2.1 Introduction

Spanning Tree Protocol (STP) is a Layer 2 protocol that runs on bridges and switches. The specification for STP is IEEE 802.1D. The main purpose of STP is to ensure that you do not create loops when you have redundant paths in your network. Loops are deadly to a network.

STP PortFast causes a Layer 2 LAN interface configured as an access port to enter the forwarding state immediately, bypassing the listening and learning states. You can use Port Fast on Layer 2 access ports connected to a single workstation or server to allow those devices to connect to the network immediately, instead of waiting for STP to converge[8,9]. Interfaces connected to a single workstation or server should not receive bridge protocol data units (BPDUs). When configured for PortFast, a port[12,16] is still running the spanning tree protocol. A PortFast enabled port can immediately transition to the blocking state if necessary (this could happen on receipt of a superior BPDU).

When enabled on a port, BPDU Guard shuts down a port that receives a BPDU. When configured globally, BPDU Guard is only effective on ports in the operational PortFast state. In a valid configuration, PortFast Layer 2 LAN interfaces do not receive BPDUs. Reception of a BPDU by a PortFast Layer 2 LAN interface signals an invalid configuration, such as connection of an unauthorized device[2,3,6]. BPDU Guard provides a secure response to invalid configurations, because the administrator must manually put the Layer 2 LAN interface back in service. With release 12.1(11b)E, BPDU Guard can also be configured at the interface level. When configured at the interface level, BPDU Guard shuts the port down as soon as the port receives a BPDU, regardless of the Port Fast configuration.

When configured globally, Port Fast BPDU filtering applies to all operational PortFast ports[13,14]. Ports in an operational Port Fast state are supposed to be connected to hosts, that typically drop BPDUs. If an operational Port Fast port receives a BPDU, it immediately loses its operational Port Fast status. In that case, Port Fast BPDU filtering is disabled on this port and STP resumes sending BPDUs on this port.

Port Fast BPDU filtering can also be configured on a per-port basis. When Port Fast BPDU filtering[7,8] is explicitly configured on a port, it does not send any BPDUs and drops all BPDUs it receives.
Uplink Fast provides fast convergence after a direct link failure and achieves load balancing between redundant Layer 2 links using uplink groups. An uplink group is a set of Layer 2 LAN interfaces (per VLAN), only one of which is forwarding at any given time. Specifically, an uplink group consists of the root port (which is forwarding) and a set of blocked ports, except for self-looping ports[9,10]. The uplink group provides an alternate path in case the currently forwarding link fails.

Backbone Fast is initiated when a root port or blocked port on a network device receives inferior BPDU's from its designated bridge. An inferior BPDU identifies one network device as both the root bridge and the designated bridge. When a network device receives an inferior BPDU, it indicates that a link to which the network device is not directly connected (an indirect link) has failed (that is, the designated bridge has lost its connection to the root bridge). Under normal STP rules, the network device ignores inferior BPDU's for the configured maximum aging time, as specified by the STP max-age command.

The network device tries to determine if it has an alternate path to the root bridge. If the inferior BPDU arrives on a blocked port, the root port and other blocked ports on the network device become alternate paths to the root bridge. (Self-looped ports are not considered alternate paths to the root bridge.) If the inferior BPDU arrives on the root port, all blocked ports become alternate paths to the root bridge. If the inferior BPDU arrives on the root port and there are no blocked ports, the network device assumes that it has lost connectivity to the root bridge, causes the maximum aging time on the root to expire, and becomes the root bridge according to normal STP rules.

If the network device has alternate paths to the root bridge, it uses these alternate paths to transmit a new kind of Protocol Data Unit (PDU) called the Root Link Query PDU. The network device sends the Root Link Query PDU out all alternate paths to the root bridge. If the network device determines that it still has an alternate path to the root, it causes the maximum aging time to expire on the ports on which it received the inferior BPDU. If all the alternate paths to the root bridge indicate that the network device has lost connectivity to the root bridge, the network device causes the maximum aging times on the ports on which it received an inferior BPDU to expire. If one or more alternate paths can still connect to the root bridge, the network device makes all ports on which it received an inferior BPDU its designated
ports and moves them out of the blocking state (if they were in the blocking state), through the listening and learning states, and into the forwarding state.

Ether Channel guard detects a misconfigured Ether Channel where interfaces on the Catalyst 6500 series switch are configured as an Ether Channel while interfaces on the other device are not or not all the interfaces on the other device are in the same Ether Channel.

In response to misconfiguration detected on the other device, Ether Channel guard puts interfaces on the Catalyst 6500 series switch into the err disabled state.

