visibility Layer (CFM Level 7)
├── End-to-end service monitoring
├── Customer portal integration
└── SLA reporting

Service Delivery Layer (CFM Level 5)
├── PE-to-PE monitoring
├── Per-VLAN service health
└── Fault isolation

Infrastructure Layer (CFM Level 3 + LFM)
├── Transport network monitoring
├── Physical link health
└── Capacity planning data
```

Strategy 2: Proactive vs Reactive Monitoring

Proactive Monitoring:

- Continuous CCM messages detect failures before customers notice
- Performance monitoring tracks degradation trends
- Threshold alerts prevent SLA violations

Reactive Tools:

- Loopback for immediate connectivity testing
- Linktrace for path discovery during outages
- Remote loopback for circuit testing

Advanced LFM Deployment Scenarios

Scenario 1: Metro Ethernet Access

```
Customer ← LFM → CPE ← LFM → Access Switch ← LFM → PE Router
         Link 1       Link 2              Link 3

Each link monitored independently:
- Link 1: Detect customer cable issues
- Link 2: Monitor last-mile fiber
- Link 3: Track aggregation health
```

Scenario 2: Dying Gasp Implementation

When equipment loses power, it sends a "dying gasp" message:

```
Normal Operation:          Power Failure:
CPE → PDUs → Switch       CPE → DYING GASP! → Switch
                               ↓
                          Capacitor power
                          sends final PDU
```

This differentiates power failures from fiber cuts, enabling appropriate dispatch.

Advanced CFM Deployment Patterns

Pattern 1: Service Multiplexing

Multiple services (VLANs) on same physical infrastructure:

```
Physical:  [CE1]--------[PE1]====Core====[PE2]--------[CE2]

Logical:   VLAN 100: MA-GOLD   (CCM interval: 100ms)
          VLAN 200: MA-SILVER (CCM interval: 1s)
          VLAN 300: MA-BRONZE (CCM interval: 10s)
```

Different monitoring intensities based on service tier.

Pattern 2: Dual-Homed Services

Redundant connections require sophisticated monitoring:

```
        [PE1]
       /      \
    [CE1]    [CE2]
       \      /
        [PE2]

Four MEPs monitor all paths:
- CE1-PE1 path
- CE1-PE2 path
- CE2-PE1 path
- CE2-PE2 path
```

Pattern 3: Hub and Spoke Monitoring

Central site monitors all remote sites:

```
           [Hub PE]
          /   |   \
      [PE1] [PE2] [PE3]
        |     |     |
     [CE1] [CE2] [CE3]

Hub MEP monitors multiple remote MEPs
Efficient for centralized management
```

Performance Monitoring Deep Dive

Delay Measurement Methods

1. **One-Way Delay** (requires clock sync):

```
CE1 → DMM(T1) → CE2
CE2 records: Delay = T2 - T1
```

2. **Two-Way Delay** (no clock sync needed):

```
CE1 → DMM(T1) → CE2
CE2 → DMR(T2,T3) → CE1
CE1 calculates: RTT = T4-T1-(T3-T2)
```

Loss Measurement

Synthetic loss measurement using counters:

```
Frame Count Method:
MEP1: Sent 1000 frames
MEP2: Received 995 frames
```

```
Loss = 0.5%

Data Loss Method:
Counts actual data frames
More accurate for SLA
```

Integration with Other Technologies

OAM + BFD

- OAM for service monitoring
- BFD for rapid failure detection
- OAM provides details, BFD provides speed

OAM + MPLS

- CFM monitors Ethernet services
- MPLS OAM monitors transport
- Correlate both for complete view

OAM + Routing

- CFM failures can trigger routing changes
- Track CFM status in routing protocols
- Automatic service rerouting

Part 2: The Junos CLI Masterclass (The How)

Enterprise Deployment Example

Complete configuration for enterprise customer with SLA monitoring:

```
## Define SLA requirements
protocols {
    oam {
        ethernet {
            connectivity-fault-management {
                ## Performance monitoring profiles
                performance-monitoring {
                    sla-iterator-profile GOLD-SLA {
                        measurement-type two-way-delay;
                        calculation-weight {
                            delay 1;
                            delay-variation 2;
                        }
                        cycle-time 10000;        ## 10 seconds
                        iteration-period 300000;  ## 5 minutes
                    }

                    sla-iterator-profile MEASURE-LOSS {
                        measurement-type loss;
                        cycle-time 1000;         ## 1 second
                        iteration-period 60000;   ## 1 minute
                    }
                }

                ## Maintenance domains
                maintenance-domain CUSTOMER-EDGE {
                    level 7;
                    name-format character-string;

                    maintenance-association GOLD-SERVICE {
                        short-name-format vlan-id;
                        vlan-id 100;

                        continuity-check {
                            interval 100ms;
```

```
                        loss-threshold 3;
                        hold-interval 3;
                    }

                    mep 1000 {
                        interface ge-0/0/0.100;
                        direction up;
                        remote-mep 2000;

                        ## Apply both profiles
                        sla-iterator-profile [ GOLD-SLA MEASURE-LOSS ];

                        ## Thresholds for alarming
                        alarm {
                            adjacency-loss;
                            rdi;
                            remote-mep-loss;
                        }
                    }
                }
            }
        }
    }
}
```

Service Provider Hub Configuration

Complex multi-tenant monitoring setup:

```
## Multiple customers on same PE
protocols {
    oam {
        ethernet {
            connectivity-fault-management {
                ## Provider infrastructure monitoring
                maintenance-domain INFRASTRUCTURE {
                    level 3;
                    name-format mac-address;

                    maintenance-association CORE-LINKS {
                        continuity-check {
                            interval 10ms;  ## Ultra-fast detection
                        }
                        mep 301 {
                            interface xe-0/0/0;
                            direction down;
                            priority 7;     ## Highest priority
                        }
                    }
                }

                ## Per-customer domains
                maintenance-domain SP-SERVICES {
                    level 5;
                    name-format character-string;

                    ## Customer A - Financial Services
                    maintenance-association CUSTOMER-A-PRIMARY {
                        short-name-format string;
                        short-name "CUST-A-PRI";
                        vlan-id-list [ 100 101 102 ];

                        continuity-check {
                            interval 100ms;
                            loss-threshold 2;
                        }

                        mep 5001 {
                            interface ge-0/0/1.100;
```

```
                        direction down;
                        remote-mep [ 5002 5003 5004 ];  ## Multiple sites

                        lowest-priority-defect mac-status;
                    }
                }

                ## Customer B - Standard Business
                maintenance-association CUSTOMER-B-SITES {
                    vlan-id 200;

                    continuity-check {
                        interval 1s;
                    }

                    mep 5010 {
                        interface ge-0/0/2.200;
                        direction down;
                        remote-mep [ 5011 5012 ];
                    }
                }
            }

            ## Action profiles
            action-profile SLA-VIOLATION {
                event {
                    adjacency-loss;
                    rdi;
                }
                action {
                    log-and-trap;
                    ## Send SNMP trap to NMS
                }
            }

            ## Apply profiles
            apply-action-profile SLA-VIOLATION;

            ## Global settings
            linktrace {
                age 3600;        ## 1 hour
                size 4096;       ## Larger cache
            }
        }
    }
  }
}
```

Advanced LFM with Automatic Protection

```
protocols {
    oam {
        ethernet {
            link-fault-management {
                ## Define multiple action profiles
                action-profile MINOR-ERRORS {
                    event {
                        link-event-rate {
                            symbol-period 10;
                            frame-error 5;
                        }
                    }
                    action {
                        syslog;
                        snmp-trap;
                    }
                }

                action-profile MAJOR-ERRORS {
                    event {
```

```
                    link-event-rate {
                        symbol-period 100;
                        frame-error 50;
                    }
                    critical-event;
                    dying-gasp;
                }
                action {
                    syslog;
                    link-down;
                    ## Optionally trigger interface bounce
                    clear-interface;
                }
            }

            action-profile LINK-DEGRADE {
                event {
                    threshold-event;
                    link-adjacency-loss;
                }
                action {
                    syslog;
                    ## Adjust interface metrics
                    increase-interface-metric;
                }
            }

            ## Interface templates
            interface-range CUSTOMER-FACING {
                member "ge-0/0/[0-9]";
                member "ge-0/1/[0-9]";

                apply-action-profile [ MINOR-ERRORS MAJOR-ERRORS ];
                link-discovery passive;
                pdu-interval 333;        ## 3 per second

                event-thresholds {
                    symbol-period 1 per 100000;
                    frame-error 1 per 10000;
                    frame-period 1 per 1000;
                }
            }

            interface-range CORE-FACING {
                member "xe-0/0/[0-3]";

                apply-action-profile LINK-DEGRADE;
                link-discovery active;
                pdu-interval 100;        ## 10 per second
                remote-loopback;

                negotiation-options {
                    allow-remote-loopback;
                    no-allow-link-events;
                }
            }
        }
    }
  }
}
```

Integration with Other Features

```
## BFD + CFM Integration
protocols {
    oam {
        ethernet {
            connectivity-fault-management {
                maintenance-domain SERVICES {
                    maintenance-association CRITICAL {
```

```
                        mep 1001 {
                            interface ge-0/0/0.100;
                            ## Track CFM state
                            connectivity-fault-management-tracking;
                        }
                    }
                }
            }
        }
    }
}

