Experiments on Virtual Lans and the Spanning Tree Protocol Performed on Enterasys Equipments

Authors:

Federico Monaco
Mauro Pagano
Marco Passariello
Fabrizio Piacenza
Daniele Pintus

fed.monaco@gmail.com
mau.pag@gmail.com
marco.passariello@gmail.com
bris@fastwebnet.it
dani.pintus@gmail.com

Computer Science Dept., Engineering School of the Third University of Rome
## List of Devices Used in the Experiments

<table>
<thead>
<tr>
<th>Laptops</th>
<th>Assigned IP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jason</td>
<td>192.168.1.10</td>
</tr>
<tr>
<td>Danny</td>
<td>192.168.1.11</td>
</tr>
<tr>
<td>Tommy</td>
<td>192.168.1.12</td>
</tr>
<tr>
<td>Jimmy</td>
<td>192.168.1.13</td>
</tr>
<tr>
<td>Freddy</td>
<td>192.168.1.14</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Enterasys Devices</th>
<th>Firmware Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>C2H124-48</td>
<td>04.00.31</td>
</tr>
<tr>
<td>C2G124-24</td>
<td>05.00.59</td>
</tr>
<tr>
<td>B2H154-48P</td>
<td>04.00.22</td>
</tr>
<tr>
<td>N3 7H4382-25(Fast Ethernet)</td>
<td>05.11.29</td>
</tr>
<tr>
<td>N3 7G4282-41(Gb Ethernet)</td>
<td>05.13.06</td>
</tr>
</tbody>
</table>

Switches will be called $C_{48}, C_{24}, B, Nfe, \text{ and } Nge$, respectively
Index of the Experiments

Experiment # 0: configuration and set-up

Experiment # 1: 1 VLAN on 1 Switch

Experiment # 2: 2 VLANs on 1 Switch

Experiment # 3: 3 Asymmetric VLANs on 1 Switch

Experiment # 4: Spanning Tree: Root Bridge Election

Experiment # 5: Spanning Tree: Root Path Cost Variation

Experiment # 6: Spanning Tree: Port Blocking

Experiment # 7: Spanning Tree: Port Blocking and VLANs

Experiment # 8: Trunk IEEE 802.1Q

Experiment # 9: Multiple Spanning Tree: IEEE 802.1s

Experiment # 10: Different Root Bridge for different MSTI

Experiment # 11: Visibility between different STP Regions

Experiment # 12: Root Bridge Election for CST & CIST

Experiment # 13: VLAN & Link Aggregation

Experiment # 14: More VLANs & Link Aggregation

Appendix: Commands Explain
Experiment “0”: preliminaries

• Target:
  – Describe how to setup a laptop to perform the next experiments and how to connect such a laptop to a switch
Experiment “0”: preliminaries

• To connect a laptop to a switch:
  – Administration/Setting: on the serial port of the device (if needed, use an adapter to connect the laptop on a USB port)
  – Setting: UTP cat 5 on default VLAN
Experiment “0”: preliminaries

• Used software (there are other options, of course):
  – Tera Term (Windows)
  – Minicom (Linux)

• Connect the laptop to the specified switch on the COM port

• If enabled on the device it’s possible to use, in both OS, the telnet command:
  \textit{telnet <device_ip_address>} in a shell

• Insert \textit{username} and \textit{password}
Experiment “0”: setting ip address

• In the Enterasys devices, this command is used to setup the ip address:

  • set ip address <ip-address> mask <net-mask>
Experiment 1:
1 VLAN on 1 Switch

• Target:
  – Create a VLAN for two hosts
  – Check the visibility between hosts
Experiment 1:  
1 VLAN on 1 Switch
Experiment 1:
1 VLAN on 1 Switch

• Commands:
  – set vlan create 2
  – set vlan name 2 RED
  – set port vlan fe.1.3 2
  – set port vlan fe.1.4 2
Experiment 1:
1 VLAN on 1 Switch

- Jason (192.168.1.10) on port 3 → RED VLAN
- Freddy (192.168.1.14) on port 4 → RED VLAN

2 hosts on the same VLAN (RED) have no communication problems. Pinging from 192.168.1.10 to 192.168.1.14 returns:

```
Jason@host:~$ ping 192.168.1.14
PING 192.168.1.14 (192.168.1.14) 56(84) bytes of data.
64 bytes from 192.168.1.14: icmp_seq=1 ttl=64 time=2002 ms
64 bytes from 192.168.1.14: icmp_seq=2 ttl=64 time=992 ms
64 bytes from 192.168.1.14: icmp_seq=3 ttl=64 time=0.120 ms
64 bytes from 192.168.1.14: icmp_seq=4 ttl=64 time=0.150 ms
```
Experiment 2: 2 VLANs on 1 Switch

• Target:
  – Create two different VLANs
  – Check the visibility between hosts on different VLANs
Experiment 2:
2 VLANs on 1 Switch

Bridge B:
192.168.1.1

Port N. 3

Port N. 4

red vlan

green vlan

Freddy

Jason
Experiment 2:
2 VLANs on 1 Switch

• Commands:
  – `set vlan create 2`
  – `set vlan create 3`
  – `set vlan name 2 GREEN`
  – `set vlan name 3 RED`
  – `set port vlan fe.1.3 2`
  – `set port vlan fe.1.4 3`
Experiment 2:  
2 VLANs on 1 Switch

- Jason (192.168.1.10) on port 3 → GREEN VLAN
- Freddy (192.168.1.14) on port 4 → RED VLAN
Experiment 3:
3 Asymmetric VLANs on 1 Switch

• Target:
  – Create three different asymmetric VLANs
  – Check the visibility between hosts
Experiment 3: 3 Asymmetric VLANs on 1 Switch

Monaco, Pagano, Passariello, Piacenza, Pintus
Experiments on Virtual Lans and the Spanning Tree Protocol Performed on Enterasys Equipments
Experiment 3:
3 Asymmetric VLANs on 1 Switch

• Commands:
  – set vlan create 2
  – set vlan create 3
  – set vlan create 4
  – set vlan name 2 RED
  – set vlan name 3 GREEN
  – set vlan name 4 BLUE
  – set port vlan fe.1.7 2
  – set port vlan fe.1.8 3
  – set port vlan fe.1.9 4
Experiment 3: 3 Asymmetric VLANs on 1 Switch

• With this configuration, as seen in the previous experiment, each host only sees itself

• So, we need to specify the visibility of each host, setting the ports of the switch
Experiment 3:
3 Asymmetric VLANs on 1 Switch
(inside the switch – port configuration)
Experiment 3:
3 Asymmetric VLANs on 1 Switch
(inside the switch – port configuration)

Now BLUE VLAN sees also GREEN VLAN
Experiment 3: 3 Asymmetric VLANs on 1 Switch (inside the switch – port configuration)

• Commands:
  – `set vlan egress 4 fe.1.8 untagged`
Experiment 3:
3 Asymmetric VLANs on 1 Switch (inside the switch – port configuration)

Monaco, Pagano, Passariello, Piacenza, Pintus
Experiments on Virtual Lans and the Spanning Tree Protocol Performed on Enterasys Equipments
Experiment 3:
3 Asymmetric VLANs on 1 Switch (inside the switch – port configuration)

• Commands:
  – set vlan egress 2 fe.1.8 untagged
Experiment 3: 3 Asymmetric VLANs on 1 Switch (inside the switch – port configuration)

Now GREEN VLAN sees both BLUE and RED VLANs
Experiment 3:
3 Asymmetric VLANs on 1 Switch
(inside the switch – port configuration)

• Commands:
  – `set vlan egress 3 fe.1.7 untagged`
  – `set vlan egress 3 fe.1.9 untagged`
Experiment 3: 3 Asymmetric VLANs on 1 Switch (inside the switch – port configuration)

- In the last diagram we set different **egress** ports to create three different asymmetric VLANs

- We set:
  - Port 7 as egress for GREEN VLAN
  - Port 8 as egress for RED and BLUE VLANs
  - Port 9 as egress for GREEN VLAN
Experiment 3:
3 Asymmetric VLANs on 1 Switch

- The effects are that:
  1. Host Jason (RED) can’t see host Tommy (BLUE)

```
Jason@host:~$ ping 192.168.1.12
PING 192.168.1.12 (192.168.1.12) 56(84) bytes of data.
From 192.168.1.10 icmp_seq=1 Destination Host Unreachable
From 192.168.1.10 icmp_seq=2 Destination Host Unreachable
From 192.168.1.10 icmp_seq=3 Destination Host Unreachable
From 192.168.1.10 icmp_seq=4 Destination Host Unreachable
From 192.168.1.10 icmp_seq=5 Destination Host Unreachable
From 192.168.1.10 icmp_seq=6 Destination Host Unreachable
```
Experiment 3: 3 Asymmetric VLANs on 1 Switch

• The effects are that:
  1. Host Jason (RED) can’t see host Tommy (BLUE)
  2. Host Tommy (BLUE) can’t see host Jason (RED)

```
Tommy@host:~$ ping 192.168.1.10
PING 192.168.1.10 (192.168.1.10) 56(84) bytes of data.
From 192.168.1.12 icmp_seq=1 Destination Host Unreachable
From 192.168.1.12 icmp_seq=2 Destination Host Unreachable
From 192.168.1.12 icmp_seq=3 Destination Host Unreachable
From 192.168.1.12 icmp_seq=5 Destination Host Unreachable
From 192.168.1.12 icmp_seq=6 Destination Host Unreachable
From 192.168.1.12 icmp_seq=7 Destination Host Unreachable
```
Experiment 3:
3 Asymmetric VLANs on 1 Switch

• The effects are that:

1. Host Jason (RED) can’t see host Tommy (BLUE)
2. Host Tommy (BLUE) can’t see host Jason (RED)
3. Host Jason (RED) sees host Danny (GREEN)

```
Jason@host:~$ ping 192.168.1.11
PING 192.168.1.11 (192.168.1.11) 56(84) bytes of data.
64 bytes from 192.168.1.11: icmp_seq=1 ttl=128 time=3.34 ms
64 bytes from 192.168.1.11: icmp_seq=2 ttl=128 time=0.218 ms
64 bytes from 192.168.1.11: icmp_seq=3 ttl=128 time=0.218 ms
64 bytes from 192.168.1.11: icmp_seq=4 ttl=128 time=0.217 ms
64 bytes from 192.168.1.11: icmp_seq=5 ttl=128 time=0.213 ms
64 bytes from 192.168.1.11: icmp_seq=6 ttl=128 time=0.216 ms
```
Experiment 3: 3 Asymmetric VLANs on 1 Switch

• The effects are that:

1. Host Jason (RED) can’t see host Tommy (BLUE)
2. Host Tommy (BLUE) can’t see host Jason (RED)
3. Host Jason (RED) sees host Danny (GREEN)
4. Host Tommy (BLUE) sees host Danny (GREEN)

```
Tommy@host:~$ ping 192.168.1.11
PING 192.168.1.11 (192.168.1.11) 56(84) bytes of data.
64 bytes from 192.168.1.11: icmp_seq=1 ttl=128 time=1.15 ms
64 bytes from 192.168.1.11: icmp_seq=2 ttl=128 time=0.843 ms
64 bytes from 192.168.1.11: icmp_seq=3 ttl=128 time=1.18 ms
64 bytes from 192.168.1.11: icmp_seq=4 ttl=128 time=0.697 ms
64 bytes from 192.168.1.11: icmp_seq=5 ttl=128 time=1.11 ms
64 bytes from 192.168.1.11: icmp_seq=6 ttl=128 time=0.561 ms
```
Experiment 3:
3 Asymmetric VLANs on 1 Switch

• The effects are that:
  1. Host Jason (RED) can’t see host Tommy (BLUE)
  2. Host Tommy (BLUE) can’t see host Jason (RED)
  3. Host Jason (RED) sees host Danny (GREEN)
  4. Host Tommy (BLUE) sees host Danny (GREEN)
  5. Host Danny (GREEN) sees host Tommy (BLUE) and host Jason (RED)

Danny@host:~$ ping 192.168.1.10
PING 192.168.1.10 (192.168.1.10) 56(84) bytes of data.
64 bytes from 192.168.1.10: icmp_seq=1 ttl=128 time=1.15 ms
64 bytes from 192.168.1.10: icmp_seq=2 ttl=128 time=0.843 ms
64 bytes from 192.168.1.10: icmp_seq=3 ttl=128 time=1.18 ms
64 bytes from 192.168.1.10: icmp_seq=4 ttl=128 time=0.697 ms
64 bytes from 192.168.1.10: icmp_seq=5 ttl=128 time=1.11 ms
64 bytes from 192.168.1.10: icmp_seq=6 ttl=128 time=0.561 ms

Danny@host:~$ ping 192.168.1.12
PING 192.168.1.12 (192.168.1.12) 56(84) bytes of data.
64 bytes from 192.168.1.12: icmp_seq=1 ttl=128 time=1.15 ms
64 bytes from 192.168.1.12: icmp_seq=2 ttl=128 time=0.843 ms
64 bytes from 192.168.1.12: icmp_seq=3 ttl=128 time=1.18 ms
64 bytes from 192.168.1.12: icmp_seq=4 ttl=128 time=0.697 ms
64 bytes from 192.168.1.12: icmp_seq=5 ttl=128 time=1.11 ms
64 bytes from 192.168.1.12: icmp_seq=6 ttl=128 time=0.561 ms
Experiment 4: Spanning Tree: Root Bridge Election

• Target:
  – Check the Root Bridge election between two switches with the same priority value, then assign a lower priority to one of them, and test which one becomes the Root Bridge
Experiment 4:
Spanning Tree: Root Bridge Election

Both switches have the same priority value
Experiment 4: Spanning Tree: Root Bridge Election

• By default the enabled STP version is 802.1s: Multiple Spanning Tree Protocol (MSTP)

• For this configuration we prefer to use 802.1d version: Spanning Tree Protocol (STP)
Experiment 4:
Spanning Tree: Root Bridge Election

• Commands:
  – set spantree version stpcompatible
Experiment 4: Spanning Tree: Root Bridge Election

