RRSTP: A Spanning Tree Protocol for Obviating Count-to-Infinity from Switched Ethernet Networks

Syed Muhammad Atif syed.muhammad.atif@gmail.com
M.S Computer Networks
Department of Computer System Engineering
Usman Institute of Technology
Karachi, Pakistan.

Abstract

This paper proposes a highly reliable and rapidly converging spanning tree protocol named as Reliable Rapid Spanning Tree Protocol. The need of this spanning tree protocol is felt because reliability of switched Ethernet networks is heavily dependent upon that of spanning tree protocol. But current standard spanning tree protocol – Rapid Spanning Tree Protocol – is well known for its susceptibility to classical count-to-infinity problem. Because of this problem the protocol has extremely variable and unexpectedly high convergence time even in small networks. As a result network wide congestion, frame loss and frame delay may occur. Even forwarding loops may be induced into the network under certain circumstances. It is expected that the new protocol – RRSTP – will significantly increase the dependability of switched Ethernet networks by providing guaranteed protection against the count-to-infinity problem.

Keywords: Network Reliability, Count-to-Infinity, Network Convergence, RSTP.

1. INTRODUCTION

For last two decades, switched Ethernet networks are the most popular local area networks (LANs). The reasons of it are its auto configuration, easy availability, low cost, backward compatibility and scalability to higher bandwidth. In switched Ethernet networks, switches are core devices. Ethernet switch is a multi-port layer 2 network device that forwards frame to specific ports rather than, as in conventional hub, broadcasting every frame to every port. In this way, the connections between ports deliver the full bandwidth available. That is why switched Ethernet networks exhibit appreciably better performance, throughput and scalability than that of conventional Ethernet networks.

In its pure form switched Ethernet networks cannot be used with a physical topology having cycles or redundant links. The reason is two folded. First, broadcast frames and unknown unicast frames flooded by switches may circulate in cycle forever. Second, dynamic address learning mechanism may pollute the filtering table of the switch. Since redundant links in physical topology are highly essential for fault tolerance that is why most present switches use a vital management protocol known as spanning tree protocol in order to allow physical topologies having redundant link or cycles. This protocol puts redundant links of physical topology in hot standby position by developing a logical spanning tree, in distributed fashion, over an underlying physical topology. Thus the physical topology seems to be cycle free for all switches in the network. Current standard spanning tree protocol, commonly known as Rapid Spanning Tree Protocol, is a variant of distance vector routing protocol. Distance vector routing protocols and so RSTP [1] are consider highly vulnerable to count-to-infinity problem. But it was Mayer at al. [2] who first mentioned that count-to-infinity problem may become severe under certain circumstances in RSTP controlled network. This highly undesirable behavior were later studied in detail by Elmeleegy at el. [3] [4] and Atif [5]. When count-to-infinity occurs, convergence time of RSTP sharply increased to tens of seconds [3]. Moreover, forwarding loops may also be induced into the network to further complicate the problem [4]. This vulnerability of RSTP severely affects the reliability of switched Ethernet networks as their reliability is heavily dependent on their spanning tree protocols. This paper will present an all new spanning tree protocol – named as Reliable Rapid Spanning Tree Protocol – specifically tailored to provide protection to Ethernet network against highly undesirable count-to-infinity problem. RRSTP is designed in such a manner that its convergence time is comparable to that of RSTP. Therefore, the new protocol will dramatically increase the reliability of switched Ethernet networks without compromising on their availability.

The rest of paper is organized as follows. Section 2 will give a brief overview of RSTP [1]. Section 3 will discuss the conditions that need to be satisfied for count-to-infinity to occur in a spanning tree protocol controlled network. Section 4 will propose the solution to handle this problem. Section 5 will discuss the related work and then section 6 will conclude the paper.

2. OVERVIEW OF RAPID SPANNING TREE PROTOCOLS

Current standard spanning tree protocol – Rapid Spanning Tree Protocol – is the successor of Spanning Tree Protocol. The earlier standard Ethernet spanning tree protocol – STP – was first proposed by Perlman in [6]. RSTP [1] is basically an integration of work of Mick Seaman presented in [7], [8], [9], [10] to reduce the convergence time of STP [11]. This section gives a brief overview of RSTP.

In RSTP every switch and every port of a switch has a unique identifier. A Root Switch, a switch having the smallest switch identifier, is elected through a distributed mechanism. Each switch calculates and maintains the shortest path to the Root Switch to construct the spanning tree. Switch and Port Identifiers are used as tie breaker when two paths are otherwise same.
Switches use Bridge Protocol Data Units (BPDUs) to exchange information among them. A port that is receiving the BPDU having the best path to Root Switch becomes the root port of the switch. All remaining ports of a switch always transmit the BPDUs having information of switch's root port. Ports receiving inferior information than one they are transmitting become designated ports. A switch uses only its root port and designated ports for forwarding data. Alternate and back up ports, ports that are neither the root port nor designated ports, are kept in stand by position for use in case of link failure or topology change.

RSTP has several unique features to keep the convergence time as low as possible. In RSTP, an alternate port, a port that have a better but not the best path to the Root Switch, becomes the new root port and so immediately moves into forwarding state after retirement of the current root port [7]. For quick propagation of failure information, an RSTP switch is allowed to process inferior BPDUs on the root port and alternate ports, if they are transmitted by their respective designated port [8]. Further, RSTP uses a handshake mechanism (sync) to quickly put a designated port, connected to a point-to-point link, into forwarding state [9].

In the event of a topology change, switches need to flush some of their addresses from its forwarding table, a table that records MAC addresses and their associated ports learnt through address learning mechanism. It is necessary because a station may change its position with respect to switch after a topology change. RSTP uses an address flushing mechanism presented by Vipne Jain and Mick Seaman [10]. This mechanism can flushes the required addresses more quickly as compared to the one used by STP.

3. COUNT-TO-INFINITY IN SPANNING TREE CONTROLED NETWORKS

This section will discuss when and how count-to-infinity may occur in spanning tree controlled networks. Count-to-infinity problem is highly undesirable to Ethernet networks as it adversely effects the convergence time of the network and thus decreases the network availability. Atif, in [5], has deeply discussed the count-to-infinity problem in spanning tree controlled networks in a novel fashion using some new terminologies. In that paper he has mentioned six conditions that must be satisfied simultaneously in order to induce count-to-infinity into the network. This section will partially reproduce the original work of Atif in [5] for ease of reference.

In a fully converged spanning tree controlled network all alternate ports are dual rooted i.e. have two distinct path to the Root Switch. One path of an alternate port to the Root Switch passes through its link's designated port while the other path passes through its switch's root port. However an alternate port may loss its one or both paths to the Root Switch if the root port of its upstream switch fails. So in a network in which a switch suffering from the root port failure, an alternate port may have no, one or two path(s) to the Root Switch and thus will be called orphan, single rooted and dual rooted alternate port respectively in this text. Orphan alternate ports must not be used to reunite the network temporarily segregated due to the root port failure of a switch. Because such alternate ports have information which is no longer valid. Moreover, dual rooted alternate ports are not used by spanning tree protocols to prevent forwarding loops. This left only single rooted alternate ports that can be used reunite the partitioned network, which in fact also have the potential to do so. Hence the underlying spanning tree protocol must use only single rooted alternate ports to restore connectivity.

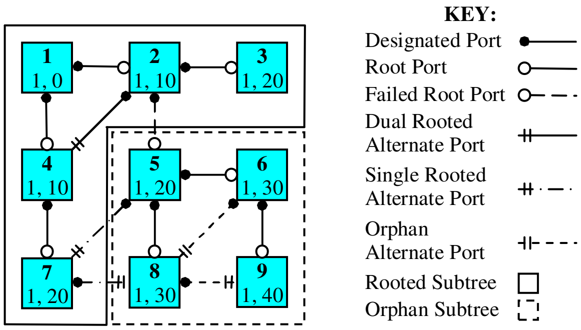


FIGURE 1: Different types of alternate ports in a network after failure of the root port of switch 5.

In a fully converged spanning tree controlled network, failure of the root port (or the designated port associated with the root port) of a switch results into a partition of underlying spanning tree into two distinct subtrees namely a *rooted subtree*, a subtree that still have the Root Switch, and an *orphan subtree*, a subtree that no longer have the previous Root Switch. It has to be noted that since all the switches in the orphan subtree have lost their path to previous Root Switch through their respective root ports. Therefore dual rooted alternate ports cannot exist in orphan subtree. In contrast, all the switches in rooted subtree have a path to the Root Switch through their respective root port. Hence orphan alternate ports cannot exist in rooted subtree. However, single rooted alternate ports can be found in both subtrees but only near the common boarder of these two subtrees. An alternate port in the rooted subtree is single rooted if and only if its associated designated port is in the orphan sub tree. Similarly an alternate port in the orphan subtree is single rooted if and only if its associated designated port is in the rooted subtree. These facts are depicted in Figure 1 through an exemplary network. Each switch is represented by a small box. The top number in the box is the Switch ID, the lower set of numbers represents the Root Switch ID as perceived by the switch and the cost to this Root Switch. All links have cost of 10. Figure 1 shows the snapshot of network immediately after failure of the root port of switch **5**. Switches **1** to **4** and switch **7** are in rooted subtree and switch **5**, **6**, **8** and **9** are in orphan subtree. Alternate port of switch **4** is still dual rooted as it is inside the rooted subtree. Moreover, Alternate port of switch **7** and that of switch **8** connected to switch **7** are single rooted alternate ports as they are near the common boarder of two subtrees. While alternate of switch **8** connected to switch **6** and that of switch **9** are orphan alternate ports as they are inside the orphan subtree.