## Use CFM state in routing
routing-instances {
    CUSTOMER-A {
        routing-options {
            static {
                route 10.1.1.0/24 {
                    next-hop 192.168.1.1;
                    ## Backup route activates if CFM fails
                    qualified-next-hop 192.168.2.1 {
                        preference 10;
                    }
                    ## Track CFM MEP
                    oam-liveness-detection {
                        ethernet {
                            cfm {
                                maintenance-domain SERVICES;
                                maintenance-association CRITICAL;
                                mep 1001;
                            }
                        }
                    }
                }
            }
        }
    }
}

## Event policies for automation
event-options {
    policy CFM-FAILURE-RESPONSE {
        events oamd_cfm_adjacency_down;
        within 10 {
            trigger on 3;  ## 3 failures in 10 seconds
        }
        then {
            execute-commands {
                commands {
                    "activate redundant-link";
                    "show oam ethernet connectivity-fault-management mep-database";
                }
            }
        }
    }
}
```

Part 3: Verification & Troubleshooting (The What-If)

Advanced Verification Commands

1. Detailed Performance Statistics

```
user@router> show oam ethernet connectivity-fault-management sla-iterator-statistics maintenance-domain SP-SERVICES
maintenance-association CUSTOMER-A-PRIMARY mep 5001 remote-mep 5002

Maintenance domain: SP-SERVICES, Level: 5
Maintenance association: CUSTOMER-A-PRIMARY
Local MEP: 5001, Remote MEP: 5002
```

```
Delay Statistics:
  Average two-way delay              : 4.532 ms
  Average two-way delay variation    : 0.234 ms
  Best case two-way delay           : 3.998 ms
  Worst case two-way delay          : 6.234 ms

Loss Statistics:
  Average forward loss ratio        : 0.01 %
  Average backward loss ratio       : 0.02 %
  Forward loss episodes            : 2
  Backward loss episodes           : 1
```

2. Check Action Profile Triggers

```
user@router> show oam ethernet link-fault-management action-profile-statistics
Action Profile: MAJOR-ERRORS
  Triggered: 3 times
  Last triggered: 2023-11-21 14:30:15 UTC
  Interface: ge-0/0/3
  Event: frame-error rate exceeded
  Action taken: link-down
```

3. Monitor Real-Time CFM

```
user@router> monitor ethernet connectivity-fault-management continuity-check maintenance-domain SP-SERVICES
Monitoring CFM CCMs for domain SP-SERVICES...

14:30:15.123 CCM received from 5002 seq 12345
14:30:15.223 CCM received from 5003 seq 23456
14:30:15.323 CCM received from 5004 seq 34567
14:30:15.423 CCM timeout for 5002 seq 12346 <-- Missing CCM!
```

Complex Troubleshooting Scenarios

Scenario 1: Intermittent CFM Flapping

Symptom: MEP randomly shows up/down

Diagnostic Commands:

```
user@router> show oam ethernet connectivity-fault-management mep-statistics 5001
CCMs transmitted: 864000
CCMs received: 863892
Out-of-sequence CCMs: 47
CCMs with bad MSDU: 12

user@router> show interfaces queue ge-0/0/1 forwarding-class network-control
Queue: 3, Forwarding classes: network-control
  Queued:
    Packets              :            2341234
    Bytes                :         234123400
  Transmitted:
    Packets              :            2341189
    Bytes                :         234118900
    Tail-dropped packets :                 45  <-- Queue drops!
```

Cause: Network control queue dropping CCMs during congestion **Solution**:

```
# Increase network-control queue allocation
set class-of-service interfaces ge-0/0/1 scheduler-map PROTECT-CCM
```

```
set class-of-service scheduler-maps PROTECT-CCM forwarding-class network-control scheduler NC-SCHEDULER
set class-of-service schedulers NC-SCHEDULER transmit-rate percent 10
set class-of-service schedulers NC-SCHEDULER buffer-size percent 10
set class-of-service schedulers NC-SCHEDULER priority high
commit
```

Scenario 2: Performance Measurements Inconsistent

Symptom: Delay measurements vary wildly

Diagnostic Commands:

```
user@router> show oam ethernet connectivity-fault-management delay-statistics remote-mep 5002 detail
Two-way delay samples:
  Sample 1: 4.123 ms
  Sample 2: 4.234 ms
  Sample 3: 45.234 ms  <-- Spike!
  Sample 4: 4.345 ms

user@router> show system processes extensive | match oamd
12345 oamd     85.2%  <-- High CPU usage
```

Cause: CPU spikes affecting timestamping **Solution**:

```
# Move OAM processing to dedicated resources
set system processes oam-process dedicated-resources

# Adjust measurement timing
set protocols oam ethernet connectivity-fault-management performance-monitoring sla-iterator-profile GOLD-SLA cycle-time
30000
set protocols oam ethernet connectivity-fault-management performance-monitoring sla-iterator-profile GOLD-SLA iteration-
period 600000
commit

# Consider hardware-assisted timestamping
set interfaces ge-0/0/1 gigether-options oam-on-npu
```

Scenario 3: LFM Not Detecting Errors

Symptom: Physical errors occur but LFM doesn't report

Diagnostic Commands:

```
user@router> show interfaces ge-0/0/3 extensive | match FCS
  Link-level type: Ethernet, FCS errors: 12345  <-- Physical errors

user@router> show oam ethernet link-fault-management interface ge-0/0/3
  Error Statistics:
    Symbol errors: 0
    Frame errors: 0  <-- Not detected by OAM!

user@router> show configuration protocols oam ethernet link-fault-management interface ge-0/0/3
## No error thresholds configured
```

Cause: Error detection thresholds not configured **Solution**:

```
# Configure appropriate thresholds
set protocols oam ethernet link-fault-management interface ge-0/0/3 event-thresholds symbol-period 1 per 1000000
set protocols oam ethernet link-fault-management interface ge-0/0/3 event-thresholds frame-error 1 per 100000
set protocols oam ethernet link-fault-management interface ge-0/0/3 event-thresholds frame-period 1 per 10000
```

```
# Enable error monitoring
set protocols oam ethernet link-fault-management interface ge-0/0/3 pdu-threshold 3
commit
```

Scenario 4: Multi-Domain Interaction Issues

Symptom: Lower level CFM blocking higher level

Diagnostic Commands:

```
user@router> traceroute ethernet maintenance-domain CUSTOMER-EDGE maintenance-association GOLD-SERVICE mep 1000
Hop  MAC Address         Response
1    00:11:11:11:11:11  Level filtered  <-- Blocked by lower level

user@router> show log messages | match "CFM.*filtered"
CFM_PDU_FILTERED: Level 7 PDU filtered by level 5 MEP
```

Cause: MEP direction configuration incorrect **Solution**:

```
# Check MEP directions
show configuration protocols oam ethernet connectivity-fault-management | match "direction|level" | display set

# Provider MEPs should face down (toward customer)
# Customer MEPs should face up (toward network)
set protocols oam ethernet connectivity-fault-management maintenance-domain SP-SERVICES maintenance-association CUSTOMER-A-PRIMARY mep 5001 direction down
set protocols oam ethernet connectivity-fault-management maintenance-domain CUSTOMER-EDGE maintenance-association GOLD-SERVICE mep 1000 direction up
commit
```

Scenario 5: SLA Iterator Not Collecting Data

Symptom: No performance data after hours of operation

Diagnostic Commands:

```
user@router> show oam ethernet connectivity-fault-management sla-iterator-statistics
No statistics to display

user@router> show log messages | match SLA
SLA_ITERATOR_RESOURCE_LIMIT: Maximum iterators (64) reached
```

Cause: Resource limits preventing measurements **Solution**:

```
# Check current iterator usage
show oam ethernet connectivity-fault-management sla-iterator-profiles | count

# Remove unused profiles
delete protocols oam ethernet connectivity-fault-management performance-monitoring sla-iterator-profile UNUSED-PROFILE

# Adjust iterator parameters
set protocols oam ethernet connectivity-fault-management performance-monitoring sla-iterator-profile GOLD-SLA max-iterations 100
commit

# Monitor resource usage
show system resource-monitor
```

Pro Tips for Production OAM

1. **Start Conservative**: Begin with longer intervals, decrease gradually

```
Week 1: CCM interval 10s
Week 2: CCM interval 1s
Week 3: CCM interval 100ms (if stable)
```

2. **Use Templates**: Create standard profiles

```
GOLD: 100ms CCM, 1% loss threshold
SILVER: 1s CCM, 3% loss threshold
BRONZE: 10s CCM, 5% loss threshold
```

3. **Correlate Multiple Signals**:
   - LFM for physical layer
   - CFM for service layer
   - Interface statistics for validation
4. **Automate Response**:

```
set event-options generate-event CFM-FAILURE time-interval 60
set event-options policy CFM-AUTO-RECOVER events CFM-FAILURE
set event-options policy CFM-AUTO-RECOVER then execute-commands commands "restart oam-process"
```

5. **Document MEP Allocation**:

```
## MEP ID Allocation Scheme ##
1-999:     Reserved
1000-1999: Customer Edge MEPs
2000-2999: Remote Site MEPs
5000-5999: Provider MEPs
9000-9999: Test MEPs
```

This module has provided advanced OAM configuration techniques and real-world troubleshooting scenarios. The combination of LFM for physical monitoring and CFM for service assurance creates a comprehensive OAM framework that enables proactive network management and rapid fault isolation - essential capabilities for maintaining carrier-grade Ethernet services.

Module 10: ERP and LAG

Part 1: The Conceptual Lecture (The Why)

The Fundamental Problems

Problem 1: Ring Topology Protection

Many service provider networks use ring topologies for cost efficiency:

- Fiber follows roads/railways in circles
- Each node connects to exactly two others
- More economical than full mesh

But rings have a critical issue: How do you provide redundancy without creating loops?