192.168.1.X - Bridge B

<table>
<thead>
<tr>
<th>5 3.984359</th>
<th>Enterasys_1a:19:fc</th>
<th>Spanning-tree.(for-b) STP</th>
<th>Conf. Root = 32768/00:11:88:1a:19:e1 Cost = 0 Port = 0x801b</th>
</tr>
</thead>
<tbody>
<tr>
<td>7 4.087923</td>
<td>Enterasys_1a:19:fc</td>
<td>Spanning-tree.(for-b) STP</td>
<td>Conf. Root = 32768/00:11:88:1a:19:69 Cost = 200000 Port = 0x801b</td>
</tr>
<tr>
<td>10 5.984444</td>
<td>Enterasys_1a:19:fc</td>
<td>Spanning-tree.(for-b) STP</td>
<td>Conf. Root = 32768/00:11:88:1a:19:69 Cost = 200000 Port = 0x801b</td>
</tr>
<tr>
<td>13 7.984522</td>
<td>Enterasys_1a:19:fc</td>
<td>Spanning-tree.(for-b) STP</td>
<td>Conf. Root = 32768/00:11:88:1a:19:69 Cost = 200000 Port = 0x801b</td>
</tr>
<tr>
<td>16 9.984563</td>
<td>Enterasys_1a:19:fc</td>
<td>Spanning-tree.(for-b) STP</td>
<td>Conf. Root = 32768/00:11:88:1a:10:69 Cost = 200000 Port = 0x801b</td>
</tr>
</tbody>
</table>

Spanning Tree Protocol (STP): version 802.1d
Configuration BPDU (C-BPDU): contains information about bridge ID (bridge priority/MAC Address)
Every single switch sends broadcast its C-BPDU, to establish which one is the Root Bridge. The election is based on comparisons between bridge IDs
Experiment 4: Spanning Tree: Root Bridge Election

192.168.1.1 - Bridge B

<table>
<thead>
<tr>
<th>5 3.984359</th>
<th>Enterasys_1a:19:fc</th>
<th>Spanning-tree.(for-bi) STP</th>
<th>Conf. Root = 32768/00:11:88:1a:19:e1 Cost = 0 Port = 0x801b</th>
</tr>
</thead>
<tbody>
<tr>
<td>7 4.087623</td>
<td>Enterasys_1a:19:fc</td>
<td>Spanning-tree.(for-bi) STP</td>
<td>Conf. Root = 32768/00:11:88:1a:19:e9 Cost = 200000 Port = 0x801b</td>
</tr>
<tr>
<td>10 5.984444</td>
<td>Enterasys_1a:19:fc</td>
<td>Spanning-tree.(for-bi) STP</td>
<td>Conf. Root = 32768/00:11:88:1a:19:e9 Cost = 200000 Port = 0x801b</td>
</tr>
<tr>
<td>13 7.984522</td>
<td>Enterasys_1a:19:fc</td>
<td>Spanning-tree.(for-bi) STP</td>
<td>Conf. Root = 32768/00:11:88:1a:19:e9 Cost = 200000 Port = 0x801b</td>
</tr>
<tr>
<td>16 9.984563</td>
<td>Enterasys_1a:19:fc</td>
<td>Spanning-tree.(for-bi) STP</td>
<td>Conf. Root = 32768/00:11:88:1a:19:e9 Cost = 200000 Port = 0x801b</td>
</tr>
</tbody>
</table>

Root Identifier: priority (default 32768)/MAC address
At the beginning every bridge considers itself as the Root Bridge (this is true until another bridge sends it a C-BPDU with a lower bridge ID value)

In the picture: Root Identifier is 32768/00:11:88:1a:19:e1
**Experiment 4:**

**Spanning Tree: Root Bridge Election**

192.168.1.3 - Bridge B

<table>
<thead>
<tr>
<th>12</th>
<th>55.125603</th>
<th>Enterasys la:19:70</th>
<th>Spanning-tree: (for bi STP)</th>
<th>Conf.</th>
<th>Root Bridge: 32768/00:11:88:1a:19:69</th>
<th>Cost = 0</th>
<th>Port = 0x8007</th>
</tr>
</thead>
<tbody>
<tr>
<td>13</td>
<td>55.149873</td>
<td>Enterasys la:19:70</td>
<td>Spanning-tree: (for bi STP)</td>
<td>Conf.</td>
<td>Root Bridge: 32768/00:11:88:1a:19:69</td>
<td>Cost = 0</td>
<td>Port = 0x8007</td>
</tr>
<tr>
<td>15</td>
<td>57.019349</td>
<td>Enterasys la:19:70</td>
<td>Spanning-tree: (for bi STP)</td>
<td>Conf.</td>
<td>Root Bridge: 32768/00:11:88:1a:19:69</td>
<td>Cost = 0</td>
<td>Port = 0x8007</td>
</tr>
<tr>
<td>18</td>
<td>59.019315</td>
<td>Enterasys la:19:70</td>
<td>Spanning-tree: (for bi STP)</td>
<td>Conf.</td>
<td>Root Bridge: 32768/00:11:88:1a:19:69</td>
<td>Cost = 0</td>
<td>Port = 0x8007</td>
</tr>
</tbody>
</table>

**Spanning Tree Protocol**

- Protocol Identifier: Spanning Tree Protocol (0x00CD)
- Protocol Version Identifier: Spanning Tree (0)
- BPD Type: Configuration (0x00)
- BPDU flags: 0x00

Root Identifier: 32768 / 00:11:88:1a:19:69

Root Path Cost: 0

Bridge Identifier: 32768 / 00:11:88:1a:19:69

Port identifier: 0x8007

**In the picture:** Root Identifier is 32768/00:11:88:1a:19:69
Experiment 4: Spanning Tree: Root Bridge Election

192.168.1.3 (32768/00:11:88:1a:19:69) has a lower bridge ID than 192.168.1.1 (32768/00:11:88:1a:19:e1)

32768/00:11:88:1a:19:69 < 32768/00:11:88:1a:19:e1

so 192.168.1.3 becomes the Root Bridge!
Experiment 4: Spanning Tree: Root Bridge Election

Bridge B: 192.168.1.1
Bridge ID: 32768/00:11:88:1a:19:e1

Bridge B: 192.168.1.3
Bridge ID: 32768/00:11:88:1a:19:69

Root Bridge

32768/00:11:88:1a:19:69 < 32768/00:11:88:1a:19:e1

so 192.168.1.3 becomes the Root Bridge!
Experiment 4: Spanning Tree: Root Bridge Election

192.168.1.1 - Bridge B

Monaco, Pagano, Passariello, Piacenza, Pintus
Experiments on Virtual Lans and the Spanning Tree
Protocol Performed on Enterasys Equipments

After the Root Bridge election, 192.168.1.1 stops considering itself as the Root Bridge.

The cost of reaching the Root Bridge through the port (Root Path Cost) is 200.000.
Experiment 4: Spanning Tree: Root Bridge Election

Switches have different priority values

Monaco, Pagano, Passariello, Piacenza, Pintus
Experiments on Virtual Lans and the Spanning Tree Protocol Performed on Enterasys Equipments
Experiment 4: Spanning Tree: Root Bridge Election

• Commands:
  – set spantree priority 0
Experiment 4:
Spanning Tree: Root Bridge Election

• We are able to lead the election, changing the bridge priority value

  – 192.168.1.1 $\rightarrow$ new priority: 0
  – 192.168.1.3 $\rightarrow$ default priority: 32768
Experiment 4:
Spanning Tree: Root Bridge Election

Bridge B: 192.168.1.1
Bridge ID: 0/00:11:88:1a:19:e1

Bridge B: 192.168.1.3
Bridge ID: 32768/00:11:88:1a:19:69

Root Bridge

192.168.1.1 \to priority value: 0
192.168.1.3 \to priority value: 32768

192.168.1.1 becomes the new Root Bridge
### Experiment 4: Spanning Tree: Root Bridge Election

**192.168.1.1 - Bridge B**

<table>
<thead>
<tr>
<th>36</th>
<th>49.912364</th>
<th>Enterasy_1a:19:fc</th>
<th>Spanning-tree-(for-br STP)</th>
<th>Conf. Root = 0/00:11:88:1a:19:e1</th>
<th>Cost = 0</th>
<th>Port = Ox801b</th>
</tr>
</thead>
<tbody>
<tr>
<td>37</td>
<td>51.912463</td>
<td>Enterasy_1a:19:fc</td>
<td>Spanning-tree-(for-br STP)</td>
<td>Conf. Root = 0/00:11:88:1a:19:e1</td>
<td>Cost = 0</td>
<td>Port = Ox801b</td>
</tr>
<tr>
<td>38</td>
<td>53.912505</td>
<td>Enterasy_1a:19:fc</td>
<td>Spanning-tree-(for-br STP)</td>
<td>Conf. Root = 0/00:11:88:1a:19:e1</td>
<td>Cost = 0</td>
<td>Port = Ox801b</td>
</tr>
<tr>
<td>39</td>
<td>55.912427</td>
<td>Enterasy_1a:19:fc</td>
<td>Spanning-tree-(for-br STP)</td>
<td>Conf. Root = 0/00:11:88:1a:19:e1</td>
<td>Cost = 0</td>
<td>Port = Ox801b</td>
</tr>
<tr>
<td>40</td>
<td>57.912493</td>
<td>Enterasy_1a:19:fc</td>
<td>Spanning-tree-(for-br STP)</td>
<td>Conf. Root = 0/00:11:88:1a:19:e1</td>
<td>Cost = 0</td>
<td>Port = Ox801b</td>
</tr>
<tr>
<td>41</td>
<td>59.912552</td>
<td>Enterasy_1a:19:fc</td>
<td>Spanning-tree-(for-br STP)</td>
<td>Conf. Root = 0/00:11:88:1a:19:e1</td>
<td>Cost = 0</td>
<td>Port = Ox801b</td>
</tr>
<tr>
<td>42</td>
<td>59.913546</td>
<td>Enterasy_1a:19:fc</td>
<td>Spanning-tree-(for-br STP)</td>
<td>Conf. Root = 0/00:11:88:1a:19:e1</td>
<td>Cost = 0</td>
<td>Port = Ox801b</td>
</tr>
<tr>
<td>43</td>
<td>61.912641</td>
<td>Enterasy_1a:19:fc</td>
<td>Spanning-tree-(for-br STP)</td>
<td>Conf. Root = 0/00:11:88:1a:19:e1</td>
<td>Cost = 0</td>
<td>Port = Ox801b</td>
</tr>
</tbody>
</table>

**Spanning Tree Protocol**

- Protocol Identifier: Spanning Tree Protocol (0x8000)
- Protocol Version Identifier: Spanning Tree (0)
- BPDU Type: Configuration (0x00)
- **BPDU flags:** 0x00
  - Root Identifier: 0 / 00:11:88:1a:19:e1
  - Root Path Cost: 0
  - Bridge Identifier: 0 / 00:11:88:1a:19:e1
  - Port identifier: 0x801b

After changing the priority value, **192.168.1.1 becomes the new Root Bridge!**
Experiment 5: Spanning Tree: Root Path Cost Variation

• Target:
  – Test how the Root Path Cost changes after a topology variation
Experiment 5:
Spanning Tree: Root Path Cost Variation

Monaco, Pagano, Passariello, Piacenza, Pintus
Experiments on Virtual Lans and the Spanning Tree Protocol Performed on Enterasys Equipments
After the election, **192.168.1.2** becomes the Root Bridge (Root Path Cost = 0)
Experiment 5: Spanning Tree: Root Path Cost Variation

192.168.1.1 – Bridge C48

<table>
<thead>
<tr>
<th>IP Address</th>
<th>MAC Address</th>
<th>Protocol</th>
<th>Configuration</th>
<th>Root Path Cost</th>
<th>Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.168.1.1</td>
<td>192.168.1.2</td>
<td>IPv4</td>
<td>Bridge C48</td>
<td>200.000</td>
<td>0x801c</td>
</tr>
</tbody>
</table>

**Spanning Tree Protocol**

- Protocol Identifier: Spanning Tree Protocol (0x0000)
- Protocol Version Identifier: Spanning Tree (0)
- Bridge Identifier: Configuration (0x00)

**BPDU flags: 0x00**

- Root Identifier: 32768 / 00:01:f4:5c:3f:20
- Root Path Cost: 200000
- Bridge Identifier: 32768 / 00:11:88:45:5b:00
- Port Identifier: 0x801c

**192.168.1.1:** Root Path Cost = 200.000 (to reach 192.168.1.2)
Experiment 5:
Spanning Tree: Root Path Cost Variation

192.168.1.3 – Bridge B

<table>
<thead>
<tr>
<th>IP Address</th>
<th>Mac Address</th>
<th>Protocol</th>
<th>Configuration</th>
<th>Root Path Cost</th>
<th>Port Identifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>90.135.896076</td>
<td>Enterasys_1a:19:83</td>
<td>Spanning-tree-(for-br STP)</td>
<td>Conf. Root = 32768/00:01:f4:5c:3f:20</td>
<td>Cost = 200000</td>
<td>Port = 0x801a</td>
</tr>
<tr>
<td>91.137.8966156</td>
<td>Enterasys_1a:19:83</td>
<td>Spanning-tree-(for-br STP)</td>
<td>Conf. Root = 32768/00:01:f4:5c:3f:20</td>
<td>Cost = 200000</td>
<td>Port = 0x801a</td>
</tr>
<tr>
<td>92.139.896212</td>
<td>Enterasys_1a:19:83</td>
<td>Spanning-tree-(for-br STP)</td>
<td>Conf. Root = 32768/00:01:f4:5c:3f:20</td>
<td>Cost = 200000</td>
<td>Port = 0x801a</td>
</tr>
</tbody>
</table>

**Spanning Tree Protocol**
- Protocol Identifier: Spanning Tree Protocol (0x0000)
- Protocol Version Identifier: Spanning Tree (0)
- BPCU Type: Configuration (0x00)

**BPCU flags: 0x00**
- Root Identifier: 32768 / 00:01:f4:5c:3f:20
- Root Path Cost: 200000
- Bridge Identifier: 32768 / 00:11:88:1a:19:69
- Port Identifier: 0x801a

192.168.1.3: Root Path Cost = 200.000 (to reach 192.168.1.2)
Experiment 5: Spanning Tree: Root Path Cost Variation

• When topology changes (have a look at the next slide), lacking of a direct link to the Root Bridge, 192.168.1.3 now sends its packets through 192.168.1.1 to reach 192.168.1.2

• While acrossing more ports the Root Path Cost increases
Experiment 5: Spanning Tree: Root Path Cost Variation

Freddy

Jason

Bridge B:
192.168.1.3

Bridge C24:
192.168.1.2

Bridge C48:
192.168.1.1

Bridge ID:
32768/00:11:88:45:5b:80

Root Bridge

Bridge ID:
32768/00:11:88:1a:19:69

Bridge ID:
32768/00:01:f4:5c:3f:20

Freddy

Tommy

Jason

Tommy, Monaco, Pagano, Passariello, Piacenza, Pintus
Experiments on Virtual Lans and the Spanning Tree Protocol Performed on Enterasys Equipments
Experiment 5:
Spanning Tree: Root Path Cost Variation