Switches in a spanning tree controlled network use messages to communicate with each other. These messages experience a transmission delay when passing through the network. Thus, failure the root port of a switch may put all its downstream switches, that is switches in orphan subtree, into an inconsistent state for a period of time. The absolute period of inconsistence for a switch **B** is from the time when one of its upstream switch's root port (or the designated port associated with the upstream switch's root port) fails to the time when this information will be received on the root port and all alternate ports (if any) of the switch **B**. The effective period of inconsistence for a switch **B** is a bit small and it spans from the time when the first time switch **B** receives failure information of its upstream switch's root port on its root (or alternate) port to the time this will be received on all its remaining alternate port(s) (and the root port). Clearly, only inconsistent switches may have orphan alternate port(s) because of lack of information. Further, such switches cannot differentiate an orphan alternate port from the other two types of alternate ports.

Count-to-infinity occurs in the part of network constituting the orphan subtree, if six conditions are satisfied simultaneously. Three of them have to be satisfied by an inconsistent switch **B**:

1. Switch **B** has an orphan alternate port **a** such that its root path cost is smaller than that of the best single rooted alternate port in the network.
2. Switch **B** starts to declare its orphan alternate port **a** as designated port or the root port when it is still in the effective inconsistent port or switch **B** is declaring its orphan alternate port **a** as designated port when it is entering into the absolute inconsistent state.
3. Switch **B** is injecting the stale BPDU through its retiring orphan alternate port **a** that is becoming designated port or through its retiring root port that is becoming the designated port because the orphan alternate **a** is becoming the new root port.

Two conditions must be satisfied by an upstream switch **A** along with above three conditions:

4. Switch **A** accepts the stale BPDU, transmitted by switch **B**, on its designated port **d**, as it is conceived as superior BPDU by switch **A**. This makes port **d** the new root port of switch **A**. It may happen only if the switch cannot differentiate between stale and fresh BPDUs.
5. Switch **A** begins to propagate the stale BPDU further through its now designated ports.

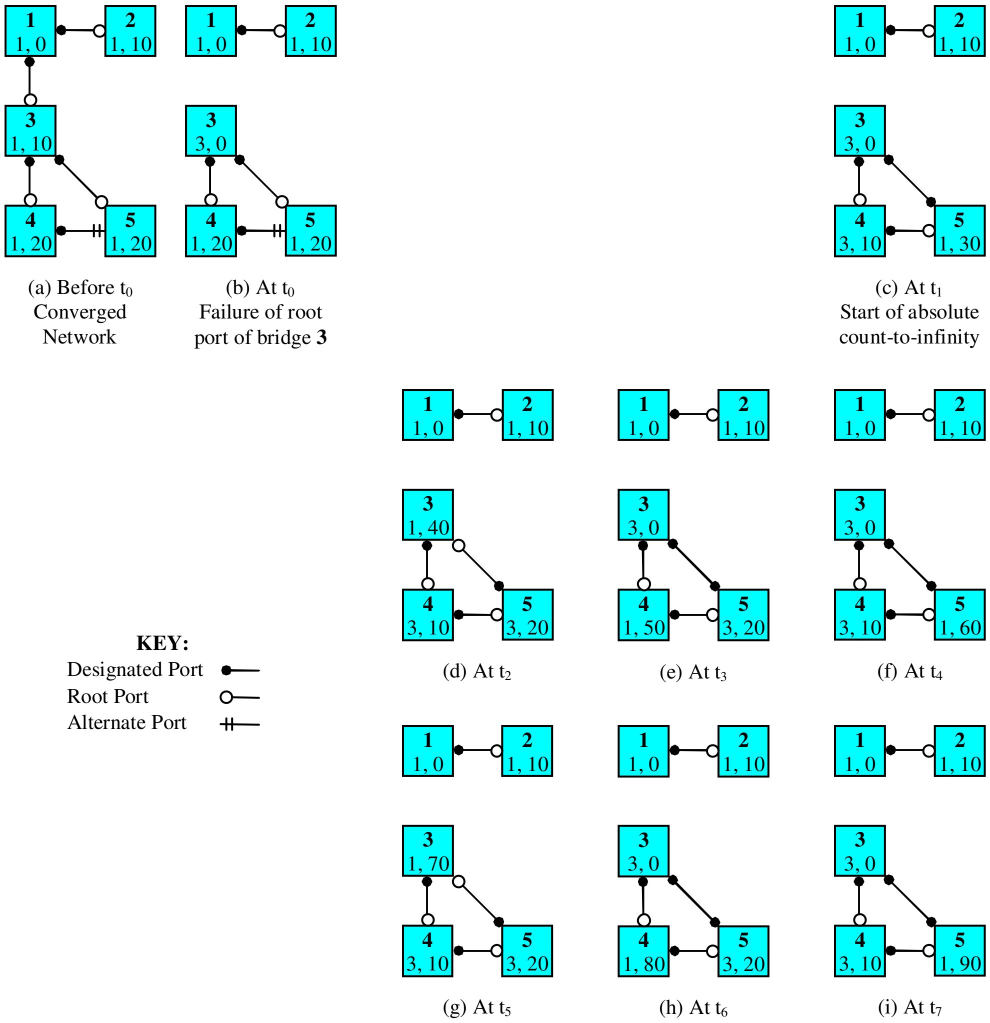
One condition needs to be met by underlying network.

6. There is at least one (unbroken) cycle in the network passing through switch **A**'s new root port **d** and switch **B**'s orphan alternate port a.

The first and the last condition for count-to-infinity are unavoidable in a high available fault tolerant network. However, remaining conditions can be easily avoided from being satisfied, by making slight modifications in underlying spanning tree protocol, to make the underlying network completely secure from the highly dangerous count-to-infinity problem.

When count-to-infinity occurs, the stale information begins to circulate in cycle and thus increments the root path cost of suffering switches with a definite offset, equal to the cycle's path cost, in each complete cycle (see Figure 2). Theoretically speaking, count-to-infinity in the network may be temporary or absolute. Temporary count-to-infinity in the network terminates after a definite interval of time. On the other hand absolute count-to-infinity persists forever. Temporary count-to-infinity occurs when the network is temporary segregated i.e. there is at least one single rooted alternate port in the network but a downstream switch mistakenly turns its orphan alternate port into root or designated port to reunite the partitioned network. When this happen count-to-infinity lasts until root path cost of one of the suffering switch exceed to that of the best single rooted alternate port in the network. Absolute count-to-infinity occurs when the network is absolutely segregated ,that is there is no single rooted alternate port in the network or in other words the best single rooted alternate port in the network has the root path cost of infinity, but a downstream switch starts to use an orphan alternate port to reunite the partitioned network.

Backup port can be made designated port after failure of its corresponding designated port without any fear of induction of count-to-infinity into the network. The reason is two folded. First, all the root ports on the shared medium start to pretend like single rooted alternate ports that can provide a path to Root Switch through the backup port corresponding to the failed designated port. Second, the root path cost of these pretending single rooted alternate ports is better than that of all orphan alternate ports in the orphan subtree i.e. violation of condition 1 of six conditions required for count-to-infinity. Moreover, change in port cost of the root port of a switch also forces the port to act like a single rooted alternate port.



(a) Before t_0 Converged Network

(b) At t_0 Failure of root port of bridge **3**

(c) At t_1 Start of absolute count-to-infinity

(d) At t_2

(e) At t_3

(f) At t_4

(g) At t_5

(h) At t_6

(i) At t_7

FIGURE 2: A network suffering from absolute count-to-infinity after failure of switch 3's root port because switch 5 is declaring its orphan alternate port as the new root port.

In RSTP controlled switched Ethernet networks above mentioned six conditions for count-to-infinity may be satisfied quite easily. That is why such networks are highly vulnerable to both temporary and absolute count-to-infinities. This highly undesirable behavior of RSTP was discussed, in detail, by Atif in [5].

4. RRSTP: THE RELIABLE RAPID SPANNING TREE PROTOCOL

Reliable Rapid Spanning Tree Protocol – RRSTP – is a spanning tree protocol that is specifically designed to protect switched Ethernet networks against count-to-infinity. The protocol consist of two mechanisms namely Rapid BPDU Distribution Mechanism and Root Switch Reelection Mechanism for timely convergence of network to a new topology after a topological change. This section will discuss the operation of RRSTP in detail.

Rapid BPDU Distribution Mechanism

It is mentioned in section 3 that switches in orphan subtree cannot distinguish between single rooted and orphan alternate ports when they are in effective inconsistent state. However, in RSTP, switches in orphan subtree are allowed to use their alternate ports to keep the convergence time as low as possible. This opens a gate for induction of count-to-infinity into the network. In contrast, RRSTP does not allow switches to use their alternate ports. But to keep the convergence time comparable to RSTP, an alternative mechanism is provided in RRSTP and it is named as Rapid BPDU Distribution Mechanism. It is the core or primary convergence mechanism used by RRSTP.

The Rapid BPDU Distribution Mechanism, in its simplified form, works as follows:

1. If a non edge designated port of a switch fails (that may be marked as failure of neighboring root port on the link) then
   a. Transmits Configuration BPDU on all its non-edge designated port(s).
   b. Transmits a Request BPDU on its root port.
2. If a switch receives an Request BPDU on its non-edge designated port then it must do a and b of 1.

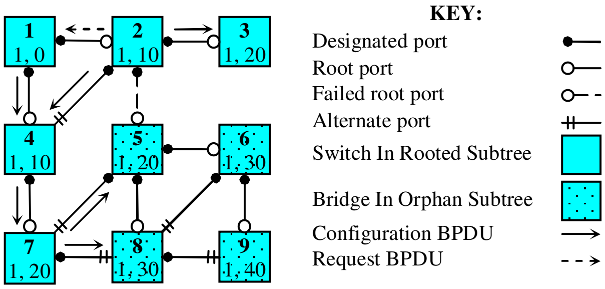


FIGURE 3: Flow of Configuration and Request BPDU after failure of the root port of switch 5 in an RRSTP controlled Ethernet network using Rapid BPDU Distribution Mechanism.

Upon failure of designated port of a switch, all the descendent (downstream) switches of that port become part of an orphan subtree. But the switch experiencing the designated port failure is still in the rooted subtree. That is why that switch initiates the Rapid BPDU distribution mechanism in RRSTP. Configuration BPDUs propagated through this mechanism may enter into an orphan subtree only through single rooted alternate ports as depicted in Figure 3. So, switches in an orphan subtree may use such BPDUs without any fear of induction of count-to-infinity.