### 2.2 Motivation

The wide deployment of Layer 2/Layer 3 switches in the wiring closet and distribution layers of the campus and enterprise networks has cause lot of dependency on the Spanning Tree Protocol to ensure loop free networks. To define, spanning tree is a protocol designed to eliminate loops in a topology having redundant links.

To overcome the above limitation of existing measurement techniques, we propose an accurate and non redundant framework called spanning tree routing and switching architecture (STPRSA). First STPRSA consist of complementary measurement schemes like avoid redundancy, reestablishment in change in topology through channeling and journal system. Finally STPRSA is designed to run in a fully distributed fashion on existing IEEE 802.1D. We conduct an in depth evaluation of STPRSA in C&C++ Language and experimentation on a Linux based implementation. Our results shows that STPRSA unicast based techniques decrease the time complexity in measurements over the broadcast based approach, moreover STPRSA direction aware link quality measurement enables the opportunistic use of asymmetric links and helps the underline routing protocol finds the best quality and thus improve the channel efficiency up to 89.8%.

The rest of this chapter is organized as follows. Section 2.2 describes the motivation of this work. Section 2.3 presents the STPRSA architecture and algorithm and section 2.4 evaluates our implementation and experimental results. Section 2.5 discuss the remaining issues associated with STPRSA, and finally concludes the chapter.
2.2.1 Why spanning tree protocol

STP runs on bridges and switches that are 802.1D-compliant. There are different flavors of STP, but 802.1D is the most popular and widely implemented way we implement STP on bridges and switches in order to prevent loops in the network. Using STP in situations where you want redundant links, but not loops. Redundant links are as important as backups in the case of a failover in a network. A failure of your primary activates the backup links so that users can continue to use the network. Without STP on the bridges and switches, such a failure can result in a loop. If two connected switches run different flavors of STP, they require different timings to converge. When different flavors are used in the switches, it creates timing issues between Blocking and Forwarding states. Therefore, it is recommended to use the same flavors of STP.

2.2.1a Rules of Operation

This section lists rules for how STP works. When the switches first come up, they start the root switch selection process. Each switch transmits a BPDU to the directly connected switch on a per-VLAN basis.

As the BPDU goes out through the network, each switch compares the BPDU that the switch sends to the BPDU that the switch receives from the neighbors. The switches then agree on which switch the root switch is. The switch with the lowest bridge ID in the network wins this election process.

Remember that one root switch is identified per-VLAN. After the root switch identification, the switches adhere to these rules:

- **STP Rule 1**—all ports of the root switch must be in forwarding mode.
  
  Next, each switch determines the best path to get to the root. The switches determine this path by a comparison of the information in all the BPDUs that the switches receive on all ports. The switch uses the port with the least amount of information in the BPDU in order to get to the root switch; the port with the least amount of information in the BPDU is the root port. After a switch determines the root port, the switch proceeds to rule 2.

- **STP Rule 2**—the root port must be set to forwarding mode.
  
  In addition, the switches on each LAN segment communicate with each other to determine which switch is best to use in order to move data from that segment to the root bridge. This switch is called the designated switch.
• **STP Rule 3**—in a single LAN segment, the port of the designated switch that connects to that LAN segment must be placed in forwarding mode.

• **STP Rule 4**—all the other ports in all the switches (VLAN-specific) must be placed in blocking mode. The rule only applies to ports that connect to other bridges or switches. STP does not affect ports that connect to workstations or PCs. These ports remain forwarded.

The addition or removal of VLANs when STP runs in per-VLAN spanning tree (PVST / PVST+) mode triggers spanning tree recalculation for that VLAN instance and the traffic is disrupted only for that VLAN. The other VLAN parts of a trunk link can forward traffic normally. The addition or removal of VLANs for a Multiple Spanning Tree (MST) instance that exists triggers spanning tree recalculation for that instance and traffic is disrupted for all the VLAN parts of that MST instance.

By default, spanning tree runs on every port. The spanning tree feature cannot be turned off in switches on a per-port basis. Although it is not recommended, you can turn off STP on a per-VLAN basis, or globally on the switch. Extreme care should be taken whenever you disable spanning tree because this creates Layer 2 loops within the network.

### 2.2.2 Limitations of Existing Techniques

The wide deployment of Layer 2/Layer 3 switches in the wiring closet and distribution layers of the campus and enterprise networks has cause lot of dependency on the Spanning Tree Protocol to ensure loop free networks. To define, spanning tree is a protocol designed to eliminate loops in a topology having redundant links.

This tries to explore the trivial and widely used STP to introduce two enhancements, which can help in improving performance of the network in terms of convergence time and better link utilization.

With the introduction of the concept of Spanning Tree Channels, the backbone switches can be configured to make use of the links which are added for redundancy, but are not used when primary link is up. This increases load balancing and overall network speeds.