```
Ring Without Protection:        Ring With Loop:
    [A]---[B]                      [A]---[B]
     |     |                        |     |
    [D]---[C]                      [D]---[C]
         ↓                               ↓
   If B-C fails:                  If all links active:
   D→C traffic lost               Broadcast storm!
```

Problem 2: Bandwidth Limitations

Single links create bottlenecks:

- 10G link carrying 8G traffic
- Add 3G more traffic → Drops!
- Upgrade to 40G → Expensive and disruptive

What if you could combine multiple 10G links?

Ethernet Ring Protection (ERP) - ITU-T G.8032

ERP provides sub-50ms protection switching for Ethernet rings without using Spanning Tree Protocol.

How ERP Works

1. **Ring Protection Link (RPL)**: One link is blocked to prevent loops
2. **Ring APS Messages**: Automatic Protection Switching messages
3. **Failure Detection**: When link fails, RPL unblocks
4. **Recovery**: When fault clears, RPL blocks again

```
Normal Operation:         Link Failure:          Recovery:
    [A]---[B]                [A]---[B]              [A]---[B]
     |     |                  |     X                |     |
    [D]---[C]                [D]---[C]              [D]---[C]
       RPL↑                      ↓                     RPL↑
    (Blocked)              RPL Unblocks             (Blocked)
```

ERP Components

1. **RPL Owner**: Node that blocks/unblocks the RPL
2. **RPL Neighbor**: Node connected to RPL Owner
3. **Ring Nodes**: All other nodes in ring
4. **APS Channel**: VLAN carrying protection messages

ERP Versions

- **ERPv1**: Basic ring protection
- **ERPv2**: Adds multi-ring, sub-ring support

ERP Messages (R-APS)

```
R-APS PDU Format:
+----------+--------+--------+--------+
| MEL      | OpCode | Flags  | TLV    |
| (3 bits) | (8b)   | (8b)   | (var)  |
+----------+--------+--------+--------+

Key Messages:
- NR (No Request): Normal state
- SF (Signal Fail): Link failure detected
- NR,RB (No Request, RPL Blocked): RPL is blocked
- FS (Forced Switch): Manual switch
```

Link Aggregation Groups (LAG) - IEEE 802.3ad

LAG combines multiple physical links into one logical link, providing:

- Increased bandwidth
- Redundancy
- Load balancing

LAG Concepts

```
Without LAG:              With LAG:
[Switch A]                [Switch A]
  | | | |                     |
  4 x 1G                    LAG0
  | | | |                  (4G Total)
[Switch B]                    |
                         [Switch B]
```

How LAG Works

1. **Link Aggregation Control Protocol (LACP)**
   - Negotiates LAG membership
   - Monitors link health
   - Standards-based (vs static LAG)
2. **Load Balancing Algorithms**
   - Layer 2: Based on MAC addresses
   - Layer 3: Based on IP addresses
   - Layer 4: Based on ports
   - Adaptive: Dynamic rebalancing
3. **Member Link States**
   - Active: Passing traffic
   - Standby: Ready but not used
   - Down: Failed or disabled

LACP Protocol Details

```
LACPDU Format:
+------------------+
| Subtype (1)      |
| Version (1)      |
| Actor Info TLV   |
| Partner Info TLV |
| Collector TLV    |
| Terminator TLV   |
+------------------+

Key Information:
- System ID: Identifies device
- Port Priority: Determines active/standby
- Port Number: Physical port
- State: Active/Passive/Aggregatable
```

Why Use ERP vs STP for Rings?

Feature	STP	ERP
Convergence Time	1-50 seconds	<50ms
Complexity	High	Low
Ring Optimized	No	Yes
Multi-Ring	Complex	Native
Standard	IEEE 802.1D	ITU-T G.8032

Why Use LAG vs Single Links?

Feature	Single Link	LAG
Bandwidth	Fixed	Scalable
Redundancy	None	N-1 links

Feature	Single Link	LAG
Upgrade	Disruptive	Add links
Load Sharing	No	Yes
Cost	Per link speed	Aggregate small links

Part 2: The Junos CLI Masterclass (The How)

Configuring Ethernet Ring Protection

Step 1: Basic Ring Configuration

```
# Define protection group
set protocols protection-group ethernet-ring RING-1 {
    ring-id 1;
    restoration-interval 5;    ## Wait 5 minutes before restoring
    guard-interval 500;        ## 500ms guard time

    ## Define east and west interfaces
    east-interface {
        control-channel ge-0/0/0.100;
        ring-protection-link-owner;    ## This node owns RPL
    }
    west-interface {
        control-channel ge-0/0/1.100;
    }

    ## Data channels (VLANs to protect)
    data-channel {
        vlan [ 200-299 ];
    }
}
```

Key concepts:

- `ring-id` : Unique identifier for ring
- `restoration-interval` : Prevents flapping
- `control-channel` : Carries R-APS messages
- `data-channel` : VLANs protected by ring

Step 2: Configure Ring Nodes

```
## RPL Owner Node
set protocols protection-group ethernet-ring RING-1 {
    ring-id 1;
    east-interface {
        control-channel ge-0/0/0.100;
        ring-protection-link-owner;    ## RPL owner
    }
    west-interface {
        control-channel ge-0/0/1.100;
    }
    data-channel {
        vlan [ 200-299 ];
    }
}

## RPL Neighbor Node
set protocols protection-group ethernet-ring RING-1 {
    ring-id 1;
    east-interface {
        control-channel ge-0/0/0.100;
    }
    west-interface {
        control-channel ge-0/0/1.100;
        ring-protection-link-neighbor;  ## RPL neighbor
    }
```

```
    data-channel {
        vlan [ 200-299 ];
    }
}

## Regular Ring Node
set protocols protection-group ethernet-ring RING-1 {
    ring-id 1;
    east-interface {
        control-channel ge-0/0/0.100;
    }
    west-interface {
        control-channel ge-0/0/1.100;
    }
    data-channel {
        vlan [ 200-299 ];
    }
}
```

Step 3: Configure VLANs for ERP

```
## Control channel VLAN
set vlans CONTROL-VLAN {
    vlan-id 100;
    ## Add ring interfaces
    interface ge-0/0/0.100;
    interface ge-0/0/1.100;
}

## Data VLANs
set vlans DATA-VLAN-200 {
    vlan-id 200;
    interface ge-0/0/0.200;
    interface ge-0/0/1.200;
    ## Add customer-facing interfaces
    interface ge-0/0/10.200;
}
```

Configuring Link Aggregation

Step 1: Basic LAG Configuration

```
# Create aggregated interface
set interfaces ae0 {
    aggregated-ether-options {
        lacp {
            active;    ## Actively send LACPDU
            periodic fast;    ## 1 second interval
        }
    }
    unit 0 {
        family ethernet-switching {
            interface-mode trunk;
            vlan {
                members [ 100-200 ];
            }
        }
    }
}

# Add physical interfaces to LAG
set interfaces ge-0/0/0 {
    gigether-options {
        802.3ad ae0;
    }
}
set interfaces ge-0/0/1 {
    gigether-options {
        802.3ad ae0;
```

```
    }
}
```

Step 2: Advanced LAG Options

```
set interfaces ae0 {
    aggregated-ether-options {
        lacp {
            active;
            periodic fast;
            system-id 00:11:22:33:44:55;
            system-priority 100;    ## Lower = higher priority

            ## Accept slower LACP from partner
            accept-data-on-aggregation-sans-lacp;
        }

        ## Minimum links before LAG is up
        minimum-links 2;

        ## Maximum links in LAG
        link-speed 10g;

        ## Load balancing
        load-balance {
            per-packet;    ## or adaptive, per-flow
        }
    }
}
```

Step 3: LAG with Member Link Options

```
## Configure LACP on member interfaces
set interfaces ge-0/0/0 {
    gigether-options {
        802.3ad {
            ae0;
            lacp {
                port-priority 100;    ## Higher priority
                force-up;            ## Force active
            }
        }
    }
}

set interfaces ge-0/0/1 {
    gigether-options {
        802.3ad {
            ae0;
            lacp {
                port-priority 200;    ## Lower priority (backup)
            }
        }
    }
}
```

Complete Reference Configuration

```
## Interfaces Configuration
interfaces {
    ## LAG Configuration
    ae0 {
        description "LAG to Core Switch";
        aggregated-ether-options {
            lacp {
                active;
                periodic fast;
                system-priority 32768;
```

```
            }
            minimum-links 2;
            link-protection;    ## 1:1 link protection
        }
        unit 0 {
            family ethernet-switching {
                interface-mode trunk;
                vlan {
                    members [ 100-299 ];
                }
            }
        }
    }

    ## LAG Members
    ge-0/0/0 {
        description "LAG ae0 member 1";
        gigether-options {
            802.3ad ae0;
        }
    }
    ge-0/0/1 {
        description "LAG ae0 member 2";
        gigether-options {
            802.3ad ae0;
        }
    }

    ## ERP Ring Interfaces
    ge-0/0/10 {
        description "Ring East Interface";
        flexible-vlan-tagging;
        encapsulation flexible-ethernet-services;
        unit 100 {
            description "ERP Control Channel";
            vlan-id 100;
            family ethernet-switching;
        }
        unit 200 {
            description "Customer Data";
            vlan-id 200;
            family ethernet-switching;
        }
    }
    ge-0/0/11 {
        description "Ring West Interface";
        flexible-vlan-tagging;
        encapsulation flexible-ethernet-services;
        unit 100 {
            vlan-id 100;
            family ethernet-switching;
        }
        unit 200 {
            vlan-id 200;
            family ethernet-switching;
        }
    }
}