Configuration BPDU in RRSTP, similar to RST BPDU in RSTP, has two new fields namely Sequence Number and Originator Root Path Cost fields to facilitate switches in an orphan subtree in recognizing fresh (valid) BPDUs transmitted through Rapid BPDU Distribution Mechanism. Only Root Switch is allowed to decrement Sequence Number. Non-root switches can generate BPDUs with the latest Sequence Number it has received from the Root Switch. Switches in orphan subtree must discard a BPDU if the BPDU has Sequence Number worse than they already have. Further, a switch in orphan subtree can accept a Configuration BPDU with same Sequence Number if and only if its Originator Root Path Cost is better than that conceived by receiving switch. A non-root switch can generate Configuration BPDUs only with Originator Root Path Cost worse than or equal to its own Root Path Cost. However, it is perfectly allowed for a switch to relay BPDUs with Originator Root Path Cost better than its own Root Path Cost.

Root Switch Reelection Mechanism

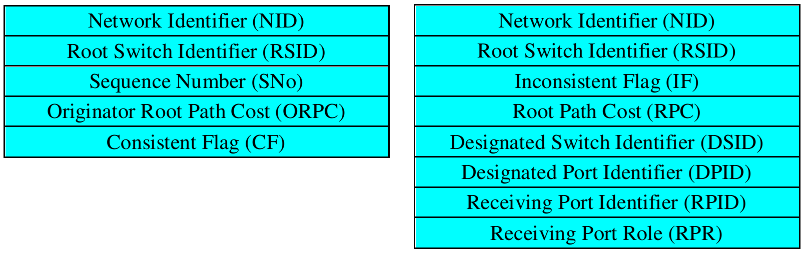
In RRSTP, switches in orphan subtree are solely dependent upon switches in the rooted subtree for converging to a new topology. However, Configuration BPDUs generated through Rapid BPDU Distribution Mechanism cannot enter into an orphan subtree if the network is absolutely segregated i.e. the network has no single rooted alternate port. This may force switches in an orphan subtree to stick to previous topology indefinitely. To overwhelm this problem, RRSTP has Root Switch Reelection Mechanism. This mechanism is used only as backup or secondary mechanism for converging the network.

The Root Switch Reelection Mechanism, in its simplified form, works as follows

1. If the root port of a switch fails then
   a. Set the mode of the switch to Inconsistent
   b. Start Inconsistent Mode timer
2. If a switch receive a fresh (valid) BPDU with Consistent Flag is clear on its root port then do a and b of 1.
3. If a switch in Inconsistent mode receives a fresh (valid) BPDU with Consistent Flag is set then it reverts back to Consistent mode again.
4. If the Inconsistent Mode timer of switch expires and the switch is still in the Inconsistent Mode then the switch must declare itself the Root Switch and move into Consistent mode again.

RRSTP adds a new field that is Network Identifier into the Configuration BPDU. A non-root switch in an orphan subtree decrements its Network Identifier field before declaring itself the root switch after expiration of its Inconsistent Mode Timer. It makes fresh (valid) BPDUs, originated by that switch, more preferable over stale (invalid) BPDUs, previously originated by the now inaccessible Root Switch. An incoming BPDU is accepted if Network Identifier in the BPDU is less than that conceives by the receiving switch. It ensures that stale BPDUs of the now inaccessible Root Switch will not override fresh BPDUs announcing a switch of orphan subtree as the Root Switch.

Protocol Definition



Network Identifier (NID)
Root Switch Identifier (RSID)
Sequence Number (SNo)
Originator Root Path Cost (ORPC)
Consistent Flag (CF)

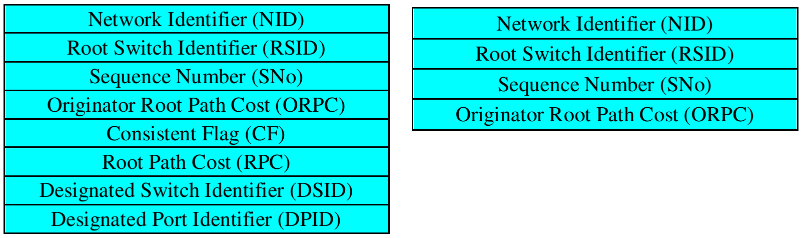
Network Identifier (NID)
Root Switch Identifier (RSID)
Inconsistent Flag (IF)
Root Path Cost (RPC)
Designated Switch Identifier (DSID)
Designated Port Identifier (DPID)
Receiving Port Identifier (RPID)
Receiving Port Role (RPR)

(a) Network Priority Vector

(b) Configuration Priority Vector

FIGURE 4: Structure of Priority Vector Pair in RRSTP.

Operation of RRSTP can be defined precisely and concisely with the help of six pairs of Priority Vectors namely Switch Priority Vector Pair, Root Priority Vector Pair, Root Path Priority Vector Pair, Port Priority Vector Pair, Designated Priority Vector Pair and Message Priority Vector Pair. Each Priority Vector Pair consist of a Network Priority Vector and a Configuration Priority Vector. Figure 4 is showing the structure of a Network Priority Vector and a Configuration Priority Vector in a Priority Vector Pair. Further, RRSTP uses two types of BPDUs that is Request BPDU and Configuration BPDU for communication with neighboring switches. Structure of Configuration and Request BPDUs are shown in Figure 5. A Request Priority Vector is also used in RRSTP to store and process Request BPDUs (See Figure 6 for structure of Request Priority Vector).



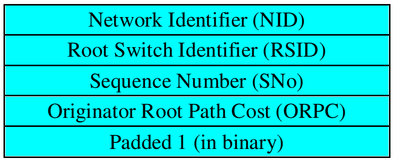
Network Identifier (NID)
Root Switch Identifier (RSID)
Sequence Number (SNo)
Originator Root Path Cost (ORPC)
Consistent Flag (CF)
Root Path Cost (RPC)
Designated Switch Identifier (DSID)
Designated Port Identifier (DPID)

(a) Configuration BPDU

Network Identifier (NID)
Root Switch Identifier (RSID)
Sequence Number (SNo)
Originator Root Path Cost (ORPC)

(b) Request BPDU

FIGURE 5: Structure of Configuration and Request BPDUs in RRSTP.



Network Identifier (NID)
Root Switch Identifier (RSID)
Sequence Number (SNo)
Originator Root Path Cost (ORPC)
Padded 1 (in binary)

(a) Request Priority Vector

FIGURE 6: Structure of Request Priority Vector in RRSTP.

In detail, the RRSTP works as follows:

1. An RRSTP switch may operates in two modes that is an Inconsistent mode and a Consistent mode. Inconsistent mode is a transient mode that lasts only few seconds. Whereas Consistent mode is an stable mode.
2. An RRSTP switch initializes Network Identifier, Sequence Number and Originator Root Path Cost fields of its Switch Priority Vector Pair to all 1s.
3. In RRSTP a received Configuration BPDU, stored in Message Priority Vector Pair, can be differentiated into five types specifically Better BPDU, Inferior BPDU, Inconsistent BPDU, Repeated BPDU and Refresher BPDU (See Figure 7)
4. In RRSTP, a received Configuration BPDU is considered as Better BPDU if
   a. Network Priority Vector of Message Priority Vector Pair is better than (numerically less than) or same as (numerically equal to) that of Port Priority Vector Pair and Configuration Priority Vector of Message Priority Vector Pair is better than (numerically less than) that of Port Priority Vector Pair.
   b. Network Priority Vector of Message Priority Vector Pair is better than (numerically less than) that of Port Priority Vector Pair and Configuration Priority Vector of Message Priority Vector Pair is worse than (numerically greater than) that of Port Priority Vector Pair but Designated Switch Identifier and Designated Port Identifier are same (numerically equal) in both Priority Vector Pairs and the received BPDU is not on the root port.
5. In RRSTP, a received Configuration BPDU is known as Inconsistent BPDU if Network Priority Vector of Message Priority Vector Pair is better than (numerically less than) that of Port Priority Vector Pair and Configuration Priority Vector of Message Priority Vector Pair is worse than (numerically greater than) that of Port Priority Vector Pair but Designated Switch Identifier and Designated Port Identifier are same (numerically equal) in both Priority Vector Pairs and the received BPDU is on the root port.
6. In RRSTP, a received Configuration BPDU is identified as Refresher BPDU if Network Priority Vector of Message Priority Vector Pair is better than (numerically less than) that of Port Priority Vector Pair but Configuration Priority Vector of Message Priority Vector Pair is same as (numerically equal to) that of Port Priority Vector Pair and the received BPDU is on the root port.