The Spanning Tree Zone concept is mainly for huge clusters of switches which have spanning tree configured on them. These take high convergence times, even to address a link failure or topology change at one end. By dividing into zones,
the convergence time and recalculation overhead is considerably reduced and is not felt across the network.

2.2.2a Avoid Redundancy

Consider this network:

![Figure 2.1: Two stations with the redundant link between switch A and switch B](image)

In this network, a redundant link is planned between Switch A and Switch B. However, this setup creates the possibility of a bridging loop. For example, a broadcast or multicast packet that transmits from Station M and is destined for Station N simply continues to circulate between both switches.

2.2.2b Reestablishment in change of topology

For networks having huge number or layer 2 devices, the convergence time is often not acceptable, just for any minor topology change. For instance, consider a network as in Figure 2, for a change/toggle in link, say X (marked in fig) the TCN exchange and then stabilizing the network to be safe for traffic forwarding would take atleast 30-60 secs.

![Figure 2.2: TCN Exchange for stabilizing the network](image)
To solve such issues in larger networks, we introduce the concept of division into zones or areas. This is similar to what routing protocols like OSPF use. The administrator has the task of dividing the network to zones. Each zone will have a primary root based on selection criteria similar to selecting root in a combined network.

Consider a zone to be comprising of switches E, F and G. Assume E is the root here. E now has the responsibility of talking (directly or indirectly) to the outside world. What we achieve out of this is that topology changes or fluctuations within a zone, like say switches A, B, C and D will be contained and resolved without putting the whole stream of switches downstream on discarding or blocking state.

### 2.2.2c Prioritizing merging across the wire

This step tries to merge the connection in different connection/switch box to maximize resource sharing. The sharing of tracks will lead to a reduction in the number of switches used. A merging process in which the most timing critical net is processed first and the least timing-critical net last is performed repeatedly for the nets. The possible merging of subnets is then performed from the subnet containing the source terminal toward the subnets containing the sink terminals.

The below figure depicts the merging process.

![Routing results considering an implementation for net 'a' with tracks](image-url)
2.2.2d Making link and routing paths more efficient

Routers and Forwarders in a Router domain would likely have different amount of hardware Resources. The least capable switch should not hold the performance of the Router network domain to ransom. This write up lists some smart choices the forwarders and routers in the Router domain can make to most efficiently utilize the hardware forwarding capabilities of each node in the network.

In a Router domain, all forwarders and routers belonging to a single Router domain keep every other Forwarder or router updated of all the host routes for each host known and present in the network.

The above is done for every subnet in the Router domain.

Each forwarder and router installs all the routes in the fast path forwarding database (for example hardware forwarding tables).
In this new scheme, all the routers and forwarders learn all the routes, build the topology table for all subnets, but the difference is the router/forwarder can choose to put subnet of the routes in the fast path database (hardware database). This would save of the hardware resources, without sacrificing the network performance in most cases. In the cases where network path chosen is potentially suboptimal a further set of enhancements comes and loosens the optimization to further improve the network performance without wasting the forwarding resource utilization.

Both algorithms found equally good solutions in most of the cases, as demonstrated by our preliminary experiments. However, this adaptive algorithm is more robust and stable than the greedy one. The algorithm based on evolutionary computing also demonstrated a better calculation time, although it had a little time overhead on small applications. As can be seen, this algorithm has a success ratio bigger than 18% for large applications.

The fact that the success ratio of this algorithm increases with the application size is (or appears to be) due to an increased number of additional constraints imposed by the application components that greedy algorithm cannot handle. The limitation of the current implementation of this algorithm is the resource model that supports only specific types of resources, such as hosts and network link physical constraints. We identify a need to use a generic resource model, where each host can provide any kind of resource. This generic model has to take into use, for example, displays, microphones and other peripheral devices offering additional services for the user. Moreover, in a future implementation of the algorithm.

2.3 Spanning tree routing & switching Architecture

This section details the architecture of STPRSA. First, the design rationals and main algorithm of STPRSA are outlined. Next, we define the link quality of STPRSA which deals with describing all the measurement schemes. STPRSA describes the solutions for first two limitations and other two are covered in chap 3 & 4. Finally, we analyze the complexity of STPRSA.

Spanning trees are a crucial asset and are evolving toward architectures in which networks, computer systems, and storage devices act in unison. Data centers of the future will rely heavily on an architecture in which a comprehensive system provides performance and cost advantage. As IT organizations migrate from fragmented, older data centers to more cost-effective and agile ones, they will need to
Spanning Tree Protocol Optimization

consider the requirements of data center consolidation, business continuance, branch consolidation, virtualization, and application optimization. Next-generation data centers will require a sound architecture as a building block and mandate that this architecture reduce the total cost of ownership and deliver easy manageability.