192.168.1.3 – Bridge B

<table>
<thead>
<tr>
<th>IP Address</th>
<th>Bridge Identifier</th>
<th>Spanning Tree Protocol</th>
<th>Configuration Root</th>
<th>Port Cost</th>
<th>Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.168.1.3</td>
<td>0x801a</td>
<td>Spanning tree (for br STP)</td>
<td>32768/00:01:f4:5c:3f:20</td>
<td>400000</td>
<td>0x801a</td>
</tr>
<tr>
<td>192.168.1.3</td>
<td>0x801a</td>
<td>Spanning tree (for br STP)</td>
<td>32768/00:01:f4:5c:3f:20</td>
<td>400000</td>
<td>0x801a</td>
</tr>
</tbody>
</table>

Protocol Identifier: Spanning Tree Protocol (0x0000)
Protocol Version Identifier: Spanning Tree (0)
BPDU Type: Configuration (0x00)
BPDU flags: 0x00

Root Identifier: 32768 / 00:01:f4:5c:3f:20
Root Path Cost: 400000
Bridge Identifier: 32768 / 00:11:88:1a:19:69
Port Identifier: 0x801a

After link failure, the cost to reach the Root Bridge from 192.168.1.3 increases up to 400.000
Experiment 6: Spanning Tree: Port Blocking

• Target:
  – Check how Spanning Tree Protocol avoids formation of cycles in the network, blocking redundant links
Experiment 6:
Spanning Tree: Port Blocking

Bridge B: 192.168.1.1
MAC Address: 00:11:88:1a:19:e1
Port 2

Bridge B: 192.168.1.2
MAC Address: 00:11:88:1a:19:69
Port 2

STP 802.1d

Jason

Jimmy

Monaco, Pagano, Passariello, Piacenza, Pintus
Experiments on Virtual Lans and the Spanning Tree Protocol Performed on Enterasys Equipments
Experiment 6:
Spanning Tree: Port Blocking

• Commands:
  – `set lacp disable`
  – `set spantree version stpcompatible`
Experiment 6: Spanning Tree: Port Blocking

Bridge B:
MAC Address: 192.168.1.1

Port 2

<table>
<thead>
<tr>
<th>SID</th>
<th>Port</th>
<th>State</th>
<th>Role</th>
<th>Cost</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>fe.1.2</td>
<td>Forwarding</td>
<td>Root</td>
<td>200000</td>
<td>128</td>
</tr>
</tbody>
</table>

Spanning tree status: enabled
Spanning tree instance: 0
Designated Root MacAddr: 00:11:88:1A:19:69
Designated Root Port: fe.1.2
Designated Root Priority: 32768
Designated Root Cost: 200000
Root Max Age: 20
Root Hello Time: 2
Root Forward Delay: 15
Bridge ID MAC Address: 00:11:88:1A:19:E1
Bridge ID Priority: 32768
Bridge Max Age: 20
Bridge Hello Time: 2
Bridge Forward Delay: 15
Topology Change Count: 7
Time Since Top Change: 0 days 0:4:9
Max Hops: 20
Experiment 6: Spanning Tree: Port Blocking

Spanning tree status: enabled
Spanning tree instance: 0
Designated Root MacAddr: 00:11:88:1A:19:69
Designated Root Port: 0
Designated Root Priority: 32768
Designated Root Cost: 0
Root Max Age: 20
Root Hello Time: 2
Root Forward Delay: 15
Bridge ID MAC Address: 00:11:88:1A:19:69
Bridge ID Priority: 32768
Bridge Max Age: 20
Bridge Hello Time: 2
Bridge Forward Delay: 15
Topology Change Count: 6
Time Since Top Change: 0 days 0:2:25
Max Hops: 20

<table>
<thead>
<tr>
<th>SID</th>
<th>Port</th>
<th>State</th>
<th>Role</th>
<th>Cost</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>fe.1.2</td>
<td>Forwarding</td>
<td>Designated</td>
<td>200000</td>
<td>128</td>
</tr>
</tbody>
</table>

Bridge B: 192.168.1.2
MAC Address: 00:11:88:1A:19:69
Experiment 6:
Spanning Tree: Port Blocking

• No ports need to be blocked if using only one link
Experiment 6: Spanning Tree: Port Blocking

Jason@host:~$ ping 192.168.1.13
PING 192.168.1.13 (192.168.1.13) 56(84) bytes of data.
64 bytes from 192.168.1.13: icmp_seq=1 ttl=64 time=0.672 ms
64 bytes from 192.168.1.13: icmp_seq=2 ttl=64 time=0.158 ms
64 bytes from 192.168.1.13: icmp_seq=3 ttl=64 time=0.165 ms
64 bytes from 192.168.1.13: icmp_seq=4 ttl=64 time=0.163 ms
64 bytes from 192.168.1.13: icmp_seq=5 ttl=64 time=0.167 ms
64 bytes from 192.168.1.13: icmp_seq=6 ttl=64 time=0.162 ms

Jimmy@host:~$ ping 192.168.1.10
PING 192.168.1.10 (192.168.1.10) 56(84) bytes of data.
64 bytes from 192.168.1.10: icmp_seq=1 ttl=64 time=0.166 ms
64 bytes from 192.168.1.10: icmp_seq=2 ttl=64 time=0.156 ms
64 bytes from 192.168.1.10: icmp_seq=3 ttl=64 time=0.163 ms
64 bytes from 192.168.1.10: icmp_seq=4 ttl=64 time=0.164 ms
64 bytes from 192.168.1.10: icmp_seq=5 ttl=64 time=0.170 ms
64 bytes from 192.168.1.10: icmp_seq=6 ttl=64 time=0.155 ms
Experiment 6: Spanning Tree: Port Blocking

Bridge B: 192.168.1.1
MAC Address: 00:11:88:1a:19:e1
Port 2
Port 5

Bridge B: 192.168.1.2
MAC Address: 00:11:88:1a:19:69
Port 2
Port 5

STP 802.1d

Jason

Jimmy
Experiment 6:
Spanning Tree: Port Blocking

• Adding a link we create a cycle

• To avoid this problem, STP will block one or more ports of the devices
Experiment 6:
Spanning Tree: Port Blocking

Bridge B: 192.168.1.1
MAC Address: 00:11:88:1a:19:e1
Port 5
Port 2

Bridge B: 192.168.1.2
MAC Address: 00:11:88:1a:19:69
Port 2
Port 5

STP 802.1d

Jason

Jimmy

STP 802.1d
Experiment 6: Spanning Tree: Port Blocking

Bridge B:
MAC Address: 192.168.1.1

Port 2
Port 5

MAC Address: 00:11:88:1a:19:e1

Spanning tree status - enabled
Spanning tree instance - 0
Designated Root MacAddr - 00:11:88:1A:19:69
Designated Root Port - fe.1.2
Designated Root Priority - 32768
Designated Root Cost - 200000
Root Max Age - 20
Root Hello Time - 2
Root Forward Delay - 15
Bridge ID MAC Address - 00:11:88:1A:19:E1
Bridge ID Priority - 32768
Bridge Max Age - 20
Bridge Hello Time - 2
Bridge Forward Delay - 15
Topology Change Count - 7
Time Since Top Change - 0 days 0:7:30
Max Hops - 20

<table>
<thead>
<tr>
<th>SID</th>
<th>Port</th>
<th>State</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>fe.1.2</td>
<td>Forwarding</td>
<td>Root</td>
</tr>
<tr>
<td>0</td>
<td>fe.1.5</td>
<td>Discarding</td>
<td>Alternate</td>
</tr>
</tbody>
</table>

Cost: 200000
Priority: 128
Experiment 6: Spanning Tree: Port Blocking

Spanning tree status: enabled
Spanning tree instance: 0
Designated Root MacAddr: 00:11:88:1A:19:69
Designated Root Port: 0
Designated Root Priority: 32768
Designated Root Cost: 0
Root Max Age: 20
Root Hello Time: 2
Root Forward Delay: 15
Bridge ID MAC Address: 00:11:88:1A:19:69
Bridge ID Priority: 32768
Bridge Max Age: 20
Bridge Hello Time: 2
Bridge Forward Delay: 15
Topology Change Count: 9
Time Since Top Change: 0 days 0:0:1
Max Hops: 20

<table>
<thead>
<tr>
<th>SID</th>
<th>Port</th>
<th>State</th>
<th>Role</th>
<th>Cost</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>fe.1.2</td>
<td>Forwarding</td>
<td>Designated</td>
<td>200000</td>
<td>128</td>
</tr>
<tr>
<td>0</td>
<td>fe.1.5</td>
<td>Forwarding</td>
<td>Designated</td>
<td>200000</td>
<td>128</td>
</tr>
</tbody>
</table>
Experiment 6: Spanning Tree: Port Blocking

- Root Bridge: does not have blocked ports because they are all designated

- “Non-Root” Bridge: 802.1d blocks port 5 because its ID value is higher than port 2’s one
Experiment 6: Spanning Tree: Port Blocking

Jason@host:~$ ping 192.168.1.13
PING 192.168.1.13 (192.168.1.13) 56(84) bytes of data.
64 bytes from 192.168.1.13: icmp_seq=1 ttl=64 time=0.168 ms
64 bytes from 192.168.1.13: icmp_seq=2 ttl=64 time=0.165 ms
64 bytes from 192.168.1.13: icmp_seq=3 ttl=64 time=0.162 ms
64 bytes from 192.168.1.13: icmp_seq=4 ttl=64 time=0.168 ms
64 bytes from 192.168.1.13: icmp_seq=5 ttl=64 time=0.157 ms
64 bytes from 192.168.1.13: icmp_seq=6 ttl=64 time=0.164 ms

Jimmy@host:~$ ping 192.168.1.10
PING 192.168.1.10 (192.168.1.10) 56(84) bytes of data.
64 bytes from 192.168.1.10: icmp_seq=1 ttl=64 time=0.169 ms
64 bytes from 192.168.1.10: icmp_seq=2 ttl=64 time=0.153 ms
64 bytes from 192.168.1.10: icmp_seq=3 ttl=64 time=0.165 ms
64 bytes from 192.168.1.10: icmp_seq=4 ttl=64 time=0.162 ms
64 bytes from 192.168.1.10: icmp_seq=5 ttl=64 time=0.162 ms
64 bytes from 192.168.1.10: icmp_seq=6 ttl=64 time=0.159 ms
Experiment 7: Spanning Tree: Port Blocking and VLANs

• Target:
  – Check how Spanning Tree Protocol avoids formation of cycles in the network, blocking redundant links, in presence of VLANs too
Experiment 7: Spanning Tree: Port Blocking and VLANs
Experiment 7:
Spanning Tree: Port Blocking and VLANs

• Commands:

  – set lacp disable
  – set spantree version stpcompatible
  – set vlan create 2
  – set vlan create 3
  – set vlan name 2 GREEN
  – set vlan name 3 RED
  – set port vlan fe.1.2 2
  – set port vlan fe.1.5 2
  – set port vlan fe.1.13 2
  – set port vlan fe.1.6 3
  – set port vlan fe.1.11 3
Experiment 7:
Spanning Tree: Port Blocking and VLANs

• With 3 links we create cycles:
  
  – Root Bridge: does not have blocked ports because they are all designated
  
  – “Non-Root” Bridge: 802.1d doesn’t allow multiple instances of Spanning Tree, so port 5 and port 6 are blocked because their ID values are higher than port 2’s one, even though they belong to different VLANs
Experiment 7: Spanning Tree: Port Blocking and VLANs

Monaco, Pagano, Passariello, Piacenza, Pintus
Experiments on Virtual Lans and the Spanning Tree Protocol Performed on Enterasys Equipments

<table>
<thead>
<tr>
<th>SID</th>
<th>Port</th>
<th>State</th>
<th>Role</th>
<th>Cost</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>fe.1.2</td>
<td>Forwarding</td>
<td>Root</td>
<td>200000</td>
<td>128</td>
</tr>
<tr>
<td>0</td>
<td>fe.1.5</td>
<td>Discarding</td>
<td>Alternate</td>
<td>200000</td>
<td>128</td>
</tr>
<tr>
<td>0</td>
<td>fe.1.6</td>
<td>Discarding</td>
<td>Alternate</td>
<td>200000</td>
<td>128</td>
</tr>
</tbody>
</table>
Experiment 7: Spanning Tree: Port Blocking and VLANs

- In this case we have:
  - GREEN VLAN: 1 enabled port (port 2) and 1 blocked port (port 5)
  - RED VLAN: 1 blocked port (port 6)
Experiment 7: Spanning Tree: Port Blocking and VLANs

<table>
<thead>
<tr>
<th>Spanning tree status</th>
<th>- enabled</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spanning tree instance</td>
<td>- 0</td>
</tr>
<tr>
<td>Designated Root MacAddr</td>
<td>- 00:11:00:1A:19:69</td>
</tr>
<tr>
<td>Designated Root Port</td>
<td>- 0</td>
</tr>
<tr>
<td>Designated Root Priority</td>
<td>- 32768</td>
</tr>
<tr>
<td>Designated Root Cost</td>
<td>- 6</td>
</tr>
<tr>
<td>Root Max Age</td>
<td>- 20</td>
</tr>
<tr>
<td>Root Hello Time</td>
<td>- 2</td>
</tr>
<tr>
<td>Root Forward Delay</td>
<td>- 15</td>
</tr>
<tr>
<td>Bridge ID MAC Address</td>
<td>- 00:11:00:1A:19:69</td>
</tr>
<tr>
<td>Bridge ID Priority</td>
<td>- 32768</td>
</tr>
<tr>
<td>Bridge Max Age</td>
<td>- 20</td>
</tr>
<tr>
<td>Bridge Hello Time</td>
<td>- 2</td>
</tr>
<tr>
<td>Bridge Forward Delay</td>
<td>- 15</td>
</tr>
<tr>
<td>Topology Change Count</td>
<td>- 6</td>
</tr>
<tr>
<td>Time Since Top Change</td>
<td>- 0 days 0:0:2</td>
</tr>
<tr>
<td>Max Hops</td>
<td>- 20</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SID</th>
<th>Port</th>
<th>State</th>
<th>Role</th>
<th>Cost</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>fe.1.2</td>
<td>Forwarding</td>
<td>Designated</td>
<td>2000000</td>
<td>128</td>
</tr>
<tr>
<td>0</td>
<td>fe.1.5</td>
<td>Forwarding</td>
<td>Designated</td>
<td>2000000</td>
<td>128</td>
</tr>
<tr>
<td>0</td>
<td>fe.1.6</td>
<td>Forwarding</td>
<td>Designated</td>
<td>2000000</td>
<td>128</td>
</tr>
</tbody>
</table>
Experiment 7:
Spanning Tree: Port Blocking and VLANs