## ERP Configuration
protocols {
    protection-group {
        ethernet-ring METRO-RING {
            ring-id 1;
            restoration-interval 5;
            guard-interval 500;
            hold-interval 0;

            east-interface {
                control-channel {
                    ge-0/0/10.100;
```

```
                }
                ## Only on RPL owner node:
                ring-protection-link-owner;
            }

            west-interface {
                control-channel {
                    ge-0/0/11.100;
                }
            }

            data-channel {
                vlan [ 200-299 ];
            }

            ## Version 2 features
            compatibility-version 2;
            revertive;    ## Restore RPL after fault clears
        }
    }
}

## VLAN Configuration
vlans {
    CONTROL {
        vlan-id 100;
        ## No MAC learning on control VLAN
        no-mac-learning;
    }
    CUSTOMER-DATA {
        vlan-id 200;
        ## Regular data VLAN
    }
}

## Class of Service for ERP
class-of-service {
    interfaces {
        ge-0/0/10 {
            unit 100 {
                classifiers {
                    dscp CONTROL-CLASSIFIER;
                }
                rewrite-rules {
                    dscp CONTROL-REWRITE;
                }
            }
        }
    }
}
```

Part 3: Verification & Troubleshooting (The What-If)

Essential Verification Commands

1. Check ERP Status

```
user@router> show protection-group ethernet-ring
Protection group: METRO-RING
  Ring ID: 1, Node ID: 00:11:22:33:44:55
  Ring state: Idle
  RPL role: Owner
  RPL state: Blocked
  East interface: ge-0/0/10.100 (Up)
  West interface: ge-0/0/11.100 (Up)
  Ring protocol: G.8032v2

Statistics:
```

```
  R-APS messages sent: 12345
  R-APS messages received: 12340
  Local SF detected: 2
  Remote SF detected: 3
```

2. Check LAG Status

```
user@router> show interfaces ae0 extensive
Physical interface: ae0, Enabled, Physical link is Up
  Interface index: 128, SNMP ifIndex: 501
  Link-level type: Ethernet, Speed: 20Gbps

  Aggregate member links: 2

  LACP info:
    Local System ID: 00:11:22:33:44:55
    Local System priority: 32768
    Partner System ID: 00:aa:bb:cc:dd:ee
    Partner System priority: 32768
    LACP State: Active Fast

  Aggregate member links:
    ge-0/0/0: Active
    ge-0/0/1: Active
```

3. Check LACP Details

```
user@router> show lacp interfaces ae0
Aggregated interface: ae0
  LACP state: Role   Exp   Def  Dist  Col  Syn  Aggr  Timeout  Activity
  ge-0/0/0   Actor    No    No   Yes  Yes  Yes   Yes     Fast    Active
             Partner  No    No   Yes  Yes  Yes   Yes     Fast    Active
  ge-0/0/1   Actor    No    No   Yes  Yes  Yes   Yes     Fast    Active
             Partner  No    No   Yes  Yes  Yes   Yes     Fast    Active

LACP protocol: Receive State  Transmit State  Mux State
  ge-0/0/0     Current        Fast periodic   Collecting distributing
  ge-0/0/1     Current        Fast periodic   Collecting distributing
```

Common Troubleshooting Scenarios

Scenario 1: ERP Not Blocking RPL

Symptom: Loop detected in ring

Diagnostic Commands:

```
user@router> show protection-group ethernet-ring METRO-RING
RPL state: Unblocked   <-- Should be blocked!

user@router> show log messages | match R-APS
ERP: R-APS SF received on east interface
ERP: R-APS SF received on west interface   <-- Loop!
```

Cause: Multiple nodes configured as RPL owner **Solution**:

```
# Check all ring nodes - only ONE should be RPL owner
show configuration protocols protection-group | match owner

# Fix configuration
```

```
delete protocols protection-group ethernet-ring METRO-RING east-interface ring-protection-link-owner
commit
```

Scenario 2: LAG Using Only One Link

Symptom: Traffic not load-balanced across LAG members

Diagnostic Commands:

```
user@router> show interfaces ae0 statistics
  Input packets:  1234567 (1234567 pps)
  Output packets: 1234567 (1234567 pps)

user@router> show interfaces ge-0/0/0 statistics
  Output packets: 1234567 (1234567 pps)   <-- All traffic!

user@router> show interfaces ge-0/0/1 statistics
  Output packets: 0 (0 pps)                <-- No traffic!
```

Cause: Load balancing algorithm not optimal for traffic pattern **Solution**:

```
# Check current load balance method
show configuration interfaces ae0 aggregated-ether-options

# Try different algorithm
set interfaces ae0 aggregated-ether-options load-balance adaptive
# or
set interfaces ae0 aggregated-ether-options load-balance per-packet
commit

# For L3/L4 hashing
set forwarding-options hash-key family inet layer-3
set forwarding-options hash-key family inet layer-4
```

Scenario 3: ERP Slow Convergence

Symptom: Ring takes several seconds to recover

Diagnostic Commands:

```
user@router> show protection-group ethernet-ring detail
Guard timer: 500ms
Hold-off timer: 0ms      <-- Not configured!
WTR timer: 300 seconds   <-- 5 minutes!

user@router> monitor traffic interface ge-0/0/10.100
(Long gaps between R-APS messages)
```

Cause: Timers not optimized **Solution**:

```
# Optimize timers for faster convergence
set protocols protection-group ethernet-ring METRO-RING guard-interval 100
set protocols protection-group ethernet-ring METRO-RING restoration-interval 1
set protocols protection-group ethernet-ring METRO-RING hold-interval 0

# Ensure control VLAN has priority
set class-of-service interfaces ge-0/0/10 unit 100 forwarding-class network-control
commit
```

Scenario 4: LACP Not Forming

Symptom: LAG shows down despite physical links up

Diagnostic Commands:

```
user@router> show lacp interfaces
ge-0/0/0: LACP State: Down
  No LACP PDUs received

user@router> show lacp statistics interfaces ge-0/0/0
LACP PDUs sent: 1234
LACP PDUs received: 0    <-- Not receiving!
```

Cause: LACP blocked or misconfigured on partner **Solution**:

```
# Try passive mode
set interfaces ae0 aggregated-ether-options lacp passive

# Or force LAG without LACP (static)
delete interfaces ae0 aggregated-ether-options lacp
set interfaces ae0 aggregated-ether-options no-lacp

# For mixed environments
set interfaces ae0 aggregated-ether-options lacp accept-data-on-aggregation-sans-lacp
commit
```

Scenario 5: ERP Sub-Ring Issues

Symptom: Sub-ring not protecting properly

Diagnostic Commands:

```
user@router> show protection-group ethernet-ring
Protection group: MAIN-RING
  Ring state: Idle

Protection group: SUB-RING
  Ring state: Protection   <-- Main ring blocks sub-ring
  Virtual channel: Blocked
```

Cause: Sub-ring configuration incorrect **Solution**:

```
# Configure sub-ring with virtual channel
set protocols protection-group ethernet-ring SUB-RING {
    ring-id 2;
    compatibility-version 2;    ## v2 required for sub-rings

    east-interface {
        control-channel ge-0/0/20.101;
    }

    ## Virtual channel to main ring
    virtual-channel {
        control-channel ge-0/0/10.100;
        attach-to-ring MAIN-RING;
    }

    data-channel {
        vlan [ 300-399 ];
    }
}
commit
```

Pro Tips for ERP and LAG

1. **Design Ring Topology Carefully**:

```
- Place RPL at least-loaded link
- Consider traffic patterns
- Document ring layout and RPL location
```

2. **LAG Best Practices**:

```
- Use LACP active on at least one side
- Match speeds on all member links
- Configure consistent MTU
- Use minimum-links for resilience
```

3. **Monitor Ring Health**:

```
set protocols protection-group ethernet-ring RING-1 statistics
set system syslog file ring-events any info
set system syslog file ring-events match "R-APS|RPL"
```

4. **Test Protection Switching**:

```
# Manual switch test
request protection-group ethernet-ring RING-1 manual-switch east

# Clear manual switch
request protection-group ethernet-ring RING-1 clear
```

5. **Combine Technologies**:

```
- Use LAG for ring interfaces (resilient rings)
- Run ERP over LAG
- Add BFD for faster detection
```

This module has covered two critical high-availability technologies. ERP provides rapid protection for ring topologies common in metro networks, while LAG enables bandwidth scaling and redundancy through link bundling. Together, they form the foundation for building resilient, high-performance Ethernet networks that can meet stringent availability requirements while optimizing infrastructure investments.

Module 11: High Availability and Network Optimization

Part 1: The Conceptual Lecture (The Why)

The Next-Level Redundancy Challenge

In Module 10, we solved link redundancy with LAG. But what happens when the entire switch fails?