Spanning tree Switching and Routing Architecture (STPRSA) Assessment analyzes technical requirements and customer's existing data center environment, technical requirements. Deliverables include findings and recommendations. The assessment also provides a more long-term data center architecture blueprint and presents the next steps in this evolution. The assessment covers topics such as STPRSA architecture and best practices, IP infrastructure, storage, data center, hardware forwarding application optimization, business continuity, operations excellence, virtualization, and service-oriented network architecture (SONA).

STPRSA architects have direct experience in all phases of the planning, design, implementation, operation, and optimization of the infrastructure of STP.

2.3.1. Overview of STPRSA

The process of forwarding a frame from a source to the direction of its destination is a two-step process. First step is learning, and the second is forwarding of the frame. If the destination address is not known, the frame will be forwarded to all ports (except the port from where it arrived).

Figure 2.6: Overall architecture of spanning tree routing & switching architecture.

Department of Computer Science, S.K University, Anantapur.
The above figure shows STPRSA has three phases, at first optimization is done at STP server, secondly it is done by UIMR & HFRD techniques (chap 3&4), Thirdly by TSR (chap 5), where in we called the whole architecture as Integrated automated router system.

This summarizes the topics and potential benefits of the spanning tree architecture Assessment Topics and Benefits.

<table>
<thead>
<tr>
<th>Architecture Assessment Topics</th>
<th>Potential Benefits</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Spanning tree Data architecture</strong></td>
<td>• Align the data center infrastructure to business needs (for example, increased business agility).</td>
</tr>
<tr>
<td></td>
<td>• Move toward a service-oriented data center infrastructure.</td>
</tr>
<tr>
<td></td>
<td>• Align IT operations with IT services and related processes (IT Infrastructure Library [ITIL]).</td>
</tr>
<tr>
<td></td>
<td>• Improve use of data center resources.</td>
</tr>
<tr>
<td></td>
<td>• Achieve cost savings through data center consolidation.</td>
</tr>
<tr>
<td></td>
<td>• Improve the performance of the current data center infrastructure.</td>
</tr>
<tr>
<td></td>
<td>• Increase the resiliency of the data center.</td>
</tr>
<tr>
<td></td>
<td>• Use IOS® Software High Availability features.</td>
</tr>
<tr>
<td></td>
<td>• Increase performance of the VPN remote-access service.</td>
</tr>
<tr>
<td>• Service-oriented data center</td>
<td></td>
</tr>
<tr>
<td>• Business drivers and alignment of DCN infrastructure to them (for example, promote business agility)</td>
<td></td>
</tr>
<tr>
<td>• Maturity of IT services</td>
<td></td>
</tr>
<tr>
<td>• Migration from a fragmented application and server approach to SONA</td>
<td></td>
</tr>
<tr>
<td>• Evolution of data center architecture</td>
<td></td>
</tr>
<tr>
<td>• Data center consolidation</td>
<td></td>
</tr>
<tr>
<td>• Reduced number of data centers</td>
<td></td>
</tr>
<tr>
<td>• Storage consolidation</td>
<td></td>
</tr>
<tr>
<td>• Server and application consolidation</td>
<td></td>
</tr>
<tr>
<td>• Integration of services on the Cisco Catalyst® platform</td>
<td></td>
</tr>
<tr>
<td>• Data center high availability</td>
<td></td>
</tr>
<tr>
<td>• Server high availability</td>
<td></td>
</tr>
<tr>
<td>• Fault tolerance in the data center</td>
<td></td>
</tr>
<tr>
<td>• IOS® Software High Availability features (nonstop forwarding [NSF] and stateful switchover [SSO])</td>
<td></td>
</tr>
<tr>
<td>• Other data center architecture topics</td>
<td></td>
</tr>
<tr>
<td>• Data center enterprise edge</td>
<td></td>
</tr>
<tr>
<td>• Unsecured network area design concerns</td>
<td></td>
</tr>
<tr>
<td>• Teleworker, extranet, and VPN architecture</td>
<td></td>
</tr>
</tbody>
</table>

| **Spanning tree Data infrastructure** | |
| • Data center server farm architecture | • Increase the availability of the data center IP infrastructure. |
| • Shared application and security services | • Improve the resiliency of the data center IP infrastructure. |
| • Server-to-server communications | • Efficiently use cabling and rack |
Spanning Tree Protocol Optimization

| Network interface card (NIC) teaming requirements | resources. |
| Blade server connectivity | - Avoid outages because of spanning tree design problems. |
| Data center server farm design | - Select the appropriate spanning tree protocols. |
| DCN Layer 2 and Layer 3 design | |
| DCN access, aggregation, and core design | |
| Spanning tree design and scalability | |
| Selection of spanning tree protocol | |
| Routing and cabinet design, cabling, and density considerations | |
| Server farm cabinet layout | |
| Cabling topics | |