- Hosts on GREEN VLAN are reachable

Jason@host:~$ ping 192.168.1.13
PING 192.168.1.13 (192.168.1.13) 56(84) bytes of data.
64 bytes from 192.168.1.13: icmp_seq=1 ttl=64 time=0.165 ms
64 bytes from 192.168.1.13: icmp_seq=2 ttl=64 time=0.188 ms
64 bytes from 192.168.1.13: icmp_seq=3 ttl=64 time=0.167 ms
64 bytes from 192.168.1.13: icmp_seq=4 ttl=64 time=0.165 ms
64 bytes from 192.168.1.13: icmp_seq=5 ttl=64 time=0.188 ms
64 bytes from 192.168.1.13: icmp_seq=6 ttl=64 time=0.165 ms

Jimmy@host:~$ ping 192.168.1.10
PING 192.168.1.10 (192.168.1.10) 56(84) bytes of data.
64 bytes from 192.168.1.10: icmp_seq=1 ttl=64 time=0.148 ms
64 bytes from 192.168.1.10: icmp_seq=2 ttl=64 time=0.148 ms
64 bytes from 192.168.1.10: icmp_seq=3 ttl=64 time=0.159 ms
64 bytes from 192.168.1.10: icmp_seq=4 ttl=64 time=0.160 ms
64 bytes from 192.168.1.10: icmp_seq=5 ttl=64 time=0.162 ms
64 bytes from 192.168.1.10: icmp_seq=6 ttl=64 time=0.158 ms
Experiment 7: Spanning Tree: Port Blocking and VLANs

• Hosts on RED VLAN are unreachable
Experiment 8: Trunk IEEE 802.1Q

• Target:
  – Create a Trunk 1Q, carrying two VLANs between two switches
Experiment 8: Trunk IEEE 802.1Q
Experiment 8: Trunk IEEE 802.1Q

• Commands:
  – `set lacp disable`
  – `set spantree version stpcompatible`
  – `set vlan create 2`
  – `set vlan create 3`
  – `set vlan name 2 GREEN`
  – `set vlan name 3 RED`
  – `set port vlan fe.1.5 2`
  – `set port vlan fe.1.20 2`
  – `set port vlan fe.1.6 3`
  – `set port vlan fe.1.30 3`
Experiment 8: Trunk IEEE 802.1Q

- As seen in previous experiments, 802.1d will block one or more ports on “Non-Root” devices
Experiment 8: 
Trunk IEEE 802.1Q

B2(su)->show spantree stats active
Spanning tree status - enabled
Spanning tree instance - 0
Designated Root MacAddr - 00:11:88:1A:19:69
Designated Root Port - 0
Designated Root Priority - 32768
Designated Root Cost - 0
Root Max Age - 20
Root Hello Time - 2
Root Forward Delay - 15
Bridge ID MAC Address - 00:11:88:1A:19:69
Bridge ID Priority - 32768
Bridge Max Age - 20
Bridge Hello Time - 2
Bridge Forward Delay - 15
Topology Change Count - 12
Time Since Top Change - 0 days 0:0:16
Max Hops - 20

<table>
<thead>
<tr>
<th>SID</th>
<th>Port</th>
<th>State</th>
<th>Role</th>
<th>Cost</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>fe.1.20</td>
<td>Forwarding</td>
<td>Designated</td>
<td>200000</td>
<td>128</td>
</tr>
<tr>
<td>0</td>
<td>fe.1.30</td>
<td>Forwarding</td>
<td>Designated</td>
<td>200000</td>
<td>128</td>
</tr>
</tbody>
</table>

Monaco, Pagano, Passariello, Piacenza, Pintus
Experiments on Virtual Lans and the Spanning Tree Protocol Performed on Enterasys Equipments
Experiment 8: Trunk IEEE 802.1Q

B2(su)->show spantree stats active
Spanning tree status - enabled
Spanning tree instance - 0
Designated Root MacAddr - 00:11:88:1A:19:69
Designated Root Port - fe.1.20
Designated Root Priority - 32768
Designated Root Cost - 200000
Root Max Age - 20
Root Hello Time - 2
Root Forward Delay - 30
Bridge ID MAC Address - 00:11:88:1A:19:E1
Bridge ID Priority - 32768
Bridge Max Age - 20
Bridge Hello Time - 2
Bridge Forward Delay - 15
Topology Change Count - 20
Time Since Top Change - 0 days 0:0:15
Max Hops - 20

<table>
<thead>
<tr>
<th>SID</th>
<th>Port</th>
<th>State</th>
<th>Role</th>
<th>Cost</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>fe.1.20</td>
<td>Forwarding</td>
<td>Root</td>
<td>200000</td>
<td>128</td>
</tr>
<tr>
<td>0</td>
<td>fe.1.30</td>
<td>Discarding</td>
<td>Alternate</td>
<td>200000</td>
<td>128</td>
</tr>
</tbody>
</table>
• We are able to communicate on GREEN VLAN because port 20 is NOT blocked

• Blocked port 30, instead, makes communicating on RED VLAN impossible
Experiment 8:
Trunk IEEE 802.1Q

Danny@host:$ ping 192.168.1.13
PING 192.168.1.13 (192.168.1.13) 56(84) bytes of data
64 bytes from 192.168.1.13: icmp_seq=1 ttl=64 time=3.21 ms
64 bytes from 192.168.1.13: icmp_seq=2 ttl=64 time=0.186 ms
64 bytes from 192.168.1.13: icmp_seq=3 ttl=64 time=0.176 ms
64 bytes from 192.168.1.13: icmp_seq=4 ttl=64 time=0.197 ms
64 bytes from 192.168.1.13: icmp_seq=5 ttl=64 time=0.182 ms

Jason@host:$ ping 192.168.1.14
PING 192.168.1.14 (192.168.1.14) 56(84) bytes of data.
From 192.168.1.10 icmp_seq=1 Destination Host Unreachable
From 192.168.1.10 icmp_seq=2 Destination Host Unreachable
From 192.168.1.10 icmp_seq=3 Destination Host Unreachable
From 192.168.1.10 icmp_seq=5 Destination Host Unreachable
From 192.168.1.10 icmp_seq=6 Destination Host Unreachable
Experiment 8: Trunk IEEE 802.1Q

- **trunk 1Q link creation**: we setup every switch on port 25 to specify that the outgoing packets have to be tagged (according to their own VLANs), and the incoming untagged packets must be discarded.

- **With trunk 1Q link we can communicate on GREEN VLAN and RED VLAN**
Experiment 8: Trunk IEEE 802.1Q

Bridge B:
192.168.1.1

Bridge B:
192.168.1.2

Port 25

Port 25

Port 5

Port 6

Trunk 1Q

STP 802.1d with TRUNK 1Q

Danny

green vlan

Jason

red vlan

Freddy

red vlan

Jimmy

green vlan

Monaco, Pagano, Passariello, Piacenza, Pintus
Experiments on Virtual Lans and the Spanning Tree Protocol Performed on Enterasys Equipments
Experiment 8: Trunk IEEE 802.1Q

• Commands:
  – `set port ingress-filter fe.1.25`
  – `set vlan egress 2 fe.1.25 tagged`
  – `set vlan egress 3 fe.1.25 tagged`
  – `set port discard fe.1.25 untagged`
Experiment 8: Trunk IEEE 802.1Q

Jason@host:~$ ping 192.168.1.14
PING 192.168.1.14 (192.168.1.14) 56(84) bytes of data.
64 bytes from 192.168.1.14: icmp_seq=1 ttl=64 time=0.186 ms
64 bytes from 192.168.1.14: icmp_seq=2 ttl=64 time=0.173 ms
64 bytes from 192.168.1.14: icmp_seq=3 ttl=64 time=0.175 ms
64 bytes from 192.168.1.14: icmp_seq=4 ttl=64 time=0.176 ms
64 bytes from 192.168.1.14: icmp_seq=5 ttl=64 time=0.171 ms

Danny@host:~$ ping 192.168.1.13
PING 192.168.1.13 (192.168.1.13) 56(84) bytes of data.
64 bytes from 192.168.1.13: icmp_seq=19 ttl=64 time=990 ms
64 bytes from 192.168.1.13: icmp_seq=20 ttl=64 time=0.165 ms
64 bytes from 192.168.1.13: icmp_seq=21 ttl=64 time=0.167 ms
64 bytes from 192.168.1.13: icmp_seq=22 ttl=64 time=0.163 ms
64 bytes from 192.168.1.13: icmp_seq=23 ttl=64 time=0.154 ms
Experiment 9:
Multiple Spanning Tree: IEEE 802.1s

• Target:
  – Create two Spanning Tree instances bound to different VLANs
Experiment 9:  
Multiple Spanning Tree: IEEE 802.1s

• 802.1s Multiple Spanning Tree Protocol (MSTP) specifies the possibility to create different Spanning Tree instances on the same device

• The port blocking problem (avoid cycles) can be solved by binding every MST instance to one or more VLANs
Experiment 9:
Multiple Spanning Tree: IEEE 802.1s

STP 802.1d

Danny
Jason
Freddy
Jimmy
Port 20
Port 1
Port 10
Port 21
Port 21
Port 20
Port 10
Port 1
Experiment 9:
Multiple Spanning Tree: IEEE 802.1s

• Commands:
  – set lacp disable
  – set spantree version stpcompatible
  – set vlan create 2
  – set vlan create 3
  – set vlan name 2 RED
  – set vlan name 3 GREEN
  – set port vlan fe.1.1 2
  – set port vlan fe.1.21 2
  – set port vlan fe.1.10 3
  – set port vlan fe.1.20 3
Experiment 9: Multiple Spanning Tree: IEEE 802.1s

• In the experiments #6 and #7 we saw how 802.1d blocks ports regardless the VLAN they belong to

• We have to configure every switch in the same region (in this case Region 1) to setup MSTP

• So we create two Spanning Tree instances and bind them to different VLANs

• In this case we bind STI #2 to RED VLAN and STI #3 to GREEN VLAN
Experiment 9: Multiple Spanning Tree: IEEE 802.1s

Bridge B: 192.168.1.1
Bridge B: 192.168.1.2

STP 802.1s
- STP Instance 2
- STP Instance 3

Region 1
- red vlan
- green vlan

Danny
- red vlan

Jason
- green vlan

Freddy
- green vlan

Jimmy
- red vlan
Experiment 9: Multiple Spanning Tree: IEEE 802.1s

• Commands:
  – set spantree version mstp
  – set spantree mstcfgid cfghname REGION1
  – set spantree msti sid 2 create
  – set spantree msti sid 3 create
  – set spantree mstmap 2 sid 2
  – set spantree mstmap 3 sid 3
Experiment 9:
Multiple Spanning Tree: IEEE 802.1s

Bridge B:
192.168.1.1

Port 1
Port 10
Port 20
Port 21

Region 1
red vlan
green vlan

STP 802.1s
- STP Instance 2
- STP Instance 3

192.168.1.1

<table>
<thead>
<tr>
<th>SID</th>
<th>Port</th>
<th>State</th>
<th>Role</th>
<th>Cost</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Te.1.20</td>
<td>Forwarding</td>
<td>Designed</td>
<td>200000</td>
<td>128</td>
</tr>
<tr>
<td>2</td>
<td>Te.1.21</td>
<td>Forwarding</td>
<td>Designed</td>
<td>200000</td>
<td>128</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SID</th>
<th>Port</th>
<th>State</th>
<th>Role</th>
<th>Cost</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>Te.1.20</td>
<td>Forwarding</td>
<td>Designed</td>
<td>200000</td>
<td>128</td>
</tr>
<tr>
<td>3</td>
<td>Te.1.21</td>
<td>Forwarding</td>
<td>Designed</td>
<td>200000</td>
<td>128</td>
</tr>
</tbody>
</table>
Experiment 9: Multiple Spanning Tree: IEEE 802.1s

192.168.1.2

<table>
<thead>
<tr>
<th>SID</th>
<th>Port</th>
<th>State</th>
<th>Role</th>
<th>Cost</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>fe.1.20</td>
<td>Forwarding</td>
<td>Root</td>
<td>200000</td>
<td>128</td>
</tr>
<tr>
<td>2</td>
<td>fe.1.21</td>
<td>Discarding</td>
<td>Alternate</td>
<td>200000</td>
<td>128</td>
</tr>
<tr>
<td>3</td>
<td>fe.1.20</td>
<td>Forwarding</td>
<td>Root</td>
<td>200000</td>
<td>128</td>
</tr>
<tr>
<td>3</td>
<td>fe.1.21</td>
<td>Discarding</td>
<td>Alternate</td>
<td>200000</td>
<td>128</td>
</tr>
</tbody>
</table>
Experiment 9: Multiple Spanning Tree: IEEE 802.1s

• It can be noted from the previous slides that we have to manually configure every STI.

• In particular we analyze the network to specify a well-balanced traffic flow by setting port priority.