```
Traditional LAG Limitation:
[Server]
  | | | |
  LAG0
  | | | |
[Switch A] ← If this fails, server loses all connectivity!
```

The Problems:

1. **Single Point of Failure**: One switch failure = total outage
2. **Maintenance Windows**: Can't upgrade switch without downtime
3. **Geographic Limitation**: All LAG members must connect to same device

4. **Scaling Constraints**: Limited by single switch capacity

Multi-Chassis LAG (MC-LAG): The Solution

MC-LAG allows a LAG to span multiple switches, providing device-level redundancy:

```
MC-LAG Architecture:
    [Server/Switch]
      | | | |
      LAG0
     /  |  \
    /   |   \
[Switch A]--[Switch B]
    ICL Link

Server thinks it's connected to one switch!
Actually connected to two coordinated switches.
```

MC-LAG Components

1. ICL (Inter-Chassis Link)

The backbone connecting MC-LAG peers:

- Carries control protocol messages
- Synchronizes MAC learning
- Forwards traffic during failures
- Must be high-bandwidth and redundant

2. ICCP (Inter-Chassis Control Protocol)

Synchronizes state between peers:

- LAG member status
- MAC address tables
- ARP/ND entries
- Configuration consistency

3. MC-LAG Peers

The two switches forming the MC-LAG system:

- **Active**: Primary decision maker
- **Standby**: Backup, ready to take over
- Both forward traffic simultaneously

How MC-LAG Works

Phase 1: Initial Setup

```
1. Establish ICL between switches
2. Exchange ICCP messages
3. Verify configuration consistency
4. Elect active/standby roles
5. Begin accepting LACP from clients
```

Phase 2: Normal Operation

```
Client → LACP → Both MC-LAG switches
         ↓
Switch A and B coordinate:
- Present same LACP System ID
```

```
- Synchronize MAC tables
- Load balance traffic
```

Phase 3: Failure Handling

```
Scenario: Switch A fails
1. Switch B detects peer failure
2. B takes over all LAG members
3. B updates LACP state
4. Client continues without disruption
```

MC-LAG vs Traditional Redundancy

Feature	STP/RSTP	Active/Standby	MC-LAG
Failover Time	1-50 seconds	3-5 seconds	<1 second
Bandwidth Utilization	50% (blocked links)	50% (standby)	100% (all active)
Configuration Complexity	Medium	Low	High
Geographic Distribution	Yes	Yes	Limited

MX Virtual Chassis: Unified Management

Virtual Chassis takes a different approach - making multiple MX routers appear as one logical device:

```
Traditional:                    Virtual Chassis:
[MX-1] Config 1                [VC-MASTER]
  ↓                               /    \
[MX-2] Config 2           [MX-1-RE0]  [MX-2-RE1]
  ↓                               \    /
Must manage separately         Single config!
```

Virtual Chassis Benefits

1. **Simplified Management**: One configuration for multiple chassis
2. **Unified Control Plane**: Single routing instance
3. **Non-Stop Routing**: Hitless RE failover
4. **Flexible Topology**: Members can be geographically separated

Virtual Chassis Architecture

Components:

1. **Virtual Chassis Ports (VCP)**: High-speed links between members
2. **Master RE**: Primary routing engine
3. **Backup RE**: Standby routing engine
4. **Line Card Chassis**: Additional members without REs

Roles:

- **Master**: Runs all control plane protocols
- **Backup**: Synchronized, ready for instant takeover
- **Linecard**: Forwarding only, no local control plane

When to Use What?

MC-LAG: Best for:

- Server/storage connectivity
- Access layer redundancy
- Active-active load sharing
- Quick deployment

Virtual Chassis: Best for:

- Core/aggregation layers
- Unified management requirement
- Campus collapse scenarios
- Long-term architectural decisions

Part 2: The Junos CLI Masterclass (The How)

Configuring MC-LAG

Step 1: Configure ICL (Inter-Chassis Link)

```
## On Switch A
set interfaces ae0 description "ICL to Switch-B"
set interfaces ae0 aggregated-ether-options {
    lacp {
        active;
        periodic fast;
    }
}
set interfaces ae0 unit 0 {
    family ethernet-switching {
        interface-mode trunk;
        vlan {
            members all;
        }
    }
}

## Add physical interfaces to ICL
set interfaces xe-0/0/0 description "ICL member 1"
set interfaces xe-0/0/0 gigether-options 802.3ad ae0
set interfaces xe-0/0/1 description "ICL member 2"
set interfaces xe-0/0/1 gigether-options 802.3ad ae0
```

Step 2: Configure ICCP

```
## On Switch A
set protocols iccp {
    local-ip-addr 192.168.1.1;
    peer 192.168.1.2 {
        session-establishment-hold-time 50;
        backup-liveness-detection {
            backup-peer-ip 10.1.1.2;
        }
        liveness-detection {
            minimum-interval 500;
            multiplier 3;
        }
    }
}

## On Switch B
set protocols iccp {
    local-ip-addr 192.168.1.2;
    peer 192.168.1.1 {
        session-establishment-hold-time 50;
        backup-liveness-detection {
            backup-peer-ip 10.1.1.1;
        }
        liveness-detection {
            minimum-interval 500;
            multiplier 3;
        }
    }
}
```

Step 3: Configure MC-LAG Service

```
## Define MC-LAG parameters (Both switches)
set switch-options {
    service-id 1;    ## Must match on both peers
}

set interfaces ae1 {
    description "MC-LAG to Server";
    mtu 9192;
    aggregated-ether-options {
        lacp {
            active;
            periodic fast;
            system-id 00:11:22:33:44:55;  ## Same on both!
            admin-key 1;
        }
        mc-ae {
            mc-ae-id 1;
            chassis-id 0;                   ## 0 on Switch A, 1 on Switch B
            mode active-active;
            status-control active;          ## Or standby
            init-delay-time 120;
            redundancy-group 1;
        }
    }
}

## Add physical interfaces
set interfaces ge-0/0/10 gigether-options 802.3ad ae1
set interfaces ge-0/0/11 gigether-options 802.3ad ae1

## Configure VLANs
set interfaces ae1 unit 0 family ethernet-switching {
    interface-mode trunk;
    vlan {
        members [ 100-200 ];
    }
}
```

Step 4: Configure Multi-Homing

```
set multi-chassis {
    multi-chassis-protection 192.168.1.2 {  ## Peer IP
        interface ae0;                      ## ICL interface
    }
}

set vlans VLAN100 {
    vlan-id 100;
    mcae-mac-synchronize;                   ## Sync MACs
    interface ae1.0;
}
```

Configuring MX Virtual Chassis

Step 1: Cable and Configure VCP Ports

```
## Convert network ports to VCP
request virtual-chassis vc-port set pic-slot 1 port 0
request virtual-chassis vc-port set pic-slot 1 port 1

## After reboot, verify
show virtual-chassis vc-port
```

Step 2: Form Virtual Chassis

```
## On future master (MX-1)
set virtual-chassis preprovisioned
set virtual-chassis member 0 role routing-engine
set virtual-chassis member 0 serial-number ABC123
set virtual-chassis member 1 role routing-engine
set virtual-chassis member 1 serial-number XYZ789

## Set mastership priority
set virtual-chassis member 0 mastership-priority 255
set virtual-chassis member 1 mastership-priority 100
```

Step 3: Configure Virtual Chassis Properties

```
## Set VC-wide parameters
set virtual-chassis {
    graceful-restart;
    no-split-detection;     ## For 2-member VC
    vcp-snmp-statistics;

    ## Fast failover
    heartbeat-timeout 2000;
    heartbeat-interval 1000;
}

## Configure interfaces spanning members
set interfaces xe-0/0/0 description "Local to member 0"
set interfaces xe-1/0/0 description "On member 1"

## LAG spanning members
set interfaces ae10 {
    description "VC-LAG spanning members";
    aggregated-ether-options {
        lacp {
            active;
            system-id 00:11:22:33:44:66;
        }
    }
}

## Add members from different chassis
set interfaces xe-0/0/5 gigether-options 802.3ad ae10
set interfaces xe-1/0/5 gigether-options 802.3ad ae10
```

Complete MC-LAG Reference Configuration

```
## SWITCH A Configuration
## System Configuration
system {
    host-name MC-LAG-SWITCH-A;
}

## ICL Configuration
interfaces {
    ae0 {
        description "ICL to SWITCH-B";
        mtu 9216;
        aggregated-ether-options {
            lacp {
                active;
                periodic fast;
            }
        }
        unit 0 {
            family ethernet-switching {
                interface-mode trunk;
                vlan {
                    members all;
                }
            }
```

```
        }
    }

    ## ICL Members
    xe-0/0/0 {
        description "ICL member 1";
        gigether-options {
            802.3ad ae0;
        }
    }
    xe-0/0/1 {
        description "ICL member 2";
        gigether-options {
            802.3ad ae0;
        }
    }

    ## MC-LAG to Server
    ae1 {
        description "MC-LAG to Server-1";
        mtu 9192;
        aggregated-ether-options {
            lacp {
                active;
                periodic fast;
                system-id 00:11:22:33:44:55;
                admin-key 1;
            }
            mc-ae {
                mc-ae-id 1;
                chassis-id 0;              ## Switch A = 0
                mode active-active;
                status-control active;
                init-delay-time 120;
                redundancy-group 1;

                events {
                    iccp-peer-down {
                        prefer-status-control-active;
                    }
                }
            }
        }
        unit 0 {
            family ethernet-switching {
                interface-mode trunk;
                vlan {
                    members [ 100-200 ];
                }
            }
        }
    }

    ## MC-LAG Members
    ge-0/0/10 {
        description "To Server-1 port 1";
        gigether-options {
            802.3ad ae1;
        }
    }

    ## Management
    lo0 {
        unit 0 {
            family inet {
                address 192.168.1.1/32;
            }
        }
    }
}
```

```
## ICCP Configuration
protocols {
    iccp {
        local-ip-addr 192.168.1.1;
        peer 192.168.1.2 {
            session-establishment-hold-time 50;
            backup-liveness-detection {
                backup-peer-ip 10.1.1.2;
            }
            liveness-detection {
                minimum-interval 500;
                multiplier 3;
            }
        }
    }