**Storage**

| Storage area network (SAN) consolidation | - Consolidate SAN islands to decrease costs. |
| Performance and scalability | - Increase use of the SAN infrastructure. |
| Migration topics | - Reduce costs with the latest IP SAN technologies. |
| SAN island consolidation | - Define a clear SAN architecture plan and verify the implementation steps. |
| SAN security | - Analyze SAN security and plan steps to enhance it. |
| Data integrity and encryption | - Improve SAN management and optimize SAN performance. |
| Device authorization and traffic isolation | - Use intelligent SAN services |
| SAN management | |
| Management of changes | |
| SAN performance optimization | |
| SAN extension | |
| IP SAN | |
| Intelligent SAN services | |

**Application optimization**

| Essential enterprise services (Domain Name System [DNS] and Dynamic Host Configuration Protocol [DHCP]) | - Improve application performance. |
| Server and application load balancing | - Enhance end-user application response times. |
| Server resource offloading | - Increase availability through server and application load balancing. |
| SSL offloading and application acceleration | |
| Application security | |
| File and software distribution | |

Spanning tree is needed in networks with redundancy, where undesirable loops can form.
To provide path redundancy, STP defines a tree that spans all switches in an extended network. STP forces certain redundant data paths into a standby (blocked) state. If one network segment in the Spanning Tree Protocol becomes unreachable, or if Spanning Tree Protocol costs change, the spanning tree algorithm reconfigures the spanning-tree topology and reestablishes the link by activating the standby path.

There are further intricacies involved, like election of root, and propagation of change in topology etc, but describing them would be beyond the scope of this document.

2.3.2 Spanning tree channeling System

1) The Spanning tree protocol ensures a loop free network, but at a cost. The cost is that the redundant link will be blocked, and no traffic can flow through it. When network users buy a chassis being charged on a per port basis, this becomes considerable cost. It could be more serious for core or central layer 2 switches as it is important to have redundant links between such chassis. Consider fig 1, where there are two paths existing for traffic to flow from B to C. With STP active, only path via root will be open.

Although loop is avoided, this clearly has two disadvantages.

a. Assume 1 Gb links connecting each of the switches, from B to C, effectively there is 2 Gbps bandwidth. But only 1Gb is practically available.

b. To reach B from C, the shorter path is directly from B to C, but being blocked, the path B-A-C has to be taken.
As solution, we introduce the concept of channeling among such redundant links. This is to be configured by the administrator, preferably over the high traffic zones, backbone switches. The channel will be formed typically between switches B and C non root devices here and will tell the software that there exists a redundant path, one directly and another via the root, A.

The administrator can either configure a traffic threshold, beyond which he might want the switch to use the redundant path (B to C).

The channel will consider the link B to C and B-A-C as one bundle, sharing common properties.

Alternative solution used in field now is to make the switches A, B and C as roots for different VLANs, so that different sets of links become as blocking and hence load balancing to an extent is achieved. But then if one VLAN is known to send more traffic than other or if we have just 1 or 2 VLANs, the problem is not effectively solved.

2.3.2a Detailed solution

Given a network as shown in Figure 2.7, or say more complex networks as well like Figure 2.8a, we visualize the network as a channel, with one direct link and another virtual link. See Figure 2.8b.
Spanning Tree Protocol Optimization

Figure 2.8: (a) & (b) shows three DUT setup to see the detailed working. Here, the administrator configures ports between non-root switches as parts of channel.

Assume there was no spanning tree. Host 1 sends traffic. Switch A will flood the packets to B and C. If the destination is known to B or C, it will be accordingly forwarded instead of flooding. But in the process, the address MAC 1 will be learnt at the port through which it arrived at B and C. If the destination is not reached, the frames will further be flooded, and B will receive it from C as well (and vice versa). So B, for example, now has 2 entries for host 1, reachable via Switch A and Switch C, and packets will this way get flooded again.

To prevent this, spanning tree blocks the link between B and C. Now, let us get channeling solution into the picture.

With channeling, the switch software knows that it can expect two ways to reach a particular destination via spanning tree channel ports. This, the switch has to effectively use by load balancing.