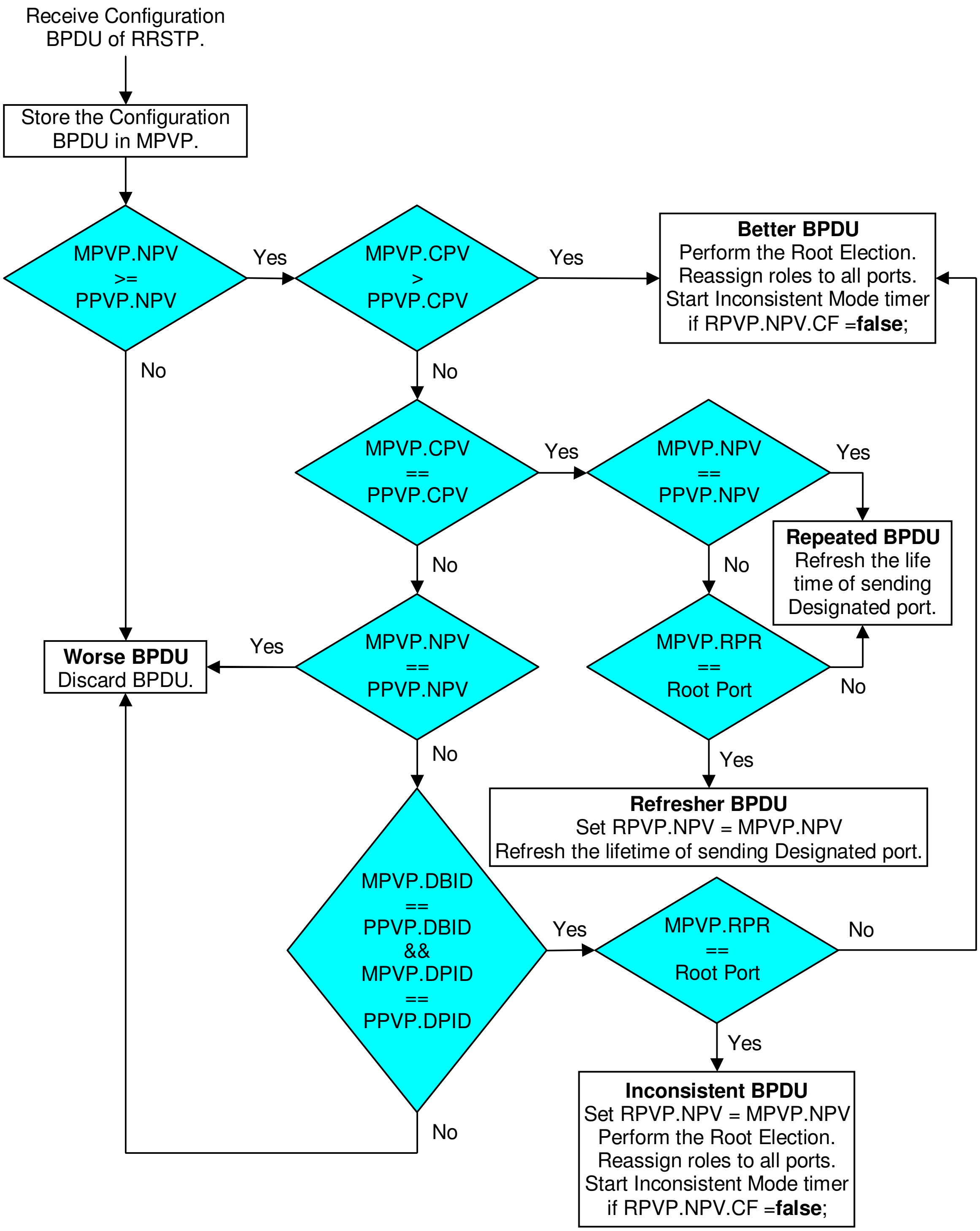

Receive Configuration BPDU of RRSTP.
Store the Configuration BPDU in MPVP.
MPVP.NPV >= PPVP.NPV
Yes
No
MPVP.CPV > PPVP.CPV
Yes
No
Better BPDU
Perform the Root Election.
Reassign roles to all ports.
Start Inconsistent Mode timer if RPVP.NPV.CF =false;
MPVP.CPV == PPVP.CPV
Yes
No
MPVP.NPV == PPVP.NPV
Yes
No
Repeated BPDU
Refresh the life time of sending Designated port.
MPVP.RPR == Root Port
No
Yes
Worse BPDU
Discard BPDU.
MPVP.NPV == PPVP.NPV
Yes
No
Refresher BPDU
Set RPVP.NPV = MPVP.NPV
Refresh the lifetime of sending Designated port.
MPVP.DBID == PPVP.DBID && MPVP.DPID == PPVP.DPID
Yes
No
MPVP.RPR == Root Port
No
Yes
Inconsistent BPDU
Set RPVP.NPV = MPVP.NPV
Perform the Root Election.
Reassign roles to all ports.
Start Inconsistent Mode timer if RPVP.NPV.CF =false;

FIGURE 7: Processing of received Configuration BPDU in RRSTP.

7. In RRSTP, a received Configuration BPDU is treated a Repeated BPDU if
   a. Network Priority Vector of Message Priority Vector Pair is better than (numerically less than) that of Port Priority Vector Pair but Configuration Priority Vector of Message Priority Vector Pair is same as (numerically equal to) that of Port Priority Vector Pair and the received BPDU is not on the root port.

   b. Both Network Priority Vector and Configuration Priority Vector of Message Priority Vector Pair are same as (numerically equal to) those of Port Priority Vector Pair respectively.

8. In RRSTP, a received Configuration BPDU is recognized as Worse BPDU if
   a. Network Priority Vector of Message Priority Vector Pair is worse than (numerically greater than) that of Port Priority Vector Pair
   b. Network Priority Vector of Message Priority Vector Pair is same as (numerically equal to) that of Port Priority Vector Pair but Configuration Priority Vector of Message Priority Vector Pair is worse than (numerically greater than) that of Port Priority Vector Pair.
   c. Network Priority Vector of Message Priority Vector Pair is better than (numerically less than) that of Port Priority Vector Pair but Configuration Priority Vector of Message Priority Vector Pair is worse than (numerically greater than) that of Port Priority Vector Pair and Designated Switch Identifier and Designated Port Identifier are not same (numerically equal) in both Priority Vector Pairs.
9. An RRSTP switch always encodes the BPDU transmitting through a port with Network Priority Vector of Root Priority Vector Pair and Configuration Priority Vector of that port's Designated Priority Vector Pair.
10. An RRSTP switch when receives a Refresher BDPU, first it sets the Network Priority Vector of its Root Priority Vector Pair to that of the root port's Message Priority Vector Pair and then forces all the ports to transmit Configuration BPDUs.
11. An RRSTP switch sets the Network Priority Vector of its Root Priority Vector Pair to that of the root port's Message Priority Vector Pair, performs the Root Election and reassigns role to all the ports when it receives an Inconsistent BPDU.
12. An RRSTP switch performs the Root Election and reassigns role to all the ports if: (see Figure 8)
    a. It receives a Better BPDU
    b. An edge port has just failed or disabled.
    c. An Alternate or Backup port has just failed, disabled, aged out or suffered from a change in Port Cost or Port Identifier.
    d. A Designated port has just suffered from a change in Port Cost.
13. When RRSTP switch that Originator Root Path Cost of Network Priority Vector of Root Priority Vector Pair is worse than (numerically greater than) Root Path Cost of Configuration Priority Vector of Root Priority Vector Pair has just suffered from failure, disabling or change in Port Identifier of Designated port, the switch: (see Figure 8)
    a. Sets Originator Root Path Cost of Network Priority Vector of Root Priority Vector Pair and that of Network Priority Vector of the root port's Port Priority Vector Pair equal to Root Path Cost of Configuration Priority Vector of Root Priority Vector Pair.
    b. Sets Request Priority Vector of port equal to first four fields of Network Priority Vector of Root Priority Vector Pair.
    c. Transmits Configuration BPDU on all ports and Request BPDU on the root port.
    d. Performs the Root Election and reassigns role to all ports.
14. If an RRSTP switch that Originator Root Path Cost of Network Priority Vector of Root Priority Vector Pair is not worse than (numerically not greater than) Root Path Cost of Configuration Priority Vector of Root Priority Vector Pair has just suffered from failure, disabling or change in Port Identifier of Designated port, the switch: (see Figure 8)
    a. Sets the first three fields of Request Priority Vector of port equal to those of Network Priority Vector of Root Priority Vector Pair.
    b. Sets Originator Root Path Cost of Request Priority Vector equal to one less than that of Network Priority Vector of Root Priority Vector Pair.
    c. Transmits Request BPDU on the root port.
    d. Performs the Root Election reassigns role to all ports.
15. When an RRSTP switch that Originator Root Path Cost of Network Priority Vector of Root Priority Vector Pair is worse than (numerically greater than) Root Path Cost of Configuration Priority Vector of Root Priority Vector Pair has just suffered from failure, disabling, aging out or change in Port Cost or Port Identifier of the root port, the switch: (see Figure 8)