• This manually configuration is fundamental in order to avoid essential ports’ blocking: because of the blocking of port 21 on 192.168.1.2, in fact, we can’t communicate on RED VLAN.
Experiment 9: Multiple Spanning Tree: IEEE 802.1s

• Every port has the same priority for each instance: this forces 802.1s to block port 21 to avoid cycles on both instances

• By changing port priority for each Spanning Tree instance even though we still have some ports in blocking, those are different for each STI

• This way the hosts can communicate
Experiment 9: Multiple Spanning Tree: IEEE 802.1s

Region 1

Bridge B:
192.168.1.1

Bridge B:
192.168.1.2

Jason
Danny
Freddy
Jimmy

Port 20
Port 1
Port 10
Port 21
Port 21
Port 20
Port 10
Port 1

red vlan
green vlan

red vlan
green vlan
red vlan
green vlan

STP 802.1s

- STP Instance 2
- STP Instance 3
Experiment 9:
Multiple Spanning Tree: IEEE 802.1s

• Commands:
  – `set spantree portpri fe.1.20 0 sid 2`
  – `set spantree portpri fe.1.21 240 sid 2`
  – `set spantree portpri fe.1.20 240 sid 3`
  – `set spantree portpri fe.1.21 0 sid 3`
Experiment 9:
Multiple Spanning Tree: IEEE 802.1s

<table>
<thead>
<tr>
<th>SID</th>
<th>Port</th>
<th>State</th>
<th>Role</th>
<th>Cost</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>fe.1.20</td>
<td>Forwarding</td>
<td>Designated</td>
<td>200000</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>fe.1.21</td>
<td>Forwarding</td>
<td>Designated</td>
<td>200000</td>
<td>240</td>
</tr>
<tr>
<td>3</td>
<td>fe.1.20</td>
<td>Forwarding</td>
<td>Designated</td>
<td>200000</td>
<td>240</td>
</tr>
<tr>
<td>3</td>
<td>fe.1.21</td>
<td>Forwarding</td>
<td>Designated</td>
<td>200000</td>
<td>0</td>
</tr>
</tbody>
</table>

STP 802.1s
- STP Instance 2
- STP Instance 3
Experiment 9: Multiple Spanning Tree: IEEE 802.1s

STP 802.1s

Bridge B:
192.168.1.2

Region 1

red vlan
green vlan

STP 802.1s

STP Instance 2
STP Instance 3

192.168.1.2

<table>
<thead>
<tr>
<th>SID</th>
<th>Port</th>
<th>State</th>
<th>Role</th>
<th>Cost</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Fe.1.20</td>
<td>Forwarding</td>
<td>Root</td>
<td>200000</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>Fe.1.21</td>
<td>Discarding</td>
<td>Alternate</td>
<td>280000</td>
<td>240</td>
</tr>
<tr>
<td>3</td>
<td>Fe.1.20</td>
<td>Discarding</td>
<td>Alternate</td>
<td>200000</td>
<td>240</td>
</tr>
<tr>
<td>3</td>
<td>Fe.1.21</td>
<td>Forwarding</td>
<td>Root</td>
<td>200000</td>
<td>0</td>
</tr>
</tbody>
</table>

Monaco, Pagano, Passariello, Piacenza, Pintus
Experiments on Virtual Lans and the Spanning Tree Protocol Performed on Enterasys Equipments
Experiment 9:
Multiple Spanning Tree: IEEE 802.1s

Freddy@host:~$ ping 192.168.1.10
PING 192.168.1.10 (192.168.1.10) 56(84) bytes of data.
64 bytes from 192.168.1.10: icmp_seq=1 ttl=64 time=3.70 ms
64 bytes from 192.168.1.10: icmp_seq=2 ttl=64 time=0.177 ms
64 bytes from 192.168.1.10: icmp_seq=3 ttl=64 time=0.170 ms
64 bytes from 192.168.1.10: icmp_seq=4 ttl=64 time=0.174 ms
64 bytes from 192.168.1.10: icmp_seq=5 ttl=64 time=0.174 ms

Jason@host:~$ ping 192.168.1.14
PING 192.168.1.14 (192.168.1.14) 56(84) bytes of data.
64 bytes from 192.168.1.14: icmp_seq=1 ttl=64 time=0.174 ms
64 bytes from 192.168.1.14: icmp_seq=2 ttl=64 time=0.178 ms
64 bytes from 192.168.1.14: icmp_seq=3 ttl=64 time=0.171 ms
64 bytes from 192.168.1.14: icmp_seq=4 ttl=64 time=0.169 ms
64 bytes from 192.168.1.14: icmp_seq=5 ttl=64 time=0.176 ms
Experiment 9:
Multiple Spanning Tree: IEEE 802.1s

Danny@host:~$ ping 192.168.1.13
PING 192.168.1.13 (192.168.1.13) 56(84) bytes of data.
64 bytes from 192.168.1.13: icmp_seq=1 ttl=64 time=3.02 ms
64 bytes from 192.168.1.13: icmp_seq=2 ttl=64 time=0.174 ms
64 bytes from 192.168.1.13: icmp_seq=3 ttl=64 time=0.171 ms
64 bytes from 192.168.1.13: icmp_seq=4 ttl=64 time=0.175 ms
64 bytes from 192.168.1.13: icmp_seq=5 ttl=64 time=0.218 ms

Jimmy@host:~$ ping 192.168.1.11
PING 192.168.1.11 (192.168.1.11) 56(84) bytes of data.
64 bytes from 192.168.1.11: icmp_seq=1 ttl=64 time=0.175 ms
64 bytes from 192.168.1.11: icmp_seq=2 ttl=64 time=0.171 ms
64 bytes from 192.168.1.11: icmp_seq=3 ttl=64 time=0.181 ms
64 bytes from 192.168.1.11: icmp_seq=4 ttl=64 time=0.168 ms
64 bytes from 192.168.1.11: icmp_seq=5 ttl=64 time=0.170 ms
Experiment 10:
Different Root Bridge for different MSTI

• Target:
  – Display that each MSTI can have a different Root Bridge
Experiment 10:
Different Root Bridge for different MSTI

Region 1

Bridge B: 192.168.1.1
Port 1
Port 10
Port 20
Port 21

Bridge B: 192.168.1.2
Port 1
Port 10
Port 20
Port 21

STP 802.1s

- STP Instance 2
- STP Instance 3

STP Instance 1

red vlan

green vlan

Danny
Jason
Freddy
Jimmy

red vlan
green vlan

red vlan
green vlan
Experiment 10: Different Root Bridge for different MSTI

• Commands:

– set lACP disable
– set vlan create 2
– set vlan create 3
– set vlan name 2 RED
– set vlan name 3 GREEN
– set port vlan fe.1.1 2
– set port vlan fe.1.21 2
– set port vlan fe.1.10 3
– set port vlan fe.1.20 3

– set spantree mstcfgid cfgname REGION1
– set spantree msti sid 2 create
– set spantree msti sid 3 create
– set spantree mstmap 2 sid 2
– set spantree mstmap 3 sid 3
– set spantree portpri fe.1.20 0 sid 2
– set spantree portpri fe.1.21 240 sid 2
– set spantree portpri fe.1.20 240 sid 3
– set spantree portpri fe.1.21 0 sid 3
Experiment 10: Different Root Bridge for different MSTI

- Switches’ configuration as from experiment #9.

- We also set every SID’s Bridge Priority so that MSTP elects a different Root Bridge for each instance.

- We want that switch **192.168.1.1** becomes Root for instance #3 and switch **192.168.1.2** becomes Root for instance #2.
Experiment 10: Different Root Bridge for different MSTI

Region 1

Bridge B: 192.168.1.1

Port 1
Port 10
Port 20
Port 21

Jane
Freddy
Danny
Jason

Bridge B: 192.168.1.2

Port 1
Port 10
Port 20
Port 21

STP 802.1s

STP Instance 2
STP Instance 3

red vlan
green vlan

Danny
Jason
Freddy
Jimmy

red vlan
green vlan
red vlan
Experiment 10:
Different Root Bridge for different MSTI

• Commands:
  • Bridge 192.168.1.1
    – set spantree priority 4096 2
    – set spantree priority 0 3
  • Bridge 192.168.1.2
    – set spantree priority 0 2
    – set spantree priority 4096 3
Experiment 10:
Different Root Bridge for different MSTI

### Bridge 192.168.1.1

<table>
<thead>
<tr>
<th>SID</th>
<th>Port</th>
<th>State</th>
<th>Role</th>
<th>Cost</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>fe.1.20</td>
<td>Discarding</td>
<td>Alternate</td>
<td>200000</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>fe.1.21</td>
<td>Forwarding</td>
<td>Root</td>
<td>200000</td>
<td></td>
</tr>
</tbody>
</table>

```bash
B2(su)-->show spantree stats active sid 2
Spanning tree status - enabled
Spanning tree instance - 2
Designated Root MacAddr - 00:11:88:1A:19:E1
Designated Root Port - fe.1.21
Designated Root Priority - 0
Designated Root Cost - 200000
Root Max Age - 20
Root Hello Time - 2
Root Forward Delay - 15
Bridge ID MAC Address - 00:11:88:1A:19:69
Bridge ID Priority - 4096
Bridge Max Age - 20
Bridge Hello Time - 2
Bridge Forward Delay - 15
Topology Change Count - 5
Time Since Top Change - 0 days 0:4:56
Max Hops - 20
```

```bash
B2(su)-->show spantree stats active sid 3
Spanning tree status - enabled
Spanning tree instance - 3
Designated Root MacAddr - 00:11:88:1A:19:69
Designated Root Port - 0
Designated Root Priority - 0
Designated Root Cost - 0
Root Max Age - 20
Root Hello Time - 2
Root Forward Delay - 15
Bridge ID MAC Address - 00:11:88:1A:19:69
Bridge ID Priority - 0
Bridge Max Age - 20
Bridge Hello Time - 2
Bridge Forward Delay - 15
Topology Change Count - 2
Time Since Top Change - 0 days 0:45:26
Max Hops - 20
```

<table>
<thead>
<tr>
<th>SID</th>
<th>Port</th>
<th>State</th>
<th>Role</th>
<th>Cost</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>fe.1.20</td>
<td>Forwarding</td>
<td>Designated</td>
<td>200000</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>fe.1.21</td>
<td>Forwarding</td>
<td>Designated</td>
<td>200000</td>
<td></td>
</tr>
</tbody>
</table>

Monaco, Pagano, Passariello, Piacenza, Pintus
Experiments on Virtual Lans and the Spanning Tree Protocol Performed on Enterasys Equipments
## Experiment 10:
### Different Root Bridge for different MSTI

B2(su)->show spantree stats active sid 2

<table>
<thead>
<tr>
<th>Spanning tree status</th>
<th>- enabled</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spanning tree instance</td>
<td>- 2</td>
</tr>
<tr>
<td>Designated Root MacAddr</td>
<td>00:11:88:1A:19:E1</td>
</tr>
<tr>
<td>Designated Root Port</td>
<td>- 0</td>
</tr>
<tr>
<td>Designated Root Priority</td>
<td>- 0</td>
</tr>
<tr>
<td>Designated Root Cost</td>
<td>- 0</td>
</tr>
<tr>
<td>Root Max Age</td>
<td>- 20</td>
</tr>
<tr>
<td>Root Hello Time</td>
<td>- 2</td>
</tr>
<tr>
<td>Root Forward Delay</td>
<td>- 15</td>
</tr>
<tr>
<td>Bridge ID MAC Address</td>
<td>00:11:88:1A:19:E1</td>
</tr>
<tr>
<td>Bridge ID Priority</td>
<td>- 0</td>
</tr>
<tr>
<td>Bridge Max Age</td>
<td>- 20</td>
</tr>
<tr>
<td>Bridge Hello Time</td>
<td>- 2</td>
</tr>
<tr>
<td>Bridge Forward Delay</td>
<td>- 15</td>
</tr>
<tr>
<td>Topology Change Count</td>
<td>- 5</td>
</tr>
<tr>
<td>Time Since Top Change</td>
<td>- 0 days 0:5:28</td>
</tr>
<tr>
<td>Max Hops</td>
<td>- 20</td>
</tr>
</tbody>
</table>

B2(su)->show spantree stats active sid 3

<table>
<thead>
<tr>
<th>Spanning tree status</th>
<th>- enabled</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spanning tree instance</td>
<td>- 3</td>
</tr>
<tr>
<td>Designated Root MacAddr</td>
<td>00:11:88:1A:19:E1</td>
</tr>
<tr>
<td>Designated Root Port</td>
<td>Te.1.20</td>
</tr>
<tr>
<td>Designated Root Priority</td>
<td>- 0</td>
</tr>
<tr>
<td>Designated Root Cost</td>
<td>- 200000</td>
</tr>
<tr>
<td>Root Max Age</td>
<td>- 20</td>
</tr>
<tr>
<td>Root Hello Time</td>
<td>- 2</td>
</tr>
<tr>
<td>Root Forward Delay</td>
<td>- 15</td>
</tr>
<tr>
<td>Bridge ID MAC Address</td>
<td>00:11:88:1A:19:E1</td>
</tr>
<tr>
<td>Bridge ID Priority</td>
<td>- 4006</td>
</tr>
<tr>
<td>Bridge Max Age</td>
<td>- 20</td>
</tr>
<tr>
<td>Bridge Hello Time</td>
<td>- 2</td>
</tr>
<tr>
<td>Bridge Forward Delay</td>
<td>- 15</td>
</tr>
<tr>
<td>Topology Change Count</td>
<td>- 2</td>
</tr>
<tr>
<td>Time Since Top Change</td>
<td>- 0 days 0:49:44</td>
</tr>
<tr>
<td>Max Hops</td>
<td>- 20</td>
</tr>
</tbody>
</table>

### Bridge 192.168.1.2

<table>
<thead>
<tr>
<th>SID</th>
<th>Port</th>
<th>State</th>
<th>Role</th>
<th>Cost</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>fe.1.20</td>
<td>Forwarding</td>
<td>Designated</td>
<td>200000</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>fe.1.21</td>
<td>Forwarding</td>
<td>Designated</td>
<td>200000</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SID</th>
<th>Port</th>
<th>State</th>
<th>Role</th>
<th>Cost</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>fe.1.20</td>
<td>Forwarding</td>
<td>Root</td>
<td>200000</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>Te.1.21</td>
<td>Discarding</td>
<td>Alternate</td>
<td>200000</td>
<td>240</td>
</tr>
</tbody>
</table>
Experiment 11:
Visibility between different STP Regions

• Target:
  – Display visibility between different STP Regions with the same VLANs
Experiment 11:
Visibility between different MSTP Regions

Monaco, Pagano, Passariello, Piacenza, Pintus
Experiments on Virtual Lans and the Spanning Tree
Protocol Performed on Enterasys Equipments
Experiment 11:
Visibility between different STP Regions

• Commands Region 1:
  – set vlan create 2
  – set vlan create 3
  – set vlan name 2 RED
  – set vlan name 3 GREEN
  – set port vlan fe.1.6 2
  – set port vlan fe.1.21 2
  – set port vlan fe.1.5 3
  – set port vlan fe.1.20 3
  – set lacp disable

  – set spantree mstcfgid cfgname REGION1
  – set spantree msti sid 2 create
  – set spantree msti sid 3 create
  – set spantree mstmap 2 sid 2
  – set spantree mstmap 3 sid 3
  – set spantree portpri fe.1.20 0 sid 2
  – set spantree portpri fe.1.21 240 sid 2
  – set spantree portpri fe.1.20 240 sid 3
  – set spantree portpri fe.1.21 0 sid 3
Experiment 11: Visibility between different STP Regions

• Commands Region 2:

– set vlan create 2
– set vlan create 3
– set vlan name 2 RED
– set vlan name 3 GREEN
– set port vlan fe.1.6 2
– set port vlan fe.1.21 2
– set port vlan fe.1.5 3
– set port vlan fe.1.20 3
– set lacp disable