The blocking port here maintains a new state called ‘Blocked Channel’. The main concept is, after the learning phase, the traffic is not forwarded immediately by these ports. The MAC is learnt and stored.
Spanning Tree Protocol Optimization

Avoid path redundancy

Algorithm: STPRSA at node $i$ during $k$-1 terminal nets

- Set the nodes according to the path criticality estimated using the measure
- For each set $n_i$ in the list
  - Find the edges in the net according to edge, path $pi$.
- If (edge, Path $pi$ is not already existing)
  - Perform shortest path from signal source.
  - Select the $k$-1 shortest two terminal nets whose route match with the signal flow of $ni$.
- Else
  - Assign edge-path $pi$ to particular pair of terminals
  - Construct a minimum spanning tree
  - Generate two terminal net for each edge of the spanning tree.
- End if;
- End for.

As next step, all traffic received from other ports of B will be forwarded to both, C as well as towards root. C's responsibility is that, it will not forward any packets received at the 'blocked channel' port to other port of the channel. Any traffic addressed to destination MACs learnt in non-channel port will be forwarded by C. These MACs will be advertised by C in a separate format to B, mentioning that these are the MACs which can be reached directly via C. Directly here means, not via other channel port. In fig 5, B3 and B4 are directly reachable by B, as seen by C.

This way, we achieve the goal of load balancing and reaching via shortest path when possible. Taking examples, assume that Sw A floods traffic from say, MAC 5 towards B and C. Both B and C will check there CAM entries (both, blocked channel reachable (special format) MACs as well as its connected ones). See fig 2.9
Figure 2.9: Shows medium access control address table, which may directly or indirectly connected.

If C does not advertise this as directly reachable while learning, it will not forward the traffic to C.

Now, assume that MAC 3 is directly connected to C. This is learnt by B using special format frames. Packets from ports B3 or B4 to MAC 3 will be load balanced by B and forwarded via B1 as well as B2. C will now entertain both, i.e. frames received from root as well as from blocked channel port.

2.3.3 Zonal approach in detail

For networks having huge number or layer 2 devices, the convergence time is often not acceptable, just for any minor topology change. For instance, consider a network as in Figure 2.10, for a change/toggle in link, say X (marked in fig) the TCN exchange and then stabilizing the network to be safe for traffic forwarding would take at least 30-60 secs.

The solution to solve such issues in larger networks, we introduce the concept of division into zones or areas. This is similar to what routing protocols like OSPF use. The administrator has the task of dividing the network to zones. Each zone will have a primary root based on selection criteria similar to selecting root in a combined network.
Spanning Tree Protocol Optimization

Consider a zone to be comprising of switches E, F and G. Assume E is the root here. E now has the responsibility of talking (directly or indirectly) to the outside world. What we achieve out of this is that topology changes or fluctuations within a zone, like say switches A, B, C and D will be contained and resolved without putting the whole stream of switches downstream on discarding or blocking state.

**Limitation:** This concept however has the limitation that the administrator has to have the additional configuration over the STP blocked ports, and the switch has the overhead of having two sets of MAC addresses. For this second level of optimization, consider the fig 2.11 as below, divided into zones.

Figure 2.10: TCN Exchange for stabilizing the network

Figure 2.11: Describes the clear zonal system, which has been divided into areas in order to reduce the time complexity in channeling
The administrator configures the topology into zones. Spanning tree now elects roots within the zones.

Assume that in zone 1, A becomes the root and E in zone 3. Now, A only has to talk to roots of other zones. Among the roots, again one is elected as primary roots, which as per normal spanning tree BPDU exchange will be the lowest MAC. This way the additional overhead is less. Now, A sees zone 2 and zone 3 as just one device. Spanning tree convergence will result in the port between zone 3 and zone 2 to be blocked, ie E to H, via G. Switches like G here should act as transparent to BPDU exchange between zone roots.

This done, now to reap the benefits! Consider a link breakage as in Y (fig 2.11) which is in forwarding mode among E, F and G. Intra zone spanning tree will immediately activate or move the forwarding the link via F. All this will be transparent to zones 1 and 2, and with less number of switches per zone, convergence is a matter of milliseconds. Similar for a case of topology change.

With the introduction of the concept of Spanning Tree Channels, the backbone switches can be configured to make use of the links which are added for redundancy, but are not used when primary link is up. This increases load balancing and overall network speeds.

The Spanning Tree Zone concept is mainly for huge clusters of switches which have spanning tree configured on them. These take high convergence times, even to address a link failure or topology change at one end. By dividing into zones, the convergence time and recalculation overhead is considerably reduced and is not felt across the network.

2.4 System Implementation and Experimentation

We have implemented STPRSA in linux based systems and evaluated it. We first give the architectural details of the implementation and then describe the experimental setup and finally we present the experimental results.