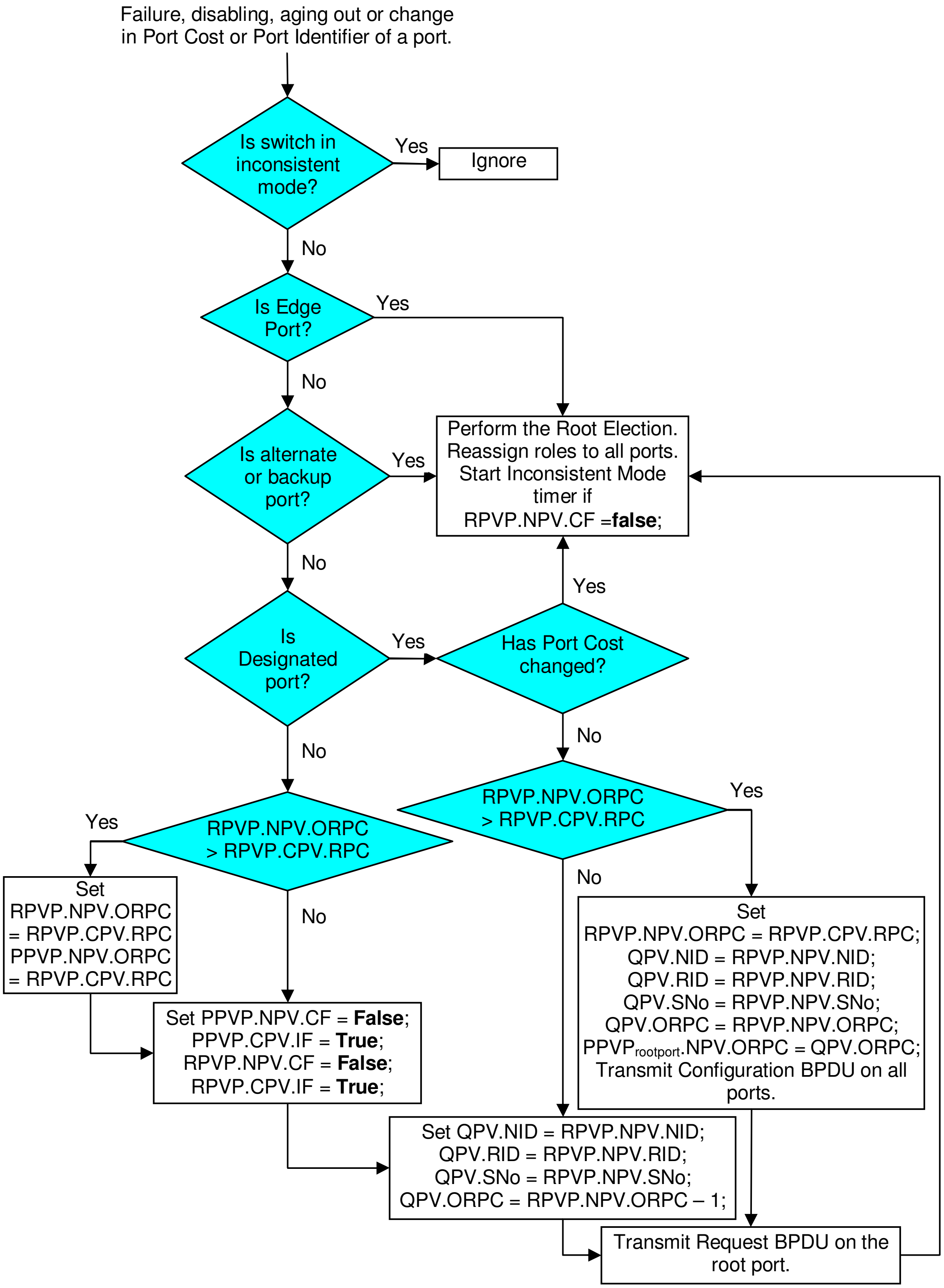

Failure, disabling, aging out or change in Port Cost or Port Identifier of a port.
Is switch in inconsistent mode?
Yes
Ignore
No
Is Edge Port?
Yes
No
Is alternate or backup port?
Yes
Perform the Root Election. Reassign roles to all ports. Start Inconsistent Mode timer if RPVP.NPV.CF =false;
No
Yes
Is Designated port?
Yes
Has Port Cost changed?
No
No
RPVP.NPV.ORPC > RPVP.CPV.RPC
Yes
Yes
RPVP.NPV.ORPC > RPVP.CPV.RPC
No
No
Set RPVP.NPV.ORPC = RPVP.CPV.RPC PPVP.NPV.ORPC = RPVP.CPV.RPC
Set RPVP.NPV.ORPC = RPVP.CPV.RPC; QPV.NID = RPVP.NPV.NID; QPV.RID = RPVP.NPV.RID; QPV.SNo = RPVP.NPV.SNo; QPV.ORPC = RPVP.NPV.ORPC; PPVProotport.NPV.ORPC = QPV.ORPC; Transmit Configuration BPDU on all ports.
Set PPVP.NPV.CF = False; PPVP.CPV.IF = True; RPVP.NPV.CF = False; RPVP.CPV.IF = True;
Set QPV.NID = RPVP.NPV.NID; QPV.RID = RPVP.NPV.RID; QPV.SNo = RPVP.NPV.SNo; QPV.ORPC = RPVP.NPV.ORPC – 1;
Transmit Request BPDU on the root port.

FIGURE 8: Handling of port failure and management changes on port in RRSTP

a. Sets Originator Root Path Cost of Network Priority Vector of Root Priority Vector Pair and that of Network Priority Vector of the root port's Port Priority Vector Pair equal to

Root Path Cost of Configuration Priority Vector of Root Priority Vector Pair.

b. Clears corresponding Consistent Flags and sets the corresponding Inconsistent Flags of both Root Priority Vector Pair and the root port's Port Priority Vector Pair.
c. Sets the first three fields of Request Priority Vector of port equal to those of Network Priority Vector of Root Priority Vector Pair.
d. Sets Originator Root Path Cost of Request Priority Vector equal to one less than that of Network Priority Vector of Root Priority Vector Pair.
e. Transmits Request BPDU on the root port of the switch.
f. Performs the Root Election and reassigns role to all ports.

16. If an RRSTP switch that Originator Root Path Cost of Network Priority Vector of Root Priority Vector Pair is not worse than (numerically not greater than) Root Path Cost of Configuration Priority Vector of Root Priority Vector Pair has just suffered from failure, disabling, aging out or change in Port Cost or Port Identifier of the root port, the switch: (see Figure 8)
    a. Clears corresponding Consistent Flags and sets corresponding Inconsistent Flag of Root Priority Vector Pair and the root port's Port Priority Vector Pair.
    b. Sets the first three fields of Request Priority Vector of port equal to those of Network Priority Vector of Root Priority Vector Pair.
    c. Sets Originator Root Path Cost of Request Priority Vector equal to one less than that of Network Priority Vector of Root Priority Vector Pair.
    d. Transmits Request BPDU on the root port of the switch.
    e. Performs the Root Election and reassigns role to all ports.

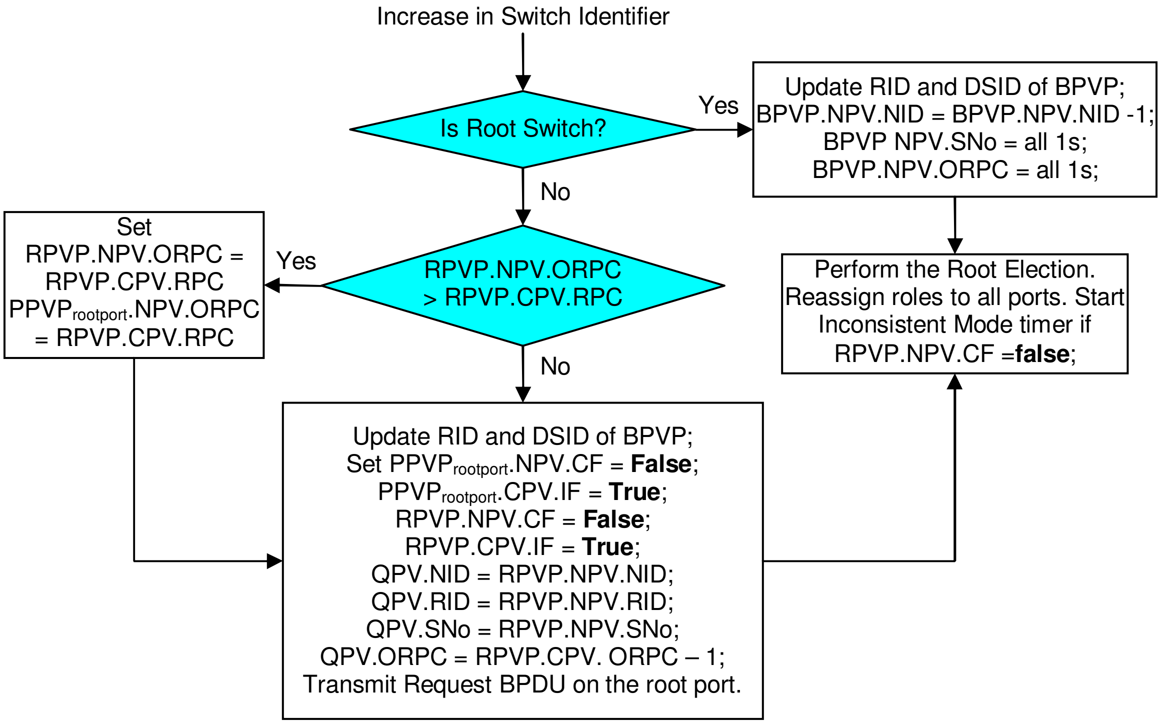

Increase in Switch Identifier
Is Root Switch?
Yes
Update RID and DSID of BPVP;
BPVP.NPV.NID = BPVP.NPV.NID -1;
BPVP NPV.SNo = all 1s;
BPVP.NPV.ORPC = all 1s;
No
Set
RPVP.NPV.ORPC =
RPVP.CPV.RPC
PPVProotport.NPV.ORPC
= RPVP.CPV.RPC
Yes
RPVP.NPV.ORPC
> RPVP.CPV.RPC
Perform the Root Election.
Reassign roles to all ports. Start
Inconsistent Mode timer if
RPVP.NPV.CF =false;
No
Update RID and DSID of BPVP;
Set PPVProotport.NPV.CF = False;
PPVProotport.CPV.IF = True;
RPVP.NPV.CF = False;
RPVP.CPV.IF = True;
QPV.NID = RPVP.NPV.NID;
QPV.RID = RPVP.NPV.RID;
QPV.SNo = RPVP.NPV.SNo;
QPV.ORPC = RPVP.CPV. ORPC – 1;
Transmit Request BPDU on the root port.

FIGURE 9: Handling of increase in Switch Identifier in RRSTP.

17. When a non-root RRSTP switch that Originator Root Path Cost of Network Priority Vector of Root Priority Vector Pair is worse than (numerically greater than) Root Path Cost of Configuration Priority Vector of Root Priority Vector Pair has just had worse Switch Identifier i.e. there is a numerical increase in Switch Identifier, the switch (See Figure 9).
    a. Sets Originator Root Path Cost of Network Priority Vector of Root Priority Vector Pair and that of Network Priority Vector of the root port's Port Priority Vector Pair equal to Root Path Cost of Configuration Priority Vector of Root Priority Vector Pair.

b. Updates Root Switch Identifier and Designated Switch Identifier of its Switch Priority Vector Pair.
c. Clears corresponding Consistent Flags and sets corresponding Inconsistent Flags of Root Priority Vector Pair and the root port's Port Priority Vector Pair.
d. Sets the first three fields of Request Priority Vector of port equal to those of Network Priority Vector of Root Priority Vector Pair.
e. Sets Originator Root Path Cost of Request Priority Vector equal to one less than that of Network Priority Vector of Root Priority Vector Pair.
f. Transmits Request BPDU on the root port of the switch.
g. Performs the Root Election and reassigns role to all ports.