    ## Routing for ICCP
    ospf {
        area 0.0.0.0 {
            interface lo0.0;
            interface ae0.0;
        }
    }
}

## MC-LAG Service Configuration
switch-options {
    service-id 1;
}

multi-chassis {
    multi-chassis-protection 192.168.1.2 {
        interface ae0;
    }
}

## VLAN Configuration
vlans {
    VLAN100 {
        vlan-id 100;
        mcae-mac-synchronize;
        interface ae1.0;
    }
    VLAN200 {
        vlan-id 200;
        mcae-mac-synchronize;
        interface ae1.0;
    }
}

## SWITCH B - Similar with:
## - chassis-id 1
## - status-control standby
## - local-ip-addr 192.168.1.2
## - peer 192.168.1.1
```

Part 3: Verification & Troubleshooting (The What-If)

Essential Verification Commands

1. Check MC-LAG Status

```
user@switch-a> show interfaces mc-ae
Member Link                  : ae1
  Current State              : Up
  Chassis Id                 : 0
  Redundancy Group           : 1
  MC-AE Mode                 : Active-Active
  Status Control             : Active
```

```
  Revert control            : Enabled
  ICCP Peer State           : Up

  MCAE Configuration:
    MC-AE ID                : 1
    Chassis ID              : 0
    Mode                    : Active-Active
    Status                  : Active

  LACP Statistics:
    LACPDUs Sent            : 45678
    LACPDUs Received        : 45670
```

2. Check ICCP Status

```
user@switch-a> show iccp
Redundancy Group Information:
  RG : 1, Service ID : 1

Peer Information:
  IP: 192.168.1.2, Peer State: Established
  ICL : ae0.0, ICL State: Up

Client Application Information:
  Client: MCSNOOPD
    Sync State: InSync

  Client: L2ALD_MCLAG_INTF
    Sync State: InSync
```

3. Check Virtual Chassis Status

```
user@mx-vc> show virtual-chassis
Virtual Chassis ID: 2c6b.f5eb.9348
Virtual Chassis Mode: Enabled
                                                Mstr          Mixed Route
Member ID  Status   Serial No    Model         prio  Role    Mode  Mode
0 (FPC 0)  Prsnt    ABC123       mx240         255   Master*   N  VC
1 (FPC 1)  Prsnt    XYZ789       mx240         100   Backup    N  VC

Member ID for next new member: 2 (FPC 2)
```

Common Troubleshooting Scenarios

Scenario 1: MC-LAG Not Synchronizing

Symptom: Peers show different MAC tables

Diagnostic Commands:

```
user@switch-a> show ethernet-switching table interface ae1
MAC flags (S - static MAC, D - dynamic MAC, L - locally learned
Ethernet switching table : 10 entries, 10 learned

user@switch-b> show ethernet-switching table interface ae1
Ethernet switching table : 5 entries, 5 learned    <-- Different count!

user@switch-a> show iccp
Client: L2ALD_MCLAG_INTF
  Sync State: OutOfSync    <-- Not synchronized!
```

Cause: VLAN not configured for MAC sync **Solution**:

```
# Enable MAC synchronization on all MC-LAG VLANs
set vlans VLAN100 mcae-mac-synchronize
set vlans VLAN200 mcae-mac-synchronize
commit

# Verify
show multi-chassis mc-lag mac-table
```

Scenario 2: Virtual Chassis Split Brain

Symptom: Both members think they're master

Diagnostic Commands:

```
user@mx1> show virtual-chassis
Member ID  Status   Serial No    Model         prio  Role
0 (FPC 0)  Prsnt    ABC123       mx240         255   Master*

user@mx2> show virtual-chassis
Member ID  Status   Serial No    Model         prio  Role
1 (FPC 1)  Prsnt    XYZ789       mx240         100   Master*   <-- Both master!
```

Cause: VCP links failed **Solution**:

```
# Check VCP status
show virtual-chassis vc-port all-members
show interfaces vcp-255/1/0

# If VCP down, check physical connections
# Then clear VC protocol
request virtual-chassis reactivate member 1

# For persistent issues, disable split detection
set virtual-chassis no-split-detection
commit
```

Scenario 3: MC-LAG Active/Active Imbalance

Symptom: All traffic going through one peer

Diagnostic Commands:

```
user@switch-a> show interfaces ae1 statistics
Output packets: 10000000 (10000000 pps)

user@switch-b> show interfaces ae1 statistics
Output packets: 0 (0 pps)    <-- No traffic!

user@switch-a> show lacp interfaces ae1
LACP state:
  ge-0/0/10  Actor    Yes   Yes   Yes  Yes   Yes   Fast    Active
             Partner  Yes   Yes   Yes  Yes   Yes   Fast    Active
```

Cause: Client hashing all traffic to one peer **Solution**:

```
# Check client load-balancing algorithm
# On switch side, ensure both peers advertise same LACP parameters

# Verify system IDs match
show configuration interfaces ae1 aggregated-ether-options lacp
```

```
# Consider different hashing on client
# Or use MC-LAG load-balancing features
set interfaces ae1 aggregated-ether-options mc-ae events iccp-peer-down force-icl-down
```

Scenario 4: ICL Bandwidth Exhaustion

Symptom: Packet loss during peer failures

Diagnostic Commands:

```
user@switch-a> show interfaces ae0 extensive | match "bps|errors"
  Bit rate: 9.8 Gbps    <-- Near 10G limit!
  Input errors: 0
  Output errors: 12345  <-- Drops!

user@switch-a> monitor interface traffic ae0
```

Cause: ICL undersized for traffic volume **Solution**:

```
# Add more links to ICL
set interfaces xe-0/0/2 gigether-options 802.3ad ae0
set interfaces xe-0/0/3 gigether-options 802.3ad ae0
commit

# Or implement local switching preference
set interfaces ae1 aggregated-ether-options mc-ae events iccp-peer-down prefer-local-switching
```

Scenario 5: Virtual Chassis Routing Issues

Symptom: Routes not properly distributed across members

Diagnostic Commands:

```
user@vc> show route forwarding-table vpn VRF-A member 0
(Shows routes)

user@vc> show route forwarding-table vpn VRF-A member 1
(Missing routes!)

user@vc> show virtual-chassis protocol adjacency
(Check if protocol adjacency is up)
```

Cause: NSR not properly configured **Solution**:

```
# Enable complete NSR/NSB
set routing-options nonstop-routing
set virtual-chassis graceful-restart

# For BGP
set protocols bgp group EXTERNAL type external
set protocols bgp group EXTERNAL graceful-restart

# Ensure commit synchronize
commit synchronize
```

Pro Tips for High Availability

1. **Design ICL Properly**:

   ```
   - Use dedicated high-bandwidth links
   - Make ICL a LAG itself
   ```

```
- Consider dark fiber for distance
- Monitor ICL utilization closely
```

2. **MC-LAG Best Practices**:

```
- Keep configurations identical
- Use configuration groups
- Test failover regularly
- Monitor ICCP status via SNMP
```

3. **Virtual Chassis Deployment**:

```
- Use diverse VCP paths
- Consider 4-member VC for better redundancy
- Plan IP addressing carefully
- Use commit synchronize always
```

4. **Testing Procedures**:

```
# Test MC-LAG failover
request interface mc-ae switchover ae1

# Test VC mastership change
request virtual-chassis routing-engine master switch

# Simulate ICL failure
set interfaces ae0 disable
```

5. **Monitoring Integration**:

```
set event-options policy MC-LAG-MONITOR events iccp_peer_down
set event-options policy MC-LAG-MONITOR then execute-commands commands "show interfaces mc-ae"
```

This module has covered advanced high-availability technologies that eliminate single points of failure at the device level. MC-LAG provides active-active redundancy perfect for server/storage connectivity, while Virtual Chassis simplifies management of multiple devices. These technologies, combined with the ERP and LAG concepts from Module 10, create a comprehensive high-availability architecture suitable for the most demanding service provider environments.

Module 12: Troubleshooting and Monitoring

Part 1: The Conceptual Lecture (The Why)

The Reality of Network Operations

After learning all these switching technologies, here's the truth: Networks break. Not because of poor design or implementation, but because:

- Hardware fails (transceivers, cables, power supplies)
- Software has bugs (memory leaks, process crashes)
- Humans make mistakes (wrong VLAN, duplicate IPs)
- Scale brings complexity (10,000 MACs, 1,000 VLANs)
- External factors interfere (power outages, fiber cuts)

The Key: It's not about preventing all failures - it's about finding and fixing them quickly.

The Troubleshooting Mindset

The OSI Layer Approach

Always start at Layer 1 and work up:

```
Layer 1 (Physical):     Is the cable plugged in?
Layer 2 (Data Link):    Are we learning MACs?
Layer 3 (Network):      Can we ping?
Layer 4+ (Transport+):  Is the application working?
```

The Divide and Conquer Method

Cut the problem space in half repeatedly:

```
Problem: PC A can't reach Server B

Test 1: Can A reach its gateway?
  Yes → Problem is beyond local network
  No → Problem is local