2.4.1 Implementation Details

The present routing algorithm is implemented using C &C++ . The proposed routing approach is implemented with changes in the scripts as follows:
Spanning Tree Protocol Optimization

(i) **Routing Table**: Every node has its own routing table which stores the information about the various routes. For each destination in the table the corresponding next hop, total hops, and the expiry time are specified. As the expiry time elapses the route gets deleted or updated. In the routing table can be accessed at any time from the tcl script and the routing updation can be studied.

(ii) A new parameter called “**priority**” has been added to the routing table in file to specify the priority to critical nodes. The critical nodes are assigned a priority of 1 while the other nodes are assigned 0.

(iii) **Neighbors Table**: Every node has its own neighbour table which stores all its neighbour ids. The neighbour table is updated using the HELLO broadcast.

(iv) **Zone formation**: The conzone is built in recvRequest(..) function in aodv.cc file. The critical node ids are checked and the function to build conzone is called. When all parents for a level is entered, the value in id_no array is changed to the new set of values. When the level is 3, the parents of current level nodes are statically assigned the id of the sink node.

(v) **Zone Routing**: The high priority packets are routed inside the zone. First, the sendReply(..) function in aodv.cc file is modified in order to block the reply to the critical nodes. After the formation of zone the routing table of each node is accessed from tcl file and checked if its on zone node.

(vi) If the node is on zone then another function is called in rtable.cc which has index, corresponding parent and level as its parameters. This function updates the route entries for the zone nodes.
The above figure represents the step as follows, First identify the zone slices that intersect objects, where each bin corresponds to pre-end, if zone slice bin is not existing ,it adds reference for object to the list, else it adds it to the collision list.

2.4.2 Experimental Setup

To evaluate our implementation we tested on multicore IBM P690SMPS machine with power PC-POWER4 with 32 cores running at 1.7 GHZ with Linux Operating system. The implementation was done with nodes for dense and with Fibonacci heaps with star bus topology For Dense ,we tested our code for various type switch 25,50,100,125,150 nodes, for sparse we tested our code for 2d grids, random connected sparsh graphs.
2.4.3 Experimental Results

Using the above steps of experimental setup, we first show how effectively STPRSA uses nodes with its link speed for measuring efficiency. Then we show that by using STPRSA based approach measures the time more accurately than the STP IEEE802.D approach. Finally we show that STPRSA link quality effectively identifies link symmetry and improves the efficiency of utilizing the channel capacity over time.

**TABLE 2.1**: Time Comparison Of IEEE801.D and STPRSA with link speed 1GBPS

<table>
<thead>
<tr>
<th>Nodes</th>
<th>Link speed</th>
<th>Revised IEEE Spec (ms)</th>
<th>STPRSA (IARS) (ms)</th>
<th>Efficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>1Gbps</td>
<td>4</td>
<td>1.28</td>
<td>2.72</td>
</tr>
<tr>
<td>50</td>
<td>1Gbps</td>
<td>8.3</td>
<td>6.8</td>
<td>1.5</td>
</tr>
<tr>
<td>100</td>
<td>1Gbps</td>
<td>12.8</td>
<td>10.89</td>
<td>1.91</td>
</tr>
<tr>
<td>125</td>
<td>1Gbps</td>
<td>16.6</td>
<td>14.2</td>
<td>2.4</td>
</tr>
<tr>
<td>150</td>
<td>1Gbps</td>
<td>24.0</td>
<td>22.2</td>
<td>1.8</td>
</tr>
</tbody>
</table>
We evaluated the effects of STPRSA approach by measuring the time over nodes. We ran several different number of nodes like 25, 50, 100, 150, then we measured the time (ms) and derived the percentage compared with IEEE & STPRSA. The above graphs plots representative time with different number of nodes.

### TABLE 2.2: Time Comparison Of IEEE802.1D and IARS with link speed 10GBPS

<table>
<thead>
<tr>
<th>Nodes</th>
<th>Link speed</th>
<th>Revised IEEE Spec (ms)</th>
<th>STPRSA(IARS) (ms)</th>
<th>Efficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>10Gbps</td>
<td>1.68</td>
<td>1.08</td>
<td>0.6</td>
</tr>
<tr>
<td>50</td>
<td>10Gbps</td>
<td>2.46</td>
<td>1.98</td>
<td>0.48</td>
</tr>
<tr>
<td>100</td>
<td>10Gbps</td>
<td>4.87</td>
<td>3.92</td>
<td>0.95</td>
</tr>
<tr>
<td>125</td>
<td>10Gbps</td>
<td>6.99</td>
<td>5.98</td>
<td>1.01</td>
</tr>
<tr>
<td>150</td>
<td>10Gbps</td>
<td>7.82</td>
<td>6.65</td>
<td>1.17</td>
</tr>
</tbody>
</table>
While this approach uses a fixed number of packets (i.e., 10) per cycle, Next the percentage of each measurement scheme shown below. By contrast, owing to the use of unicast packets, STPRSA measurement results with the link speed 1 GBPS where on the average 74.6% is the efficiency rate compared to the above specifications.