18. When a non-root RRSTP switch that Originator Root Path Cost of Network Priority Vector of Root Priority Vector Pair is not worse than (numerically not greater than) Root Path Cost of Configuration Priority Vector of Root Priority Vector Pair has just had worse Switch Identifier i.e. there is a numerical increase in Switch Identifier, the switch (See Figure 9).
    a. Updates Root Switch Identifier and Designated Switch Identifier of its Switch Priority Vector Pair.
    b. Clears corresponding Consistent Flags and sets corresponding Inconsistent Flags of Root Priority Vector Pair and the root port's Port Priority Vector Pair.
    c. Sets the first three fields of Request Priority Vector of port equal to those of Network Priority Vector of Root Priority Vector Pair.
    d. Sets Originator Root Path Cost of Request Priority Vector equal to one less than that of Network Priority Vector of Root Priority Vector Pair.
    e. Transmits Request BPDU on the root port of the switch.
    f. Performs the Root Election and reassigns role to all ports.
19. When the Root RRSTP switch has just had worse Switch Identifier i.e. there is a numerical increase in Switch Identifier, then the switch.(See Figure 9)
    a. Updates Root Switch Identifier and Designated Switch Identifier of its Switch Priority Vector Pair.
    b. Decreases Network Identifier of Switch Priority Vector Pair by one.
    c. Set Sequence Number and Originator Root Path Cost of Switch Priority Vector Pair to all 1 (in binary).
    d. Performs the Root election and reassigns role to all ports.

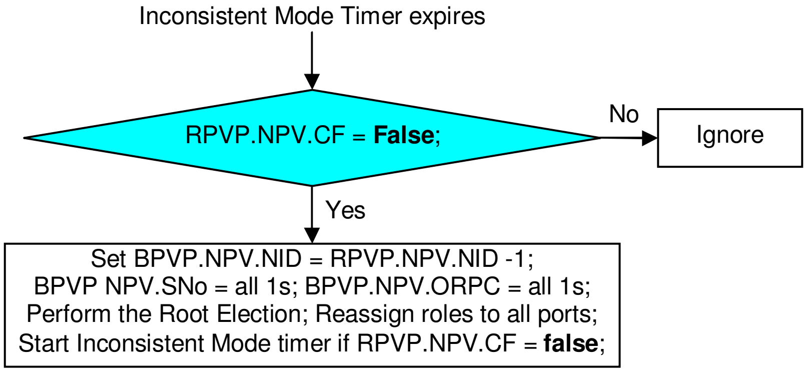

Inconsistent Mode Timer expires
RPVP.NPV.CF = **False**;
No
Ignore
Yes
Set BPVP.NPV.NID = RPVP.NPV.NID -1;
BPVP NPV.SNo = all 1s; BPVP.NPV.ORPC = all 1s;
Perform the Root Election; Reassign roles to all ports;
Start Inconsistent Mode timer if RPVP.NPV.CF = **false**;

FIGURE 10: Handling of expiration of Inconsistent Mode Timer in RRSTP.

20. An RRSTP switch that is operating in Inconsistent Mode and has just suffered from expiration of Inconsistent Mode timer must: (See Figure 10)
    a. Decrease the Network Identifier of its Switch Priority Vector Pair by one.
    b. Set Sequence Number and Originator Root Path Cost of Switch Priority Vector Pair to all 1 (in binary).
    c. Perform the Root Election and reassign role to all ports.

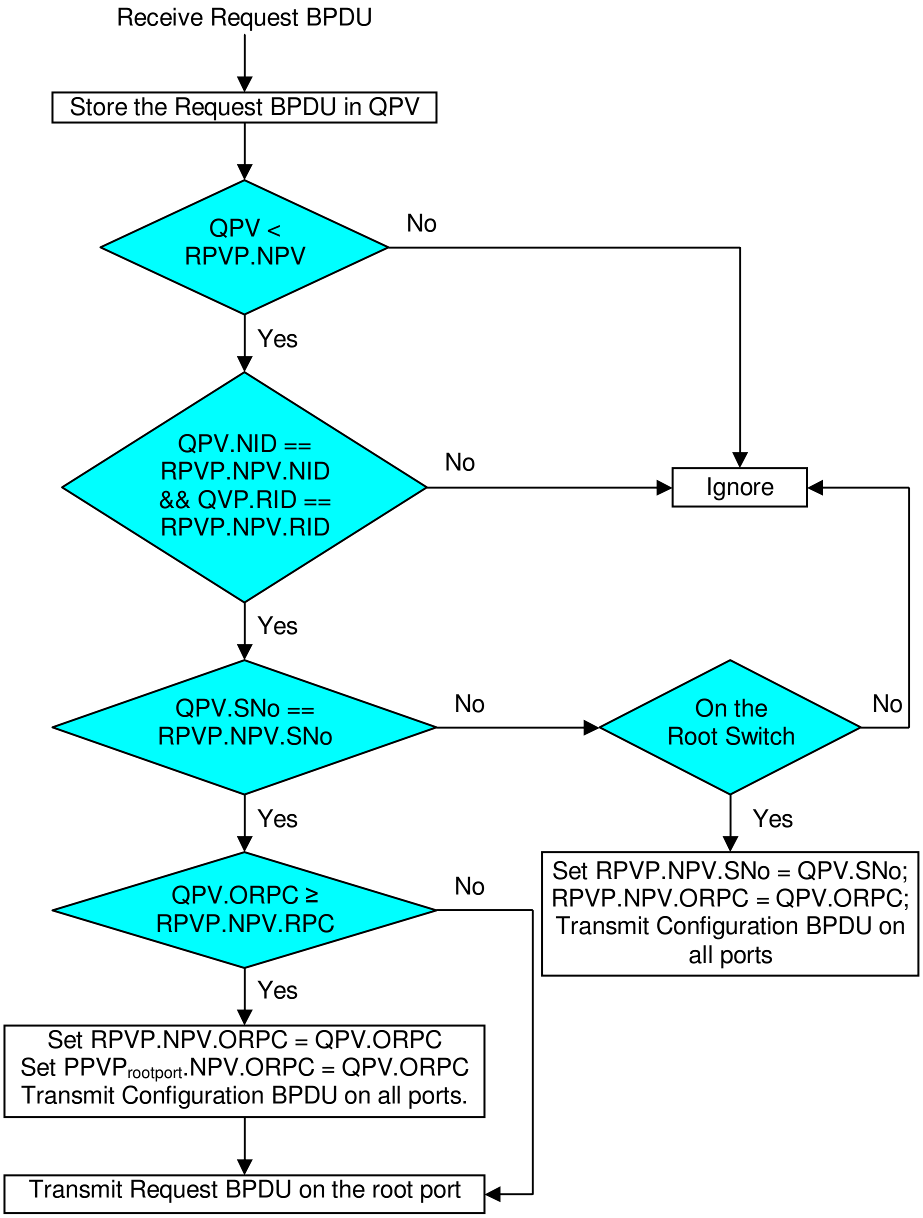

Receive Request BPDU
Store the Request BPDU in QPV
QPV < RPVP.NPV
No
Yes
QPV.NID == RPVP.NPV.NID && QVP.RID == RPVP.NPV.RID
No
Ignore
Yes
QPV.SNo == RPVP.NPV.SNo
No
On the Root Switch
No
Yes
Yes
QPV.ORPC ≥ RPVP.NPV.RPC
No
Set RPVP.NPV.SNo = QPV.SNo; RPVP.NPV.ORPC = QPV.ORPC; Transmit Configuration BPDU on all ports
Yes
Set RPVP.NPV.ORPC = QPV.ORPC
Set PPVProotport.NPV.ORPC = QPV.ORPC
Transmit Configuration BPDU on all ports.
Transmit Request BPDU on the root port

FIGURE 11: Processing of received Request BPDU in RRSTP.

21. An RRSTP switch transmits the received Request BPDU through its root port if the first three fields in the Request BPDU are same as (numerically equal to) those of Root Priority Vector Pair but Originator Root Path Cost of Request BPDU is better than (numerically less than) both Originator Root Path Cost and Root Path Cost of Root Priority Vector Pair. (see Figure 11)
22. When an RRSTP switch receives a Request BPDU such that the first three fields in the Request BPDU are same as (numerically equal to) those of Root Priority Vector Pair and Originator Root Path Cost of Request BPDU is better than (numerically less than) that of Root Priority Vector Pair but worse than or same as (numerically greater than or equal to) Root Path Cost of Configuration Priority Vector of Root Priority Vector Pair, then the switch: (see Figure 11)
    a. Sets Originator Root Path Cost of Root Priority Vector Pair to that of Request BPDU.
    b. Sets Originator Root Path Cost of the root port’s Port Priority Vector Pair (or Switch Priority Vector Pair in case of the Root switch) to that of Request BPDU.

c. Transmits Configuration BPDU on all the ports and Request BPDU on the root port.

23. When the RRSTP Root Switch receives a Request BPDU such that the first two fields in the Request BPDU are same as (numerically equal to) those of Root Priority Vector Pair but Sequence Number of Request BPDU is better than (numerically less than) that of Root Priority Vector Pair, then the switch: (see Figure 11)
    a. Sets Sequence Number and Originator Root Path Cost of Root Priority Vector Pair and Switch Priority Vector Pair to that of Request BPDU.
    b. Transmits Configuration BPDU on all the ports.