– set spantree mstcfgid cfgname REGION2
– set spantree msti sid 2 create
– set spantree msti sid 3 create
– set spantree mstmap 2 sid 2
– set spantree mstmap 3 sid 3
– set spantree portpri fe.1.20 0 sid 2
– set spantree portpri fe.1.21 240 sid 2
– set spantree portpri fe.1.20 240 sid 3
– set spantree portpri fe.1.21 0 sid 3
Experiment 11:
Visibility between different STP Regions

• Commands trunk 1Q between REGION1 and REGION2:
  – set port ingress-filter fe.1.11
  – set vlan egress 2 fe.1.11 tagged
  – set vlan egress 3 fe.1.11 tagged
  – set port discard fe.1.11 untagged
Experiment 11: Visibility between different STP Regions

• MSTP on both Regions

• Trunk 1Q connects the same VLAN on different regions

• Hosts on the same VLAN can reach each other, even if they are located on different regions
Experiment 11:
Visibility between different STP Regions

Jason@host:~$ ping 192.168.1.13
PING 192.168.1.13 (192.168.1.13) 56(84) bytes of data.
64 bytes from 192.168.1.13: icmp_seq=1 ttl=64 time=0.165 ms
64 bytes from 192.168.1.13: icmp_seq=2 ttl=64 time=0.163 ms
64 bytes from 192.168.1.13: icmp_seq=3 ttl=64 time=0.164 ms
64 bytes from 192.168.1.13: icmp_seq=4 ttl=64 time=0.162 ms
64 bytes from 192.168.1.13: icmp_seq=5 ttl=64 time=0.161 ms

Tommy@host:~$ ping 192.168.1.14
PING 192.168.1.14 (192.168.1.14) 56(84) bytes of data.
64 bytes from 192.168.1.14: icmp_seq=1 ttl=64 time=3.18 ms
64 bytes from 192.168.1.14: icmp_seq=2 ttl=64 time=0.461 ms
64 bytes from 192.168.1.14: icmp_seq=3 ttl=64 time=0.212 ms
64 bytes from 192.168.1.14: icmp_seq=4 ttl=64 time=0.210 ms
64 bytes from 192.168.1.14: icmp_seq=5 ttl=64 time=0.206 ms
Experiment 12(a):
Root Bridge Election for CST & CIST

• Target:
  – Create two or more different regions and check which bridge is the Root Bridge for CST (Common Spanning Tree) and which one is the Root Bridge for CIST (Common Instance Spanning Tree)
  
  – Topology:
    • Region 1: 2 switches
    • Region 2: 1 switch
    • Trunk 1Q between Region 1 and Region 2
Experiment 12(a): Root Bridge Election for CST & CIST

Monaco, Pagano, Passariello, Piacenza, Pintus
Experiments on Virtual Lans and the Spanning Tree Protocol Performed on Enterasys Equipments
Experiment 12(a):
Root Bridge Election for CST & CIST

- Commands Region 1:
  
  - set vlan create 2
  - set vlan create 3
  - set vlan name 2 GREEN
  - set vlan name 3 RED
  - set port vlan fe.1.5 2
  - set port vlan fe.1.20 2
  - set port vlan fe.1.6 3
  - set port vlan fe.1.21 3
  - set lacp disable

  - set spantree mstcfgid cfgname REGION1
  - set spantree msti sid 2 create
  - set spantree msti sid 3 create
  - set spantree mstmap 2 sid 2
  - set spantree mstmap 3 sid 3
  - set spantree portpri fe.1.20 0 sid 2
  - set spantree portpri fe.1.21 240 sid 2
  - set spantree portpri fe.1.20 240 sid 3
  - set spantree portpri fe.1.21 0 sid 3
Experiment 12(a):
Root Bridge Election for CST & CIST

• Commands Region 2:

   – set vlan create 2
   – set vlan create 3
   – set vlan name 2 GREEN
   – set vlan name 3 RED
   – set port vlan fe.1.5 2
   – set port vlan fe.1.20 2
   – set port vlan fe.1.6 3
   – set port vlan fe.1.21 3
   – set lacp disable
   – set spantree mstcfgid cfgname REGION2
Experiment 12(a):
Root Bridge Election for CST & CIST

- Commands:
  
  – set port ingress-filter fe.1.11 enable
  – set vlan egress 2 fe.1.11 tagged
  – set vlan egress 3 fe.1.11 tagged
  – set port discard fe.1.11 untagged

The above commands are used for the switches linked by trunk 1Q
Experiment 12(a):
Root Bridge Election for CST & CIST

- MSTP elects just one Network Root Bridge and two Regional Root Bridge one for each region

- One of the two Regional Root Bridge will also be the Network Root Bridge
Experiment 12(a):
Root Bridge Election for CST & CIST

- Sniffing from 192.168.1.4 we note that 192.168.1.6 is the Regional Root Bridge (the Root Bridge for CIST) for the two VLANs in Region 1

<table>
<thead>
<tr>
<th>MST Extension</th>
</tr>
</thead>
<tbody>
<tr>
<td>MST Config ID format selector: 0</td>
</tr>
<tr>
<td>MST Config name: REGION1</td>
</tr>
<tr>
<td>MST Config revision: 0</td>
</tr>
<tr>
<td>MST Config digest: 8A9442199657EA49D1124EA768B5D9A2</td>
</tr>
<tr>
<td>CIST Internal Root Path Cost: 20000</td>
</tr>
<tr>
<td>CIST Bridge Identifier: 32768 / 00:01:f4:7f:05:77</td>
</tr>
<tr>
<td>CIST Remaining hops: 19</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MSTID 2, Regional Root Identifier 32768 / 00:01:f4:5c:3f:20</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSTI flags: Oxfe (Master, Agreement, Forwarding, Learning, Port Role: Designated, Proposal)</td>
</tr>
<tr>
<td>MSTID 2, priority 32768 Root Identifier 00:01:f4:5c:3f:20</td>
</tr>
<tr>
<td>Internal root path cost: 20000</td>
</tr>
<tr>
<td>Bridge Identifier Priority: 8</td>
</tr>
<tr>
<td>Port identifier priority: 8</td>
</tr>
<tr>
<td>Remaining hops: 19</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MSTID 3, Regional Root Identifier 32768 / 00:01:f4:5c:3f:20</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSTI flags: Oxfe (Master, Agreement, Forwarding, Learning, Port Role: Designated, Proposal)</td>
</tr>
<tr>
<td>MSTID 3, priority 32768 Root Identifier 00:01:f4:5c:3f:20</td>
</tr>
<tr>
<td>Internal root path cost: 20000</td>
</tr>
<tr>
<td>Bridge Identifier Priority: 8</td>
</tr>
<tr>
<td>Port identifier priority: 8</td>
</tr>
<tr>
<td>Remaining hops: 19</td>
</tr>
</tbody>
</table>
Experiment 12(a):
Root Bridge Election for CST & CIST

• 192.168.1.3 is the Regional Root Bridge for Region 2

<table>
<thead>
<tr>
<th>MST Extension</th>
</tr>
</thead>
<tbody>
<tr>
<td>MST Config ID format selector: 0</td>
</tr>
<tr>
<td>MST Config name: REGION2</td>
</tr>
<tr>
<td>MST Config revision: 0</td>
</tr>
<tr>
<td>MST Config digest: 8A9442199657EA49D1124EA768B5D9A2</td>
</tr>
<tr>
<td>CIST Internal Root Path Cost: 0</td>
</tr>
<tr>
<td>CIST Bridge Identifier: 32768 / 00:01:f4:5c:06:eb</td>
</tr>
<tr>
<td>CIST Remaining hops: 20</td>
</tr>
</tbody>
</table>

| MSTID 2, Regional Root Identifier 32768 / 00:01:f4:5c:06:eb |
| MSTI flags: Ox7e (Agreement, Forwarding, Learning, Port Role: Designated, Proposal) |
| MSTID 2, priority 32768 Root Identifier 00:01:f4:5c:06:eb |
| Internal root path cost: 0 |
| Bridge Identifier Priority: 8 |
| Port identifier priority: 8 |
| Remaining hops: 20 |

| MSTID 3, Regional Root Identifier 32768 / 00:01:f4:5c:06:eb |
| MSTI flags: Ox7e (Agreement, Forwarding, Learning, Port Role: Designated, Proposal) |
| MSTID 3, priority 32768 Root Identifier 00:01:f4:5c:06:eb |
| Internal root path cost: 0 |
| Bridge Identifier Priority: 8 |
| Port identifier priority: 8 |
| Remaining hops: 20 |
Experiment 12(a): Root Bridge Election for CST & CIST

- Sniffing from 192.168.1.3 (Region 2) we note that 192.168.1.3 is also the Network Root Bridge (the Root Bridge for CST)
Experiment 12(b):
Root Bridge Election for CST & CIST

• Target:
  – Create two or more different regions and check which bridge is the Root Bridge for CST (Common Spanning Tree) and which one is the Root Bridge for CIST (Common Instance Spanning Tree)

  – Topology:
    • Region 1: 2 switches
    • Region 2: 2 switches
    • Trunk 1Q between Region 1 and Region 2
Experiment 12(b):
Root Bridge Election for CST & CIST

Monaco, Pagano, Passariello, Piacenza, Pintus
Experiments on Virtual Lans and the Spanning Tree Protocol Performed on Enterasys Equipments
Experiment 12(b):
Root Bridge Election for CST & CIST

• Commands Region 2:

  – set_vlan create 2
  – set_vlan create 3
  – set_vlan name 2 GREEN
  – set_vlan name 3 RED
  – set_port_vlan fe.1.5 2
  – set_port_vlan fe.1.20 2
  – set_port_vlan fe.1.6 3
  – set_port_vlan fe.1.21 3
  – set_lacp disable

  – set_spantree_mstcfgid cfaname REGION2
  – set_spantree_msti_sid 2 create
  – set_spantree_msti_sid 3 create
  – set_spantree_mstmap 2 sid 2
  – set_spantree_mstmap 3 sid 3
  – set_spantree_portpri fe.1.20 0 sid 2
  – set_spantree_portpri fe.1.21 240 sid 2
  – set_spantree_portpri fe.1.20 240 sid 3
  – set_spantree_portpri fe.1.21 0 sid 3

Commands used for Region 1 and Trunk 1Q link are the same as discussed before
**Experiment 12(b): Root Bridge Election for CST & CIST**

- Sniffing from **192.168.1.4** we note that **192.168.1.6** is elected as the Regional Root Bridge for Region 1

<table>
<thead>
<tr>
<th>MST Extension</th>
</tr>
</thead>
<tbody>
<tr>
<td>MST Config ID format selector: 0</td>
</tr>
<tr>
<td>MST Config name: REGION1</td>
</tr>
<tr>
<td>MST Config revision: 0</td>
</tr>
<tr>
<td>MST Config digest: 8A9442199657EA49D1124EA768B5D9A2</td>
</tr>
<tr>
<td>CIST Internal Root Path Cost: 20000</td>
</tr>
<tr>
<td>CIST Bridge Identifier: 32768 / 00:01:f4:7f:05:77</td>
</tr>
<tr>
<td>CIST Remaining hops: 19</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MSTID 2, Regional Root Identifier 32768 / 00:01:f4:5c:3f:20</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSTI flags: Oxfe (Master, Agreement, Forwarding, Learning, Port Role: Designated, Proposal)</td>
</tr>
<tr>
<td>MSTID 2, priority 32768 Root Identifier 00:01:f4:5c:3f:20</td>
</tr>
<tr>
<td>Internal root path cost: 20000</td>
</tr>
<tr>
<td>Bridge Identifier Priority: 8</td>
</tr>
<tr>
<td>Port identifier priority: 8</td>
</tr>
<tr>
<td>Remaining hops: 19</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MSTID 3, Regional Root Identifier 32768 / 00:01:f4:5c:3f:20</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSTI flags: Oxfe (Master, Agreement, Forwarding, Learning, Port Role: Designated, Proposal)</td>
</tr>
<tr>
<td>MSTID 3, priority 32768 Root Identifier 00:01:f4:5c:3f:20</td>
</tr>
<tr>
<td>Internal root path cost: 20000</td>
</tr>
<tr>
<td>Bridge Identifier Priority: 8</td>
</tr>
<tr>
<td>Port identifier priority: 8</td>
</tr>
<tr>
<td>Remaining hops: 19</td>
</tr>
</tbody>
</table>
Experiment 12(b):
Root Bridge Election for CST & CIST

- Sniffing from 192.168.1.5 we note that 192.168.1.3 is elected as Regional Root Bridge for Region 2

```
MST Extension
MST Config ID format selector: 0
MST Config name: REGION2
MST Config revision: 0
MST Config digest: 8A9442199657EA49D1124EA768B5D9A2
CIST Internal Root Path Cost: 200000
CIST Bridge Identifier: 32768 / 00:11:88:16:04:2f
CIST Remaining hops: 19

MSTID 2, Regional Root Identifier 32768 / 00:01:f4:5c:06:eb
MSTI flags: Ox7e (Agreement, Forwarding, Learning, Port Role: Designated, Proposal)
MSTID 2, priority 32768 Root Identifier 00:01:f4:5c:06:eb
Internal root path cost: 200000
Bridge Identifier Priority: 8
Port identifier priority: 8
Remaining hops: 19

MSTID 3, Regional Root Identifier 32768 / 00:01:f4:5c:06:eb
MSTI flags: Ox7e (Agreement, Forwarding, Learning, Port Role: Designated, Proposal)
MSTID 3, priority 32768 Root Identifier 00:01:f4:5c:06:eb
Internal root path cost: 200000
Bridge Identifier Priority: 8
Port identifier priority: 8
Remaining hops: 19
```
Experiment 12(b):
Root Bridge Election for CST & CIST

- *192.168.1.3* is elected Network Root Bridge and Regional Root Bridge for its own region

<table>
<thead>
<tr>
<th>Spanning Tree Protocol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Protocol Identifier: Spanning Tree Protocol (0x0000)</td>
</tr>
<tr>
<td>Protocol Version Identifier: Multiple Spanning Tree (3)</td>
</tr>
<tr>
<td>BPDU Type: Rapid/Multiple Spanning Tree (0x02)</td>
</tr>
<tr>
<td>BPDU flags: 0x7e (Agreement, Forwarding, Learning, Port Role: Designated, Proposal)</td>
</tr>
<tr>
<td>Root Identifier: 32768 / 00:01:f4:5c:06:eb</td>
</tr>
<tr>
<td>Root Path Cost: 0</td>
</tr>
<tr>
<td>Bridge Identifier: 32768 / 00:01:f4:5c:06:eb</td>
</tr>
<tr>
<td>Port Identifier: 0x8046</td>
</tr>
</tbody>
</table>
Experiment 12(c): Root Bridge Election for CST & CIST