**TABLE 2: Time Comparison Of IEEE 802.1D and IARS with link speed 10MBPS**

<table>
<thead>
<tr>
<th>Nodes</th>
<th>Link speed</th>
<th>Revised IEEE Spec (ms)</th>
<th>STPRSA(IARS) (ms)</th>
<th>Efficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>10Mbps</td>
<td>8</td>
<td>6.28</td>
<td>1.72</td>
</tr>
<tr>
<td>50</td>
<td>10Mbps</td>
<td>16.6</td>
<td>14.78</td>
<td>1.82</td>
</tr>
<tr>
<td>100</td>
<td>10Mbps</td>
<td>24.8</td>
<td>23.60</td>
<td>1.2</td>
</tr>
<tr>
<td>125</td>
<td>10Mbps</td>
<td>33.4</td>
<td>32.76</td>
<td>0.64</td>
</tr>
<tr>
<td>150</td>
<td>10Mbps</td>
<td>48.01</td>
<td>44.77</td>
<td>3.24</td>
</tr>
</tbody>
</table>
Table 2.2 & 2.3 proves the extensive measurements, we found that wired spanning tree routing & switching architecture have significant link a symmetry, and with the link speed of 10Gbps and 10 Mbps with the same set of nodes make the comparison quite interesting. On the average the efficiency is (>78 and >80) compared to revised specifications of IEEE.

**TABLE2.4: Time Comparison Of IEEE and IARS with link speed 100MBPS**

<table>
<thead>
<tr>
<th>Nodes</th>
<th>Link speed</th>
<th>Revised IEEE Spec (ms)</th>
<th>STPRSA(IARS) (ms)</th>
<th>Efficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>100Mbps</td>
<td>5.22</td>
<td>4.27</td>
<td>0.95</td>
</tr>
<tr>
<td>50</td>
<td>100Mbps</td>
<td>12.7</td>
<td>11.30</td>
<td>1.40</td>
</tr>
<tr>
<td>100</td>
<td>100Mbps</td>
<td>23.89</td>
<td>22.65</td>
<td>1.24</td>
</tr>
<tr>
<td>125</td>
<td>100Mbps</td>
<td>30.55</td>
<td>28.46</td>
<td>2.09</td>
</tr>
<tr>
<td>150</td>
<td>100Mbps</td>
<td>41.23</td>
<td>39.91</td>
<td>1.32</td>
</tr>
</tbody>
</table>
Table 2.4 shows with and number of nodes the extensive measurements, we found that wired spanning tree routing & switching architecture have significant link a symmetry, and with the link speed of 100mbps also make the comparison quite interesting. On the average the efficiency is >80% compared to revised specifications of IEEE.

2.5 Conclusion

We first discuss some of the remaining issues associated with STPRSA and then make concluding remarks.

2.5.1 Remaining Issues

Link-quality information: Although this chapter focused on avoiding redundancy and makes the flow efficient in STP, storage & hardware forwarding resources in an equally important problem. Broadcast-based sequenced flooding is one popular solution to this problem in small networks. There are also a couple of well-known approaches to the dissemination problem in STP. However, the information in wired networks has several challenges to overcome, including scalability and fault-tolerance. We will address these issues in a separate forthcoming work.

Measuring other link-quality parameters: In this work, the link speed ratio and time rate—suitable for high-throughput metrics—are considered as the link-quality
parameters. However, QoS parameters, such as net delay and path delay, should be measured to support real-time applications. These parameters can be accurately measured by STPRSA, based on MIB and NIC buffer clearing time. Thus, along with the high-throughput parameters, STPRSA can support such applications as VoIP and IPTV that use the time-related parameters.

2.5.2 Concluding remarks

In this chapter, we have presented a novel link-quality measurement framework, called STPRSA, for wired networks. STPRSA is composed of three complementary measurement techniques—Avoiding redundancy using channeling, zonal system, merging—which minimize the probing overhead and provide highly accurate link-quality information by exploiting each node’s egress. Moreover, based on accurate and direction-aware link-quality measurements, STPRSA identifies and exploits under-utilized asymmetric links, thus improving the utilization of network capacity by up to 85 percent finally; IARS is designed to be easily deployable in existing IEEE 802.1-based wired networks without any change of MAC which is discussed in chap5 firmware or system kernel compilation. STPRSA has been evaluated extensively via both distributed system experimentation on a Linux-based implementation, demonstrating its superior accuracy and efficiency over existing measurement techniques.