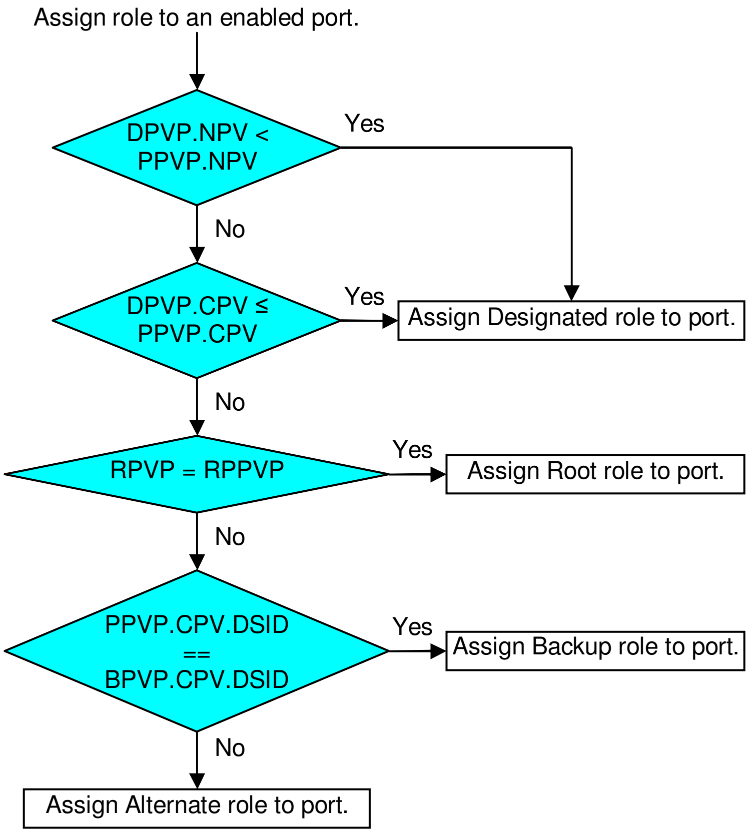

Assign role to an enabled port.
DPVP.NPV < PPVP.NPV
Yes
No
DPVP.CPV ≤ PPVP.CPV
Yes
Assign Designated role to port.
No
RPVP = RPPVP
Yes
Assign Root role to port.
No
PPVP.CPV.DSID == BPVP.CPV.DSID
Yes
Assign Backup role to port.
No
Assign Alternate role to port.

FIGURE 12: Assignment of port role to an enabled port in RRSTP.

24. An RRSTP switch assigns role to all ports, after the Root election , as follows: (See Figure 12)
    a. Assign Disabled role to a port if it is not enabled (operational).
    b. Assigns Designated role to the port if Network Priority Vector of its Designated Priority Vector Pair is better than (numerically less than) that of its Port Priority Vector Pair.
    c. Assigns Designated role to the port if Network Priority Vector of its Designated Priority Vector Pair is not better than (numerically not less than) that of its Port Priority Vector Pair but Configuration Priority Vector of its Designated Priority Vector Pair is not worse than (numerically not greater than) that of its Port Priority Vector Pair.
    d. Assigns Root role to the port if its Root Path Priority Vector Pair is same as (numerically equal to) Root Priority Vector Pair of the switch.
    e. Assigns Alternate role to the port if Network Priority Vector and Configuration Priority Vector of port's Designated Priority Vector Pair are not better than (numerically not less than) those of port's Port Priority Vector Pair, port's Root Path Priority Vector Pair is not same as (numerically not equal to) Root Priority Vector Pair of the switch and

Designated Switch Identifier of Configuration Priority Vector of port's Port Priority Vector Pair is not same as (numerically not equal to) Switch Identifier of the switch.

f. Assigns Backup role to all remaining ports.

25. An RRSTP switch forces all the ports to transmit Configuration BPDU whenever it performs the Root Election.
26. An RRSTP switch performs the Root Election as follows
    a. Find Root Path Priority Vector Pair that has the best (numerically least) Configuration Priority Vector from set of those Root Path Priority Vector Pairs that corresponding Network Priority Vectors are not worse (numerically not greater than) than that of Root Priority Vector Pair.
    b. Sets Originator Root Path Cost of Switch Priority Vector Pair to all 1 (in binary) and Network Identifier of Switch Priority Vector Pair to that of Root Path Priority Vector Pair elected in step a., if its Network Identifier is better than (numerically less than) that of Switch Priority Vector Pair.
    c. Set the Root Priority Vector Pair to that Priority Vector Pair that has better (numerically lesser) Configuration Priority Vector from set consist of Switch Priority Vector Pair and Root Path Priority Vector Pair elected in step a.

Discussion

In RSTP, a non-root switch can both generate and relay a BPDU but there is no mark distinction between them. In contrast, a non-root switch in RRSTP generates a configuration BPDU only after stamping it with its own or worse Root Path Cost i.e. setting its Originator Root Path Cost to its own or worse Root Path Cost. Hence Configuration BPDU that is relayed can be distinguished easily from one that is generated by a non-root switch in RRSTP.

RSTP is vulnerable to count-to-infinity because an RSTP switch suffering from a recent root port failure or management change on the root port may use a stale RST BPDU generated by a switch using its orphan alternate port either as the root port or designated port. In contrast, an RRSTP switch can distinguish and discard such problematic BPDUs. To obtain it, first an RRSTP switch processes only those Configuration BPDUs that have either better (numerically smaller) Sequence Number than the switch or have same Sequence Number but better Originator Root Path Cost than the switch. Second, an RRSTP switch suffering from a recent root port failure or a management change on the root port always sets its Originator Root Path Cost to its own Root Path Cost or better and clears its Consistent Flag (see Figure 8). The above two steps ensures that a switch suffering from a recent root port failure or a management change on the root port will not process a Configuration BPDU generated by a switch in the orphan subtree. As a switch in orphan subtree may generate a Configuration BPDU with Originator Root Path Cost at most equal to its own Root Path Cost which is always worse than that of a switch suffering from a recent root port failure or a management change on the root port.

As alternate ports are not using by RRSTP to restore the network connectivity, RRSTP uses the novel Rapid BPDU Distribution Mechanism in order to facilitate switches for quick and rapid convergence. Figure 8 is showing the procedure of generating a Request BPDU on the root port by a switch when one of its designated ports fails or suffers from a management change. Where as Figure 11 is illustrating the generation of Configuration BPDUs by a switch when it receives a Request BPDU that has Originator Root Path Cost worse than that switch's Root Path Cost.

Rapid BPDU Distribution Mechanism is effective only in temporary segregation of network. In case of absolute segregation of network, RRSTP reelects the root switch after expiration of Inconsistent Mode timer. RRSTP uses a conservative value (say 6 seconds) for Inconsistent Mode timer in order to provide enough time for Rapid BPDU Distribution Mechanism to successfully transmit Configuration BPDUs. It can effort such a conservative value because a switch running Inconsistent Mode timer remains intact with its previous topology and so a stations in the orphan subtree can communicate with other stations in the subtree.

Interoperability With Legacy Switches

RRSTP is not backward compatibility with legacy STP and RSTP switches. However, RRSTP can be integrated with DRSTP [5], a backward compatible solution to reasonably improve the reliability of legacy Ethernet networks. So, an DRSTP integrated RRSTP switch must operates in two different phases explicitly an DRSTP phase and an RRSTP phase. An DRSTP integrated RRSTP switch will said to be in DRSTP phase if the root port of the switch is receiving STP Configuration BPDU, RST BPDU or DRST BPDU. Otherwise the switch is said to be in RRSTP phase. When an DRSTP integrated RRSTP switch is in DRSTP phase, a port can operate as STP, RSTP or DRSTP port depending upon other switches on the link connected to that port. However, in RRSTP phase, a switch may have both legacy ports (STP, RSTP or DRSTP ports) and RRSTP ports. RRSTP Configuration BPDU must be converted into legacy BPDU, by simply stripping off newly added fields, before transmitting on a legacy port. Moreover, Root Election in an DRSTP integrated RRSTP switch is a bit more complex than that in RRSTP switch. First, the switch drives the DRSTP Root Priority Vector, using the DRSTP rules, from set of legacy ports (STP, RSTP or DRSTP ports). Second, the switch drives the RRSTP Root Priority Vector Pair, using the RRSTP rules, from set of RRSTP ports. Third, the switch drives stripped Configuration Vector. It is nothing but the Configuration Priority Vector of RRSTP Root Priority Vector Pair such that all the newly add fields of RRSTP are stripped off. The switch then moves into DRSTP phase and starts using its DRSTP Root Priority Vector if its DRSTP Root Priority Vector is better than stripped Configuration Vector. Otherwise the switch moves into RRSTP phase and starts using its RRSTP Root Priority Vector Pair.

Comparison with Contemporary Protocols

This section will critically discuss RRSTP with other contemporary protocols. The four other protocols that will be used for comparison are STP [11], RSTP [1], DRSTP [5] and RSTP with Epoch [3][4]. The five key aspects that will be discussed during comparison are vulnerability against count-to-infinity, convergence time, scalability, protocol implementation and backward compatibility.

Both STP [11] and RSTP [1] are susceptible to both temporary and absolute count-to-infinities as they cannot distinguish between stale and fresh BPDUs. In contrast, DRSTP [5] provide protection, to some extent, against both type of count-to-infinities. It is achieved by inserting a small delay to avoid usage of probably stale information. “RSTP with Epoch” [3][4] is a new protocol that is specifically designed to address the count-to-infinity problem but unfortunately it is vulnerable against temporary count-to-infinity. It is because a new epoch is stated in this protocol only when the switch suffering from root port failure has no alternate port. Fortunately, RRSTP is not vulnerable to both temporary and absolute count-to-infinities as it can distinguish between fresh and stale BPDUs and it also does not use orphan alternate ports to restore connectivity.

STP exhibits very slow convergence time of up to 50s [12] due to use of conservative timers. This also makes STP a low scalable protocol. In contrast, RSTP may converge with in 1-3s due to its aggressive and optimistic approach. But this low convergence time is showed by RSTP only in the absence of count-to-infinity. This vulnerability of RSTP also adversely affects its scalability. On the other hand, DRSTP usually exhibits convergence time of 1-3s as it can prevent count-to-infinity in most cases. Moreover, the scalability of DRSTP is generally more than RSTP. Convergence time of “RRSTP with Epoch” is order of round trip time of BPDU to the Root Switch [3][4]. In contrast, convergence time RRSTP is order of trip of BPDU around the shortest broken cycle. Further, scalability “RSTP with Epoch” is under question due to its convergence process. This protocol starts a new epoch and so reelection of the Root Switch whenever a switch having no alternate port experiences the failure of its root port. This reelection of the Root Switch produces an unnecessary network wide disturbance even when it is possible to reconverge the network without such reelection. In contrast, RRSTP avoids the reelection of the Root Switch until the expiration of Inconsistent Mode Timer. Moreover, it uses its highly decentralized “Rapid BPDU Distribution Mechanism” to reconverge the network. Hence, it expected that RRSTP will be much more scalable as compare to “RSTP with Epoch”.