• Target:
  – Create two or more different regions and check which bridge is the Root Bridge for CST (Common Spanning Tree) and which one is the Root Bridge for CIST (Common instance Spanning Tree)

  – Topology:
    • Region 1: 2 switches
    • Region 2: 2 switches
    • Region 3: 2 switches
    • Trunk 1Q between Region 1 and Region 2 and between Region 1 and Region 3
Experiment 12(c):
Root Bridge Election for CST & CIST

Monaco, Pagano, Passariello, Piacenza, Pintus
Experiments on Virtual Lans and the Spanning Tree Protocol Performed on Enterasys Equipments
Experiment 12(c): Root Bridge Election for CST & CIST

• Commands Region 3:

- `set vlan create 2`
- `set vlan create 3`
- `set vlan name 2 GREEN`
- `set vlan name 3 RED`
- `set port vlan fe.1.5 2`
- `set port vlan fe.1.20 2`
- `set port vlan fe.1.6 3`
- `set port vlan fe.1.21 3`
- `set lacp disable`
- `set spantree mstcfgid cfgname REGION3`
- `set spantree msti sid 2 create`
- `set spantree msti sid 3 create`
- `set spantree mstmap 2 sid 2`
- `set spantree mstmap 3 sid 3`
- `set spantree portpri fe.1.20 0 sid 2`
- `set spantree portpri fe.1.21 240 sid 2`
- `set spantree portpri fe.1.20 240 sid 3`
- `set spantree portpri fe.1.21 0 sid 3`

Commands used for Region 1, Region 2 and Trunk 1Q link are the same as discussed before
Experiment 12(c):
Root Bridge Election for CST & CIST

• Commands:

  – set port ingress-filter fe.1.12 enable
  – set vlan egress 2 fe.1.12 tagged
  – set vlan egress 3 fe.1.12 tagged
  – set port discard fe.1.12 untagged

The above commands are used for the switches linked by trunk 1Q in Region 1 and Region 3
Experiment 12(c):
Root Bridge Election for CST & CIST

- Sniffing from 192.168.1.2 we note that 192.168.1.1 is elected as the Regional Root Bridge for Region 3
Experiment 12(c): Root Bridge Election for CST & CIST

- Sniffing again from 192.168.1.2 we note that 192.168.1.3 is still Network Root Bridge

- We want 192.168.1.1 to become Network Root Bridge: we induce this change tuning the bridge priority values
Experiment 12(c):
Root Bridge Election for CST & CIST

STP 802.1s
- STP Instance 2
- STP Instance 3

Segment 1:
- Bridge N3: 192.168.1.4
  00:01:47:05:77

Segment 2:
- Bridge N3: 192.168.1.3
  00:01:45:06:eb

Segment 3:
- Bridge N3: 192.168.1.5
  00:11:88:16:04:2f

Monaco, Pagano, Passariello, Piacenza, Pintus
Experiments on Virtual Lans and the Spanning Tree Protocol Performed on Enterasys Equipments
Experiment 12(c):
Root Bridge Election for CST & CIST

• Commands:
  – `set spantree priority 0 0`
Experiment 12(c):
Root Bridge Election for CST & CIST

• Sniffing now from any switch of the network, we note that 192.168.1.1 has become the new Network Root Bridge
Experiment 13: VLAN & Link Aggregation

• Target:
  – Analyze the Network with Link Aggregation Control Protocol (802.3ad) enabled and disabled
Experiment 13: VLAN & Link Aggregation

• LACP is used to aggregate one or more physical ports into a single logical port, in order to improve bandwidth and fault tolerance

• By default LACP is enabled on the Enterasys devices
Experiment 13(a):
VLAN & Link Aggregation

MAC Address: 00:11:88:1A:19:69
MAC Address: 00:11:88:1A:19:E1

LACP disabled:
port 5 blocked
Experiment 13(a):
VLAN & Link Aggregation

• Commands:
  – set lacp disable
  – set vlan create 2
  – set vlan name 2 GREEN
  – set port vlan fe.1.2 2
  – set port vlan fe.1.5 2
  – set port vlan fe.1.10 2
Experiment 13(a): VLAN & Link Aggregation

<table>
<thead>
<tr>
<th>SID</th>
<th>Port</th>
<th>State</th>
<th>Role</th>
<th>Cost</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>fe.1.2</td>
<td>Forwarding</td>
<td>Root</td>
<td>200000</td>
<td>128</td>
</tr>
<tr>
<td>0</td>
<td>fe.1.5</td>
<td>Discarding</td>
<td>Alternate</td>
<td>200000</td>
<td>128</td>
</tr>
</tbody>
</table>

Spanning tree status - enabled
Spanning tree instance - 0
Designated Root MacAddr - 00:11:88:1A:19:69
Designated Root Port - fe.1.2
Designated Root Priority - 32768
Designated Root Cost - 200000
Root Max Age - 20
Root Hello Time - 2
Root Forward Delay - 15
Bridge ID MAC Address - 00:11:88:1A:19:E1
Bridge ID Priority - 32768
Bridge Max Age - 20
Bridge Hello Time - 2
Bridge Forward Delay - 15
Topology Change Count - 1
Time Since Top Change - 0 days 0:0:25
Max Hops - 20
Experiment 13(a):

VLAN & Link Aggregation

```
Jason@host:~$ ping 192.168.1.14
PING 192.168.1.14 (192.168.1.14) 56(84) bytes of data.
64 bytes from 192.168.1.14: icmp_seq=1 ttl=64 time=3.57 ms
64 bytes from 192.168.1.14: icmp_seq=2 ttl=64 time=0.178 ms
64 bytes from 192.168.1.14: icmp_seq=3 ttl=64 time=0.180 ms
64 bytes from 192.168.1.14: icmp_seq=4 ttl=64 time=0.181 ms
64 bytes from 192.168.1.14: icmp_seq=5 ttl=64 time=0.183 ms
```

```
Freddy@host:~$ ping 192.168.1.10
PING 192.168.1.10 (192.168.1.10) 56(84) bytes of data.
64 bytes from 192.168.1.10: icmp_seq=1 ttl=64 time=0.184 ms
64 bytes from 192.168.1.10: icmp_seq=2 ttl=64 time=0.182 ms
64 bytes from 192.168.1.10: icmp_seq=3 ttl=64 time=0.184 ms
64 bytes from 192.168.1.10: icmp_seq=4 ttl=64 time=0.184 ms
64 bytes from 192.168.1.10: icmp_seq=5 ttl=64 time=0.182 ms
```
Experiment 13(b):
VLAN & Link Aggregation

Bridge B:
192.168.1.1

Bridge B:
192.168.1.3

Aggregated links

Port 2
Port 2

Port 5
Port 5

Port 10
Port 10

LACP enabled
lag not associated to the
GREEN VLAN

Jason

Freddy

Monaco, Pagano, Passariello, Piacenza, Pintus
Experiments on Virtual Lans and the Spanning Tree
Protocol Performed on Enterasys Equipments
Experiment 13(b):
VLAN & Link Aggregation

• Commands:
  – set lacp enable
Experiment 13(b):
VLAN & Link Aggregation

Global Link Aggregation state: enabled
Single Port LAGs: disabled
Aggregator: lag.0.1
Actor
System Identifier: 00:11:88:1A:19:E1
System Priority: 32768
Admin Key: 32768
Oper Key: 32768
Attached Ports: fe.1.2, fe.1.5
Partner
System Identifier: 00:11:88:1A:19:69
System Priority: 32768
Admin Key: 32768
Oper Key: 0

192.168.1.3

Global Link Aggregation state: enabled
Single Port LAGs: disabled
Aggregator: lag.0.1
Actor
System Identifier: 00:11:88:1A:19:69
System Priority: 32768
Admin Key: 32768
Oper Key: 0
Attached Ports: fe.1.2, fe.1.5
Partner
System Identifier: 00:11:88:1A:19:E1
System Priority: 32768
Admin Key: 32768
Oper Key: 32768
Experiment 13(b): VLAN & Link Aggregation

Spanning tree status - enabled
Spanning tree instance - 0
Designated Root MacAddr - 00:11:88:1A:19:69
Designated Root Port - 0
Designated Root Priority - 32768
Designated Root Cost - 0
Root Max Age - 20
Root Hello Time - 2
Root Forward Delay - 15
Bridge ID MAC Address - 00:11:88:1A:19:69
Bridge ID Priority - 32768
Bridge Max Age - 20
Bridge Hello Time - 2
Bridge Forward Delay - 15
Topology Change Count - 2
Time Since Top Change - 0 days 0:0:59
Max Hops - 20

<table>
<thead>
<tr>
<th>SID</th>
<th>Port</th>
<th>State</th>
<th>Role</th>
<th>Cost</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>fe.1.2</td>
<td>Discarding</td>
<td>Disabled</td>
<td>0</td>
<td>128</td>
</tr>
<tr>
<td>0</td>
<td>fe.1.5</td>
<td>Discarding</td>
<td>Disabled</td>
<td>0</td>
<td>128</td>
</tr>
<tr>
<td>0</td>
<td>lag.0.1</td>
<td>Forwarding</td>
<td>Designated</td>
<td>100000</td>
<td>128</td>
</tr>
</tbody>
</table>
Experiment 13(b): VLAN & Link Aggregation

Spanning tree status - enabled
Spanning tree instance - 0
Designated Root MacAddr - 00:11:88:1A:19:69
Designated Root Port - lag.0.1
Designated Root Priority - 32768
Designated Root Cost - 100000
Root Max Age - 20
Root Hello Time - 2
Root Forward Delay - 15
Bridge ID MAC Address - 00:11:88:1A:19:E1
Bridge ID Priority - 32768
Bridge Max Age - 20
Bridge Hello Time - 2
Bridge Forward Delay - 15
Topology Change Count - 2
Time Since Top Change - 0 days 0:0:10
Max Hops - 20

<table>
<thead>
<tr>
<th>SID</th>
<th>Port</th>
<th>State</th>
<th>Role</th>
<th>Cost</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>fe.1.2</td>
<td>Discarding</td>
<td>Disabled</td>
<td>0</td>
<td>128</td>
</tr>
<tr>
<td>0</td>
<td>fe.1.5</td>
<td>Discarding</td>
<td>Disabled</td>
<td>0</td>
<td>128</td>
</tr>
<tr>
<td>0</td>
<td>lag.0.1</td>
<td>Forwarding</td>
<td>Root</td>
<td>100000</td>
<td>128</td>
</tr>
</tbody>
</table>

192.168.1.3
Experiment 13(b):
VLAN & Link Aggregation

Jason@host:~$ ping 192.168.1.14
PING 192.168.1.14 (192.168.1.14) 56(84) bytes of data.
From 192.168.1.10 icmp_seq=1 Destination Host Unreachable
From 192.168.1.10 icmp_seq=2 Destination Host Unreachable
From 192.168.1.10 icmp_seq=3 Destination Host Unreachable
From 192.168.1.10 icmp_seq=4 Destination Host Unreachable
From 192.168.1.10 icmp_seq=5 Destination Host Unreachable
From 192.168.1.10 icmp_seq=6 Destination Host Unreachable

Freddy@host:~$ ping 192.168.1.10
PING 192.168.1.10 (192.168.1.10) 56(84) bytes of data.
From 192.168.1.14 icmp_seq=1 Destination Host Unreachable
From 192.168.1.14 icmp_seq=2 Destination Host Unreachable
From 192.168.1.14 icmp_seq=3 Destination Host Unreachable
From 192.168.1.14 icmp_seq=4 Destination Host Unreachable
From 192.168.1.14 icmp_seq=5 Destination Host Unreachable
From 192.168.1.14 icmp_seq=6 Destination Host Unreachable
Experiment 13(b): VLAN & Link Aggregation

• By default LACP aggregates every link between the switches into lag.0.1 virtual port

• At the beginning the hosts are unreachable because the lag.0.1 port is not associated to the GREEN VLAN

• As soon as lag.0.1 port is associated to the GREEN VLAN, the hosts are reachable
Experiment 13(b): VLAN & Link Aggregation

Bridge B: 192.168.1.1

Bridge B: 192.168.1.3

Aggregated links

LACP enabled lag associated to the GREEN VLAN
Experiment 13(b):
VLAN & Link Aggregation

• Commands:
  – set port vlan lag.0.1 2
Experiment 13(b):
VLAN & Link Aggregation

Jason@host:~$ ping 192.168.1.14
PING 192.168.1.14 (192.168.1.14) 56(84) bytes of data.
64 bytes from 192.168.1.14: icmp_seq=1 ttl=64 time=242 ms
64 bytes from 192.168.1.14: icmp_seq=2 ttl=64 time=4.12 ms
64 bytes from 192.168.1.14: icmp_seq=3 ttl=64 time=999 ms
64 bytes from 192.168.1.14: icmp_seq=4 ttl=64 time=7.62 ms
64 bytes from 192.168.1.14: icmp_seq=5 ttl=64 time=999 ms

Freddy@host:~$ ping 192.168.1.10
PING 192.168.1.10 (192.168.1.10) 56(84) bytes of data.
64 bytes from 192.168.1.10: icmp_seq=1 ttl=64 time=950 ms
64 bytes from 192.168.1.10: icmp_seq=2 ttl=64 time=1.11 ms
64 bytes from 192.168.1.10: icmp_seq=3 ttl=64 time=0.884 ms
64 bytes from 192.168.1.10: icmp_seq=4 ttl=64 time=1.06 ms
64 bytes from 192.168.1.10: icmp_seq=5 ttl=64 time=1.04 ms
Experiment 14: More VLANs & Link Aggregation

• Target:
  – Analyze the Network with Link Aggregation Control Protocol (802.3ad) enabled and disabled, in presence of more VLANs too
Experiment 14:
More VLANs & Link Aggregation

STP 802.1s
- STP Instance 2
- STP Instance 3

Bridge B:
192.168.1.1

Port 3
Port 1
Port 9
Port 10

Bridge B:
192.168.1.2

Port 1
Port 3
Port 9
Port 10

red vlan
green vlan
red vlan
green vlan
red vlan
green vlan
red vlan
green vlan
red vlan
green vlan
red vlan
green vlan
red vlan
green vlan
Experiment 14: More VLANs & Link Aggregation

- set lacp disable
- set vlan create 2
- set vlan create 3
- set vlan name 2 RED
- set vlan name 3 GREEN
- set port vlan fe.1.1 3
- set port vlan fe.1.10 3
- set port vlan fe.1.3 2
- set port vlan fe.1.9 2