Test 2: Can gateway reach B?
  Yes → Problem is return path
  No → Problem is forward path
```

Common Layer 2 Issues

1. The Silent Killers

These problems cause intermittent issues:

- **Duplex Mismatch**: One side full, other half
- **Speed Mismatch**: Auto-negotiation failures
- **MTU Mismatch**: Jumbo frames partially supported
- **VLAN Mismatch**: Trunk/access misconfiguration

2. The Sudden Failures

These stop traffic immediately:

- **STP Loops**: Broadcast storms
- **VLAN Missing**: Not configured on all switches
- **Port Security**: MAC limit exceeded
- **Link Down**: Physical failure

3. The Performance Degraders

These cause slow performance:

- **Buffer Exhaustion**: Microbursts filling queues
- **MAC Table Full**: Flooding unicast traffic
- **STP Reconvergence**: Topology changes
- **Asymmetric Routing**: Different paths each direction

Systematic Troubleshooting Process

Step 1: Define the Problem

Be specific:

- Bad: "Network is slow"
- Good: "HTTP downloads from VLAN 10 to VLAN 20 achieve only 10Mbps on gigabit links"

Step 2: Gather Information

Before touching anything:

- When did it start?
- What changed?
- Who is affected?

- Is it consistent?

Step 3: Form a Hypothesis

Based on symptoms:

- Broadcast storm → Check for loops
- Specific VLAN affected → Check VLAN config
- After change → Check what was modified

Step 4: Test the Hypothesis

Use non-disruptive tests first:

- Show commands
- Monitor commands
- Packet captures

Step 5: Implement Solution

With rollback plan:

- Document current state
- Make minimal changes
- Test immediately
- Be ready to rollback

The Troubleshooting Toolkit

Information Gathering Tools

1. **Interface Statistics**: Error counters, traffic rates
2. **MAC Tables**: Learning, aging, movement
3. **STP State**: Topology, changes, root bridge
4. **System Logs**: Events, errors, warnings
5. **Protocol Status**: LACP, LLDP, CFM states

Active Testing Tools

1. **Ping**: Basic connectivity
2. **Traceroute**: Path discovery
3. **MAC Ping**: Layer 2 connectivity
4. **Monitor Traffic**: Packet capture
5. **Loopback**: Circuit testing

Advanced Diagnostics

1. **Port Mirroring**: Copy traffic for analysis
2. **Flow Sampling**: Statistical analysis
3. **SNMP Monitoring**: Historical data
4. **Commit History**: Configuration changes
5. **Core Dumps**: Process crash analysis

Building Observable Networks

Good networks are easy to troubleshoot because they:

1. **Log Everything**: Centralized syslog
2. **Monitor Continuously**: SNMP, streaming telemetry
3. **Document Thoroughly**: Topology, addressing, changes
4. **Standardize Consistently**: Naming, configuration templates
5. **Test Regularly**: Automated verification

Part 2: The Junos CLI Masterclass (The How)

Essential Troubleshooting Commands

Layer 1 Verification

```
# Check interface physical status
show interfaces ge-0/0/0 media
show interfaces diagnostics optics ge-0/0/0

# Check error counters
show interfaces ge-0/0/0 extensive | match "error|drop|collision"

# Monitor real-time statistics
monitor interface ge-0/0/0
```

Layer 2 State Examination

```
# MAC address table
show ethernet-switching table
show ethernet-switching table interface ge-0/0/0
show ethernet-switching table vlan VLAN100
show ethernet-switching mac-learning-log

# MAC table summary
show ethernet-switching table summary
show ethernet-switching statistics

# Check VLAN assignment
show vlans
show vlans extensive
show ethernet-switching interface
```

STP Troubleshooting

```
# STP global state
show spanning-tree bridge
show spanning-tree interface
show spanning-tree statistics

# STP events
show log messages | match "STP|RSTP|MSTP"
show spanning-tree interface detail | match "changes|transition"

# MSTP specific
show spanning-tree mstp configuration
show spanning-tree msti 1
```

Advanced Diagnostic Tools

Packet Capture

```
# Basic packet capture
monitor traffic interface ge-0/0/0

# With filters
monitor traffic interface ge-0/0/0 matching "vlan 100"
monitor traffic interface ge-0/0/0 matching "ether host 00:11:22:33:44:55"
monitor traffic interface ge-0/0/0 matching "ether proto 0x8100" write-file vlan.pcap

# Layer 2 headers
monitor traffic interface ge-0/0/0 layer2-headers
monitor traffic interface ge-0/0/0 extensive layer2-headers
```

Port Mirroring Configuration

```
# Local port mirroring
set forwarding-options analyzer MIRROR-SESSION {
```

```
    input {
        ingress {
            interface ge-0/0/0.0;
            interface ge-0/0/1.0;
        }
        egress {
            interface ge-0/0/0.0;
        }
    }
    output {
        interface ge-0/0/10.0;
    }
}

# VLAN mirroring
set forwarding-options analyzer VLAN-MIRROR {
    input {
        ingress {
            vlan VLAN100;
        }
    }
    output {
        interface ge-0/0/10.0;
    }
}
```

Flow Monitoring

```
# Enable sFlow
set protocols sflow {
    agent-id 1.1.1.1;
    collector 10.1.1.100 {
        udp-port 6343;
    }
    interfaces ge-0/0/0.0;
    interfaces ge-0/0/1.0;
    sample-rate 1000;
}

# J-Flow for analysis
set forwarding-options sampling {
    instance SAMPLE-INSTANCE {
        input {
            rate 100;
        }
        family ethernet {
            output {
                flow-server 10.1.1.100 {
                    port 2055;
                    version 9;
                }
            }
        }
    }
}
```

Troubleshooting Automation

Event Scripts

```
# Auto-recovery script for BPDU errors
set event-options event-script file bpdu-recovery.slax {
    python-script-user juniper;
}

set event-options policy BPDU-RECOVERY {
    events l2ald_bpdu_block;
    then {
        event-script bpdu-recovery.slax;
```

```
    }
}

# The script (bpdu-recovery.slax):
version 1.0;
ns junos = "http://xml.juniper.net/junos/*/junos";
ns xnm = "http://xml.juniper.net/xnm/1.1/xnm";
ns jcs = "http://xml.juniper.net/junos/commit-scripts/1.0";

match / {
    <event-script-results> {
        var $interface = event-script-input/junos:interface-name;
        var $clear-cmd = <command> "clear ethernet-switching bpdu-error interface " _ $interface;
        var $results = jcs:invoke($clear-cmd);

        <output> {
            expr "Cleared BPDU error on interface " _ $interface;
        }
    }
}
```

Commit Scripts for Validation

```
# Prevent VLAN misconfigurations
set system scripts commit file vlan-check.slax

# The script:
match configuration {
    for-each (interfaces/interface/unit[family/ethernet-switching/vlan/members]) {
        var $vlan = family/ethernet-switching/vlan/members;
        if (not(vlans[name = $vlan])) {
            <xnm:error> {
                <message> "VLAN " _ $vlan _ " referenced but not defined";
            }
        }
    }
}
```

Complete Troubleshooting Toolkit Configuration

```
## System-wide troubleshooting features
system {
    ## Enhanced logging
    syslog {
        file troubleshoot-log {
            any notice;
            explicit-priority;
            match "(SNMP_TRAP|L2|STP|LACP|interface|error)";
        }
        file interactive-commands {
            interactive-commands any;
        }
    }

    ## Core dumps for process crashes
    core-dumps {
        maximum-files 10;
        maximum-size 512m;
    }

    ## Process monitoring
    processes {
        monitor {
            l2ald {
                low-threshold 10;
                medium-threshold 30;
                high-threshold 50;
            }
        }
```

```
    }
}

## SNMP monitoring
snmp {
    community public {
        authorization read-only;
    }
    trap-group network-monitors {
        targets {
            10.1.1.100;
        }
        categories {
            link;
            vlan;
            stp;
        }
    }
}

## Port mirroring templates
forwarding-options {
    analyzer-template {
        BASIC-MIRROR {
            ratio 1;
            maximum-packet-length 1500;
        }
        SAMPLE-MIRROR {
            ratio 100;
            maximum-packet-length 128;
        }
    }
}

## Event automation
event-options {
    ## Interface flap detection
    policy INTERFACE-FLAP {
        events snmp_trap_link_down;
        within 60 events 5;
        then {
            execute-commands {
                commands {
                    "show interfaces extensive {$interface}";
                    "show log messages | match {$interface} | last 50";
                }
                output-filename interface-flap-diag.log;
                destination diagnose-server;
            }
        }
    }

    ## MAC move detection
    policy MAC-MOVE {
        events l2ald_mac_move;
        attributes-match {
            l2ald_mac_move.mac-count greater-than 100;
        }
        then {
            execute-commands {
                commands {
                    "show ethernet-switching mac-learning-log | last 100";
                    "show ethernet-switching statistics";
                }
            }
            raise-trap;
        }
    }
}

## Telemetry streaming
```

```
services {
    analytics {
        export-profile TELEMETRY {
            local-address 1.1.1.1;
            local-port 30000;
            reporting-rate 30;
            format gpb;
            transport udp;
        }
        sensor interface-stats {
            export-name TELEMETRY;
            resource /junos/system/linecard/interface/;
        }
        sensor mac-stats {
            export-name TELEMETRY;
            resource /junos/system/linecard/ethernet/statistics/;
        }
    }
}
```

Part 3: Verification & Troubleshooting (The What-If)

Real-World Troubleshooting Scenarios

Scenario 1: The Monday Morning Meltdown

Symptom: Entire network segment unreachable after weekend

Initial Investigation:

```
user@switch> show spanning-tree interface
[No output - STP not running!]

user@switch> show interfaces terse | match down
ge-0/0/0.0              up    down eth-switch
ge-0/0/1.0              up    down eth-switch

user@switch> show log messages | match "AM|PM" | last 20
Jul 24 02:00:01 automated-script: Saturday maintenance window started
Jul 24 02:00:15 mgd[1234]: COMMIT COMPLETED
Jul 24 02:00:16 l2ald[2345]: STP DISABLED
```

Root Cause Analysis:

```
user@switch> show configuration | compare rollback 1
[edit protocols]
-  rstp {
-      interface all;
-  }
```

Cause: Automated script removed STP configuration **Solution**:

```
# Immediate fix
configure
rollback 1
commit and-quit

# Long-term fix
set event-options policy PREVENT-STP-DELETE {
    events ui_commit;
    attributes-match {
        ui_commit.message matches ".*delete protocols.*stp.*";
    }
    then {
```

```
        execute-commands {
            commands {
                "configure private";
                "rollback 1";
                "commit and-quit";
            }
        }
        raise-trap;
    }
}
```

Scenario 2: The Intermittent Performance Problem

Symptom: Users report "sometimes slow" file transfers

Investigation Process:

```
# Step 1: Check for patterns
user@switch> show interfaces ge-0/0/0 statistics detail
(Note traffic spikes every 5 minutes)

# Step 2: Correlate with MAC learning
user@switch> show ethernet-switching mac-learning-log | match "ge-0/0/0"
Jul 24 10:00:15 MAC move: 00:11:22:33:44:55 from ge-0/0/1 to ge-0/0/0
Jul 24 10:05:15 MAC move: 00:11:22:33:44:55 from ge-0/0/0 to ge-0/0/1

# Step 3: Check for loops
user@switch> show spanning-tree interface detail | match "BPDU|received"
ge-0/0/0: BPDUs sent 1234, BPDUs received 0    <-- Not receiving!
ge-0/0/1: BPDUs sent 1234, BPDUs received 1230

# Step 4: Monitor traffic
user@switch> monitor traffic interface ge-0/0/0 matching "ether proto 0x8100" extensive
(Shows VLAN tagged broadcast storm)
```

Cause: Unidirectional link causing temporary loops **Solution**:

```
# Enable loop protection
set protocols rstp interface ge-0/0/0 loop-protection
set protocols rstp interface ge-0/0/1 loop-protection
commit

# Enable UDLD (if supported)
set protocols oam ethernet link-fault-management interface ge-0/0/0
set protocols oam ethernet link-fault-management interface ge-0/0/1
```

Scenario 3: The VLAN That Wouldn't Work

Symptom: New VLAN 500 not passing traffic

Systematic Check:

```
# 1. Verify VLAN exists
user@switch> show vlans 500
error: vlan 500 not found

# Wait, it should exist...
user@switch> show configuration vlans | display set | match 500
set vlans CUSTOMER-500 vlan-id 500

# 2. Check exact name
user@switch> show vlans CUSTOMER-500
```

```
Routing instance        VLAN name            Tag     Interfaces
default-switch          CUSTOMER-500         500
                                                    ge-0/0/10.0*

# 3. Only one interface?
user@switch> show configuration interfaces ge-0/0/0 | match 500
vlan members 500;    <-- Numeric reference

# 4. Found the issue!
user@switch> show configuration interfaces ge-0/0/10 | match 500
vlan members CUSTOMER-500;    <-- Name reference
```

Cause: Mixed VLAN reference (name vs ID) **Solution**:

```
# Standardize on VLAN names
configure
delete interfaces ge-0/0/0 unit 0 family ethernet-switching vlan members 500
set interfaces ge-0/0/0 unit 0 family ethernet-switching vlan members CUSTOMER-500
commit

# Add commit script to prevent
set system scripts commit file vlan-consistency.slax
```

Scenario 4: The Mysterious MAC Flapping

Symptom: Syslog flooded with MAC move messages

Advanced Diagnostics:

```
# Check scale
user@switch> show log messages | match "MAC move" | count
Count: 45,234 lines    <-- Massive flapping!

# Identify problematic MACs
user@switch> show log messages | match "MAC move" | last 100 | match "00:11"
Jul 24 11:15:01 MAC move: 00:11:22:33:44:55 from ae0 to ae1
Jul 24 11:15:01 MAC move: 00:11:22:33:44:55 from ae1 to ae0

# Check if it's legitimate redundancy
user@switch> show interfaces ae0 | match "MC-LAG|LACP"
(Not MC-LAG)

user@switch> show ethernet-switching interface ae0
Routing Instance Name    : default-switch
Logical Interface Name   : ae0.0
Interface State          : up
Administrative State     : up
Interface Mode           : TRUNK
Vlan Members             : [100-200]

# Same VLANs on both LAGs - potential loop!
```

Root Cause Discovery:

```
# Trace the MAC
user@switch> show ethernet-switching table interface ae0 | match 00:11:22:33:44:55
VLAN100    00:11:22:33:44:55   D        ae0.0

# Check the other end
user@remote> show lldp neighbors
Local Interface    Chassis Id          Port info
ge-0/0/0          00:aa:bb:cc:dd:ee   ge-1/0/0
```

```
ge-0/0/1          00:aa:bb:cc:dd:ee   ge-1/0/1    <-- Same device!

# Customer has a loop!
```

Solution:

```
# Immediate: Enable storm control
set forwarding-options storm-control-profiles PROTECTION all bandwidth-level 1000
set interfaces ae0 unit 0 forwarding-options storm-control-profile PROTECTION
set interfaces ae1 unit 0 forwarding-options storm-control-profile PROTECTION
commit

# Permanent: BPDU guard on customer-facing
set protocols rstp interface ae0 edge
set protocols rstp interface ae0 bpdu-block-on-edge
set protocols rstp bpdu-block disable-timeout 300
```

Scenario 5: The Graceful Degradation

Symptom: Network works but reported as "sluggish"

Performance Analysis:

```
# Check interface queues
user@switch> show interfaces queue ge-0/0/0
Queue: 0, Forwarding classes: best-effort
  Transmitted:
    Packets              :      123456789
    Bytes                :   158734567890
    Tail-dropped packets :         45678    <-- Drops!

# Check buffer utilization
user@switch> show class-of-service interface ge-0/0/0
  Egress queues: 8 supported, 4 in use
  Queue: 0, Forwarding classes: best-effort
    Transmit rate: 95 percent    <-- Near capacity
    Buffer size: 95 percent       <-- Buffers full!

# Identify traffic pattern
user@switch> monitor interface traffic
Interface    Link  Input packets        Output packets
ge-0/0/0     Up         145678 (850)     8567890 (65432)  <-- Asymmetric!

# Check for microbursts
user@switch> show interfaces ge-0/0/0 extensive | match "Output rate"
  5 second output rate 943215678 bps (125678 pps)

# But average shows different
user@switch> show interfaces ge-0/0/0 statistics detail
  Output bytes  : 234567890123, 12345678 bps    <-- Much lower average
```

Cause: Microbursts overwhelming buffers **Solution**:

```
# Increase buffer allocation
set class-of-service interfaces ge-0/0/0 output-queue 0 bandwidth percent 40
set class-of-service interfaces ge-0/0/0 output-queue 0 buffer-size percent 40

# Enable better congestion management
set class-of-service interfaces ge-0/0/0 output-queue 0 drop-profile-map loss-priority low protocol any drop-profile WRED-
PROFILE

# Configure WRED
```

```
set class-of-service drop-profiles WRED-PROFILE fill-level 50 drop-probability 10
set class-of-service drop-profiles WRED-PROFILE fill-level 75 drop-probability 50
set class-of-service drop-profiles WRED-PROFILE fill-level 95 drop-probability 90
commit
```

Master Troubleshooting Checklist

1. **Physical Layer**:

```
show interfaces media
show interfaces diagnostics optics
show chassis hardware
show chassis alarms
```

2. **Data Link Layer**:

```
show ethernet-switching interface
show ethernet-switching table
show spanning-tree interface
show lacp interfaces
```

3. **Performance Metrics**:

```
show interfaces queue
show interfaces statistics
show system buffers
show system processes extensive
```

4. **Historical Analysis**:

```
show log messages | last 1000
show log user | last 100
show configuration | compare rollback 1
show system commit
```

5. **Active Testing**:

```
ping ethernet mac-address 00:11:22:33:44:55 interface ge-0/0/0
monitor traffic interface ge-0/0/0
traceroute ethernet source interface ge-0/0/0
```

Pro Tips for Expert Troubleshooting

1. **Build a Baseline**:

```
- Normal MAC table size
- Typical interface utilization
- Expected STP topology
- Standard error rates
```

2. **Use Automation**:

```
# Auto-collect diagnostics
set event-options generate-event HOURLY-DIAG time-interval 3600
set event-options policy COLLECT-STATS events HOURLY-DIAG
set event-options policy COLLECT-STATS then execute-commands commands "show ethernet-switching statistics"
```

3. **Correlate Multiple Sources**:

```
- Interface counters + MAC moves = Loop
- CPU spikes + STP changes = Reconvergence
```

```
- Queue drops + no errors = Microbursts
```

4. **Document Everything**:

```
# Add meaningful descriptions
set interfaces ge-0/0/0 description "TO-CORE-SW1:ge-1/0/0:CUST-ABC:VLAN100-200"

# Use apply-groups for standards
set groups TROUBLESHOOTING interfaces <*> description "CIRCUIT:$CIRCUIT:PEER:$PEER"
```

5. **Test Non-Destructively**:

```
- Use 'commit confirmed'
- Mirror don't modify
- Monitor don't flood
- Compare don't assume
```

This final module has equipped you with a comprehensive troubleshooting methodology and practical tools for diagnosing and resolving Layer 2 issues. The combination of systematic approaches, powerful Junos commands, and real-world scenarios prepares you for the challenges of operating large-scale service provider networks. Remember: the best troubleshooters aren't those who never see problems - they're those who can quickly identify, isolate, and resolve issues with minimal impact to services.