The functionality of RRSTP is also very much similar to DSDV [13]. But, two major aspects in which RRSTP differ from DSDV are getting rid of settling timer and use of its decentralized “Rapid BPDU Distribution Mechanism”. These two aspects help RRSTP to keep its convergence time as low as possible.

RSTP [1] is backward compatible to STP [11]. DRSTP [5] is also backward compatible to STP [11] and RSTP [1]. But it cannot prevent count-to-infinity in the presence of STP switches in the network. Similarly, “RSTP with Epoch” [3][4] is also backward compatible to STP [11] and RSTP.[1] But this compatibility is provided at the cost of exposure of network to count-to-infinity. RRSTP can be also made backward compatible by integrating it with DRSTP [5].

In a nutshell, RRSTP is much superior to its contemporary protocol in most major aspects of network. Table 1 is showing the comparison of RRSTP with other contemporary protocols in a tabular form.

		STP	RSTP	DRSTP	RSTP with Epoch	RRSTP
Frequency of Count-to-infinity	Temporary	High	High	Low	High	Zero
	Absolute	High	High	Low	Zero	Zero
Convergence time	In case of no count-to-infinity	Up to 50s	1-3s	1-3s	Order of round trip time to Root Switch	Order of round trip time around shortest broken cycle
	In case of count-to-infinity	Order of maximum message age	Order of maximum message age.	Order of maximum message age.	Order of maximum message age	N/A
Scalability		Very Low	Low	Medium	High	Very High
Backward compatibility		N/A	Yes	Yes	Yes	Yes

TABLE1: Comparison of RRSRP with other contemporary protocols.

5. RELATED WORK

There are two schools of thought to increase reliability and scalability of Ethernet. Researchers in one school of thought are suggesting to replace current spanning tree protocol with other techniques. For example Perlman proposed Rbridges [14] and Garcia et al. proposed LSOM [15] to substitute spanning tree with more reliable and scalable link state routing. Turn prohibition technique can also be used in place of legacy spanning tree protocols to improve reliability and scalability. Up/Down routing proposed by Shoreder et al. [16], Turn Prohibition (TB) proposed by Starobinski et al. [17], Tree-Based Turn-Prohibition (TBTP) proposed by Pellegrini et al. [18] and Hierarchal Up/Down Routing and Bridging Architecture (HURP/HURBA) proposed by Ibáñez et al. [19] are few well-known algorithms based on this technique.

SEATTLE proposed by Kim et al. [20] is a completely new layer 2 network architecture. However, it is not backward compatible solution. Sharma et al. [21] introduce a multiple spanning tree architecture that improves the throughput and reliability over when using a single spanning tree. SmartBridges [22] uses the techniques of diffusing computation [23] and effective global consistency to achieve loop-freeness.

There are researcher in another school of thought that believe that reliability and scalability of spanning tree protocols can be enhance by making few modification in its operation. First serious attempt, to the best of my knowledge, is made by Elmeleegy et al. by proposing “RSTP with Epochs” [3],[4]. Other protocols that address the reliability of spanning tree protocol are DRSTP

[5] and Ether Fuse [24] proposed by Atif and Elmeleegy et al. respectively DRSTP was proposed to protect legacy RSTP controlled Ethernet networks against count-to-infinity to some extent. Ether Fuse [24] is a hardware device acting like a fuse that burn out logically before the count-to-infinity problem occurring in the network become severe.

6. CONCLUSION

This paper presents a novel spanning tree protocol, RRSTP, to safeguard the underlying Ethernet network against highly undesirable count-to-infinity problem. The protocol makes subtle modification in both structure and processing of BPDUs to achieve this goal. Further, to make the underlying Ethernet network highly available, RRSTP uses a novel decentralized "Rapid BPDU Distribution Mechanism. With the help of this mechanism, RRSTP can converge the network in order of time required for BPDU to travel along the shortest broken cycle. In a nutshell, RRSTP is expected to out class all its contemporary spanning tree protocols in all three key features specifically reliability, scalability and availability. Thus, Ethernet can now safely used along with RRSTP even in highly mission critical networks.

7. REFERENCES

1. LAN/MAN Standards Committee of the IEEE Computer Society. *"IEEE Standard for Local and metropolitan area networks: Media Access Control (MAC) Bridges - 802.1D"*. 2004.

2. Myers, T. E. Ng, and H. Zhang. *"Rethinking the Service Model: Scaling Ethernet to a Million Nodes"*. In Proceedings of the 3rd Workshop on Hot Topics in networks, San Diego,.CA, USA, 2004.

3. K. Elmeleegy, A. L. Cox and T. S. E. Ng. *"On Count-to-Infinity Induced Forwarding Loops in Ethernet Networks"*. In Proceedings of the 25th IEEE Infocom, Barcelona, Catalunya, Spain, 2006.

4. K. Elmeleegy, A. L. Cox and T. S. E. Ng. *"Understanding and Mitigating the Effects of Count to Infinity in Ethernet Networks"*. IEEE/ACM Transactions on Networking, 17(1):186-199, 2009.

5. S M Atif. *"DRSTP: A Simple Technique for Preventing Count-to-Infinity in RSTP Controlled Switched Ethernet Networks"*. International Journal of Computer Networks, 2(6):278-296, 2011.
6. R. Perlman. "*An Algorithm for Distributed Computation of a Spanning Tree in an Extended LAN"*. In the proceedings of 9th ACM Data Communications Symposium. New York, USA, 1985.

7. M Seaman. *"High Availability Spanning Tree"*. [online] Available at: www.ieee802.org/1/files/public/docs1998/hasten7.pdf. [Accessed 21 March 2011].

8. M. Seaman. *"Speedy Tree Protocol"*. [online] Available at: www.ieee802.org/1/files/public/docs1999/speedy_tree_protocol_10.pdf. [Accessed 21 March 2011].

9. M. Seaman. *"Truncating Tree Timers"*. [online] Available at: www.ieee802.org/1/files/public/docs1999/truncating_tree_timing_10.pdf. [Accessed 21 March 2011].

10. V. Jain and M. Seaman. *"Faster flushing with fewer addresses"*. [online] Available at: www.ieee802.org/1/files/public/docs1999/fast_flush_10.pdf. [Accessed 21 March 2011].

11. LAN/MAN Standards Committee of the IEEE Computer Society. *"IEEE Standard for Information technology – Telecommunications and information exchange between systems –*

Local and metropolitan area networks – common specifications, Part 3: Media Access Control (MAC) Bridges" , ISO/IEC 15802-3, ANSI/IEEE Std 802.1D, 1998.

12. Cisco Systems, Inc. *"Spanning Tree Protocol Problems and Related Design Considerations"*. [online] Available at: www.cisco.com/en/US/tech/tk389/tk621/technologies_tech_note09186a00800951ac.shtml [Accessed 21 March 2011].

13. C. E. Perkins and P. Bhagwat. *"Highly dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers"*. In Proceedings of the ACM SIGCOMM 1994, London, UK, 1994.

14. R. Perlman. *"Rbridges: Transparent routing"*. In Proceedings of the 23rd IEEE Infocom, Hong Kong, 2004.

15. R. Garcia, J. Duato and F. Silla. *"LSOM: A link state protocol over MAC addresses for metropolitan backbones using optical Ethernet switches"*. In Proceedings of the 2nd IEEE International Symposium on Network Computing and Applications, Cambridge, MA, USA, 2003.

16. M. Schroeder, A. Birrell, M. Burrows, H. Murray, R. Needham, T. Rodeheffer, E. Satterthwaite, C. Thacker. *"Autonet: A High-Speed, Self–Configuring Local Area Network Using Point–to–Point Links"*. IEEE Journal on Selected Areas in Communications, 9(8):1318–1335, 1991.

17. D. Starobinski, G. Karpovsky, F. Zakrevsky. *"Applications of network calculus to general topologies"*, IEEE/ACM Transactions on Networking, 11(3):411–422, 2003.

18. F. D. Pellegrini, D. Starobinski, M. G. Karpovsky and L. B. Levitin. *"Scalable cycle-breaking algorithms for gigabit Ethernet backbones"*. In Proceedings of the 23rd IEEE Infocom, Hong Kong, 2004.

19. Guillermo Ibáñez, Alberto García-Martínez, Juan A. Carral, Pedro A. González, Arturo Azcorra, José M. Arco. *"HURP/HURBA: Zero-configuration hierarchical Up/Down routing and bridging architecture for Ethernet backbones and campus networks"*, Computer Networks, 54(1):41-56,2010.

20. C. Kim, M. Caesar, and J. Rexford. "*Floodless in SEATTLE: A Scalable Ethernet Architecture for Large Enterprises"*. In Proceedings of the ACM SIGCOMM. 2008, Seattle, WA, USA, 2008.

21. S. Sharma, K. Gopalan, S. Nanda, and T. Chiueh. "*Viking: A multispanning tree Ethernet architecture for metropolitan area and cluster networks"*. In Proceedings of the 23rd IEEE Infocom, Hong Kong,. 2004.

22. T. L. Rodeheffer, C. A. Thekkath, and D. C. Anderson. *"SmartBridge: A scalable bridge architecture"*. In Proceedings of the ACM SIGCOMM. 2000, Stockholm, Sweden, 2000.

23. E. W. Dijkstra, C. S. Scholten. *"Termination detection for diffusing computations"*. Information Processing Letters, 11(1):1-4, 1980.

24. K. Elmeleegy, A. L. Cox and T. S. E. Ng. *"EtherFuse: An Ethernet Watchdog"*. In Proceedings of the ACM SIGCOMM 2007, Kyoto, Japan, 2007.