- set spantree mstcfgid cfgname REGION1
- set spantree msti sid 2 create
- set spantree msti sid 3 create
- set spantree mstmap 2 sid 2
- set spantree mstmap 3 sid 3
- set spantree portpri fe.1.1 0 sid 3
- set spantree portpri fe.1.1 240 sid 2
- set spantree portpri fe.1.3 0 sid 2
- set spantree portpri fe.1.3 240 sid 3
Experiment 14:
More VLANs & Link Aggregation

Monaco, Pagano, Passariello, Piacenza, Pintus
Experiments on Virtual Lans and the Spanning Tree
Protocol Performed on Enterasys Equipments
Experiment 14: More VLANs & Link Aggregation

- Commands:
  - `set port vlan fe.1.2 3`
  - `set port vlan fe.1.4 2`
  - `set spantree portpri fe.1.2 0 sid 3`
  - `set spantree portpri fe.1.2 240 sid 2`
  - `set spantree portpri fe.1.4 0 sid 2`
  - `set spantree portpri fe.1.4 240 sid 3`
Experiment 14: More VLANs & Link Aggregation

- As seen in previous experiments, Spanning Tree Protocol blocks ports to avoid cycles.
Experiment 14: More VLANs & Link Aggregation

Monaco, Pagano, Passariello, Piacenza, Pintus
Experiments on Virtual Lans and the Spanning Tree Protocol Performed on Enterasys Equipments
Experiment 14:
More VLANs & Link Aggregation

Bridge B:
192.168.1.1
Danny
Jimmy
Port 9
Port 10
Port 1
Port 2
Port 3
Port 4

Monaco, Pagano, Passariello, Piacenza, Pintus
Experiments on Virtual Lans and the Spanning Tree Protocol Performed on Enterasys Equipments
Experiment 14:
More VLANs & Link Aggregation

• What if we enable Link Aggregation Control Protocol?
Experiment 14: More VLANs & Link Aggregation

LACP enabled STP 802.1s

- STP Instance 2
- STP Instance 3

Monaco, Pagano, Passariello, Piacenza, Pintus
Experiments on Virtual Lans and the Spanning Tree Protocol Performed on Enterasys Equipments
Experiment 14:
More VLANs & Link Aggregation

• Commands:
  – set lacp enable
Experiment 14: More VLANs & Link Aggregation

• As seen in the latest experiment, by default LACP aggregates every link between the switches into lag.0.1 virtual port

• We must create a new logical port to separate links: we put together ports 1 and 2 on lag.0.1, and ports 3 and 4 on lag.0.2
Experiment 14:
More VLANs & Link Aggregation

LACP enabled
STP 802.1s
- STP Instance 2
- STP Instance 3

Monaco, Pagano, Passariello, Piacenza, Pintus
Experiments on Virtual Lans and the Spanning Tree Protocol Performed on Enterasys Equipments
Experiment 14: More VLANs & Link Aggregation

• Commands:
  – set lacp static lag.0.2 fe.1.3
  – set lacp static lag.0.2 fe.1.4
  – set lacp static lag.0.1 fe.1.1
  – set lacp static lag.0.1 fe.1.2
Experiment 14:
More VLANs & Link Aggregation

• As seen in the latest experiment to allow communication between hosts on the same VLAN we must associate the logical port to the VLAN.
Experiment 14:
More VLANs & Link Aggregation

LACP enabled
STP 802.1s
- STP Instance 2
- STP Instance 3
Experiment 14: More VLANs & Link Aggregation

• Commands:
  – set port vlan lag.0.1 3
  – set port vlan lag.0.2 2
  – set spantree portpri lag.0.1 0 sid 3
  – set spantree portpri lag.0.1 240 sid 2
  – set spantree portpri lag.0.2 0 sid 2
  – set spantree portpri lag.0.2 240 sid 3
Experiment 14: More VLANs & Link Aggregation

- The logical ports behave like physical ports: we must set their priority for Multiple Spanning Tree Protocol

- By changing port priority for each Spanning Tree instance even though we still have some ports in blocking, those are different for each STI

- This way the hosts can communicate
Experiment 14: More VLANs & Link Aggregation

<table>
<thead>
<tr>
<th>SID</th>
<th>Port</th>
<th>State</th>
<th>Role</th>
<th>Cost</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>fe.1.1</td>
<td>Discarding</td>
<td>Disabled</td>
<td>0</td>
<td>240</td>
</tr>
<tr>
<td>2</td>
<td>fe.1.2</td>
<td>Discarding</td>
<td>Disabled</td>
<td>0</td>
<td>240</td>
</tr>
<tr>
<td>2</td>
<td>fe.1.3</td>
<td>Discarding</td>
<td>Disabled</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>fe.1.4</td>
<td>Discarding</td>
<td>Disabled</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>lag.0.1</td>
<td>Discarding</td>
<td>Alternate</td>
<td>10000</td>
<td>240</td>
</tr>
<tr>
<td>2</td>
<td>lag.0.2</td>
<td>Forwarding</td>
<td>Root</td>
<td>10000</td>
<td>0</td>
</tr>
</tbody>
</table>

LACP enabled STP 802.1s

STP Instance 2

STP Instance 3
Experiment 14:
More VLANs & Link Aggregation

<table>
<thead>
<tr>
<th>SID</th>
<th>Port</th>
<th>State</th>
<th>Role</th>
<th>Cost</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>fe.1.1</td>
<td>Discarding</td>
<td>Disabled</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>fe.1.2</td>
<td>Discarding</td>
<td>Disabled</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>fe.1.3</td>
<td>Discarding</td>
<td>Disabled</td>
<td>0</td>
<td>240</td>
</tr>
<tr>
<td>3</td>
<td>fe.1.4</td>
<td>Discarding</td>
<td>Disabled</td>
<td>0</td>
<td>240</td>
</tr>
<tr>
<td>3</td>
<td>lag.0.1</td>
<td>Forwarding</td>
<td>Root</td>
<td>10000</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>lag.0.2</td>
<td>Discarding</td>
<td>Alternate</td>
<td>10000</td>
<td>240</td>
</tr>
</tbody>
</table>
Experiment 14:
More VLANs & Link Aggregation

Danny@host:~$ ping 192.168.1.10
PING 192.168.1.10 (192.168.1.10) 56(84) bytes of data.
64 bytes from 192.168.1.10: icmp_seq=1 ttl=64 time=22.5 ms
64 bytes from 192.168.1.10: icmp_seq=2 ttl=64 time=0.167 ms
64 bytes from 192.168.1.10: icmp_seq=3 ttl=64 time=0.174 ms
64 bytes from 192.168.1.10: icmp_seq=4 ttl=64 time=0.164 ms
64 bytes from 192.168.1.10: icmp_seq=5 ttl=64 time=0.180 ms
64 bytes from 192.168.1.10: icmp_seq=6 ttl=64 time=0.158 ms

Jimmy@host:~$ ping 192.168.1.14
PING 192.168.1.14 (192.168.1.14) 56(84) bytes of data.
64 bytes from 192.168.1.14: icmp_seq=1 ttl=64 time=22.5 ms
64 bytes from 192.168.1.14: icmp_seq=2 ttl=64 time=0.167 ms
64 bytes from 192.168.1.14: icmp_seq=3 ttl=64 time=0.174 ms
64 bytes from 192.168.1.14: icmp_seq=4 ttl=64 time=0.164 ms
64 bytes from 192.168.1.14: icmp_seq=5 ttl=64 time=0.180 ms
64 bytes from 192.168.1.14: icmp_seq=6 ttl=64 time=0.158 ms
Appendix:
Commands Explain

• The following slides explain each command with its relative usage and options
Appendix: Commands Explain

set ip address <ip-address> mask <netmask>

Command: set ip  Set host interface properties.

Object: address  Set the IP address.

Usage: set ip address <ip_address> [mask <ip_mask>] [gateway <ip_gateway>]

ip_address  IP address to be assigned to the host
mask  The network mask (dotted decimal notation)
gateway  IP address of the next-hop router
Appendix: Commands Explain

set vlan create <vlan-list>

Command: create a new VLAN

Object: vlan-list Set the VLANs.
Usage: set vlan create <vlan-list>

<vlan-list> VLAN ID or range of IDs (1-4093)
Appendix: Commands Explain

set vlan name <vlan-list> <name>

Command: set the name for the specified VLAN.

Object: address Set the name of the VLAN.

Usage: set vlan name <vlan-list> <name>

<vlan-list> 802.1Q VLAN index [1..4093] to assign.

<name> the text name of the VLAN
Appendix: Commands Explain

set port vlan <port-string> <pvid>

Command: 802.1Q VLAN/port association.

Usage: set port vlan <port-string> <pvid>

<pport-string> Port or range of ports
<pvid> 802.1Q VLAN index [1..4093] to assign.
Appendix: Commands Explain

set vlan egress <vlan-list> <port-string> <option>

Command: The port listing for this VLAN.

Usage: set vlan egress <vlan-list> <port-string> <option>

<vlan-list> 802.1Q VLAN index [1..4093] to assign
[port-string] Port or range of ports (default: all ports)
<option>
  untagged VLAN untagged egress.
  forbidden VLAN forbidden egress.
  tagged VLAN tagged egress.
Appendix: Commands Explain

**set spantree priority <priority> <sid>**

*Command:* Use the *set spantree* command to enable or disable the Spanning Tree algorithm or to modify Spanning Tree parameters. Set the bridge priority on a *sid*.

*Object:* `spantree` modify Spanning Tree parameters
*Object:* `priority` set the bridge priority on a *sid*

*Usage:* `set spantree priority <priority> <sid>vlan name <vlan-list>`

- `<priority>` Enter an integer in the range of 0 - 61440.
- `<sid>` Spanning tree instance
Appendix: Commands Explain

**set spantree mstcfgid cfgname <cfgname>**

*Command: set spantree mstcfgid cfgname*
Set the MST Configuration Identifier name

*Object: mstcfgid*  
Set the MST Configuration Identifier elements

*Object: cfgname*  
Configuration name

*Usage: set spantree mstcfgid cfgname <cfgname>*

*<cfgname>*  
Configuration name
Appendix: Commands Explain

**set spantree msti sid <sid> create**

**Command:** `set spantree msti`  
set the status to create or delete a spanning tree instance.

**Object:** `sid`  
Spanning tree instance

**Usage:** `set spantree msti sid <sid> [create | delete]`

- `create`  
creates an instance of spanning tree

- `delete`  
deletes an instance of spanning tree
Appendix: Commands Explain

**set spantree mstimap <fid-num> sid <sid>**

**Command: set spantree mstimap <fid-num> sid <sid>**

**Object: mstmap**  Set the assignment of a FID to a Spanning Tree Instance

**Usage: set spantree mstimap <fif-num> sid <sid>**

<table>
<thead>
<tr>
<th>&lt;fid-num&gt;</th>
<th>FID or range of FIDs (2-4093)</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;sid&gt;</td>
<td>Spanning tree instance</td>
</tr>
</tbody>
</table>
Appendix: Commands Explain

`set spantree portpri <port-string> <portpri> sid <sid>`

Command: `set spantree portpri` Set the port priority on a port and SID

Object: `portpri`

Usage: `set spantree portpri <port-string> <priority> [sid <sid>]`

- `<port-string>` Port or range of ports
- `<priority>` The port priority, in increments of 16 (in decimal) range 0 - 240
- `<sid>` Spanning tree instance
Appendix: Commands Explain

*set spantree version*

**Command:** use the `set spantree` command to enable or disable the spanning tree algorithm or to modify spanning tree parameters.

**Object:** version  
Set the version of spanning tree protocol.

**Usage:** `set spantree version <version>`

- `mstp`  
  enable IEEE802.1s
- `rstp`  
  enable IEEE802.1w
- `stpcompatible`  
  enable IEEE802.1d
# Appendix: Commands Explain

## `set lacp`

<table>
<thead>
<tr>
<th>Command: <code>set lacp</code></th>
<th>Set lacp parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Object:</strong> <code>aadminkey</code></td>
<td>Actor Admin Key</td>
</tr>
<tr>
<td><strong>Object:</strong> <code>asyspri</code></td>
<td>Actor System Priority</td>
</tr>
<tr>
<td><strong>Object:</strong> <code>disable</code></td>
<td>Disables LACP processing</td>
</tr>
<tr>
<td><strong>Object:</strong> <code>enable</code></td>
<td>Enables LACP processing</td>
</tr>
<tr>
<td><strong>Object:</strong> <code>singleportlag</code></td>
<td>Formation of single port lags</td>
</tr>
<tr>
<td><strong>Object:</strong> <code>static</code></td>
<td>Assign specific ports to a lag</td>
</tr>
</tbody>
</table>
Appendix: Commands Explain

`set lACP static <lagPortString> <port-string>`

**Command:** static  
**Assign specific ports to a lag**

**Object:** `<lagPortString>`  
**The LAG port to assign values to (example: lag.0.1)**

**Object:** `<port-string>`  
**Port or range of ports**
Appendix: Commands Explain

**set port ingress-filter <port-string> [...]**

Command: set port ingress-filter  Limits the forwarding of received frames based on port VLAN egress lists.

Object: disable  Disable the port ingress filter function on the ports specified in the port-string.

Object: enable  Enable the port ingress filter function on the ports specified in the port-string.

Usage: set port ingress-filter <port-string> [enable | disable]

<port-string> Specifies the port(s) to be added to the device's port ingress list.
Appendix: Commands Explain

**set port discard** <port-string>

*Command:* Discard specific frames on a (group of) port(s).

*Usage:* `set port discard <port-string> [both | none | tagged | untagged]`

*<port-string>* Port or range of ports (default: all ports)

*Object: both* Discard tagged and untagged frame types.

*Object: none* Discard none.

*Object: tagged* Discard tagged frame types.

*Object: untagged* Discard untagged frame types.
Appendix: Commands Explain

show vlan

Command: show VLAN properties.

Object: portinfo  Show VLAN attributes related to one or more ports.
Object: dynamicegress  Show status of dynamic egress for listed VLAN.
Object: static  Display only static VLANs
Object: <vlan-list>  Enter VLAN-list
Appendix: Commands Explain

*show spantree stats*

**Command:** show the spanning tree info per port / sid

**Usage:** show spantree stats [active] [port <port-string>] [sid <sid>]

- **active**
  - show only the ports that have received at least one config or rstp bpdu

- **port**
  - port or range of ports

- **sid**
  - spanning tree instance
Appendix: Commands Explain

*show lACP*

**Command:** show lACP information

**Usage:** show lACP [<port-string>]

- `<port-string>` specifies the lag port(s) to display