**LITERATURE SURVEY**

**OpenFlow-Based Segment Protection in Ethernet Networks**

Abstract:

Ethernet networks and industrial area networks have to provide a resilient comeback in case of a network failure. Open Flow Architecture is based on this Ethernet cables. This paves the way for link failure in these networks. So, this paper is based on providing a segment based rerouting algorithm for efficient transfer of messages in case of link failures.

Advantages:

This mechanism is efficient in choosing the backup path (secondary path) if a link is failed in between.

The secondary path is calculated within few tens of milliseconds.

Issues:

The proposed scheme takes a lot of memory space and is inefficient when it comes to memory handling.

Also, the time taken to recover is quite high which is approximately 1ms.

**Fast Recovery in Software-Defined Networks**

Abstract:

This paper implements a failover scheme with per-link Bidirectional Forwarding Detection sessions and preconfigured primary and secondary paths computed by an OpenFlow controller. The paper uses the usual way of detecting failures in Ethernet networks like if an acknowledgement is not received within 50-150ms , then the link is said to be broken. Uses two steps of process –The first step involve a fast switch-initiated recovery based on preconfigured forwarding rules guaranteeing end-to-end connectivity. The second step involves the controller calculating and configuring new optimal paths.

Advantages:

A lower detection time due to decreased session round trip time (RTT)

Removal of false positives. As each session spans a single link, false positives due to network congestion can be easily removed by prioritizing the small stream of control packets.

Issues:

Memory is not efficiently handled since the flow table should store two ways of possible communication.

Redundancy of routing information is stored in the group tables.

**Scalable Resilience for Software-Defined**

**Networking Using Loop-Free Alternates with Loop Detection**

Abstract:

In this paper we address both the scalability and resilience issues in OpenFlow Networks. OpenFlow switches store their flow tables in expensive, limited Ternary Content-Addressable Memory (TCAM) due to which the stored tables cannot be large, thus forwarding packets in line speed. Most resilience mechanisms require additional entries thus the implementation in OpenFlow may quickly exceed the available TCAM. Loop-Free Alternates (LFAs) are a standardized mechanism for fast reroute in IP networks which do not require additional entries. However, some LFAs cause loops when some node or multiple nodes failures occur. This renders additional links unusable. But if we were to exclude such LFAs, it would reduce the protected destination coverage even further. To overcome this, a scheme is designed to detect the loops caused by LFAs. This maximizes the protection coverage because the LFAs can be used without creating loops. This paper describes how LFAs and the loop detection scheme can be implemented in OpenFlow networks with only little packet overhead and a single additional entry per switch.

Advantages:

· They are simple and have no additional forwarding entries.

· There are no flow tables.

· The ternary content-addressable memory was used to store the flow tables. The TCAM is very expensive. Hence, here, its cost is also reduced.

· Maximized protection.

Issues:

· They mostly cannot protect all traffic and some of them cause micro-loops in case of node failures or multiple failures.

· Loop detection just helps to prevent loops for LFAs, but it cannot protect traffic for which no LFAs exist.

**Orion: A Hybrid Hierarchical Control Plane of Software-Defined Networking for Large-Scale Networks**

Abstract:

The three layers of the Orion model provide the hybrid hierarchical control layer for large scale networks. The Orion has the area controller which is responsible for collecting physical device information and link information, managing the intra-area topology and processing intra-area routing requests and updates. The domain controller synchronizes the global abstracted network view through a distributed protocol. The Domain Routing Management Sub-Module of the domain controller computes the global shortest path. When a PacketIn message reaches the area controller, the area routing management sub-module checks the source address and destination address of the message. If the destination address is in the area, the area controller employs Dijkstra algorithm to compute intra-area path and if is not present in the area, the area controller sends the source address and the destination address of the message to the domain controller, and stores the message to a waiting buffer with index. The domain controller computes the routing path for the flow and sends the routing result to the area controller.

Advantages :

· The CPU utilization of the domain controller is 40%.

· The domain controller of Orion costs lower than an OpenFlow based SDN controller.

· The load of the area controller of Orion is between the load of the controller in the flat architecture and the lowest area controller in the abstracted hierarchical SDN architecture.

Issues:

· Memory overhead due to the maintenance of area and domain controllers.

**Fault-Tolerant OpenFlow-based Software Switch Architecture with LINC Switches for a Reliable Network Data Exchange**

Abstract

The switches are essential for forwarding the packets in a local area network. If a switch fails, then the packets are not able to reach their destination. The new trend in Software Defined Networking (SDN) has made the use of software switches quite popular. These Software switches are required to be resilient to failure. This paper explains one mechanism for fault tolerance of LINC (Link Is Not Closed), an open source OpenFlow switch, which is written in Erlang programming language. We leverage the built-in concurrency, and fault-tolerant features of Erlang to realize a fault-tolerant distributed LINC switch system.

Advantages

· The LINC switches took only a little more than half the time the hardware switches took to connect hosts to the fault-tolerant system.

· When failover happens, the controller modifies the flow entries in the LINC switches which causes the packets to be sent to the new switch’s input port almost instantaneously.

Future Work:

The Erlang system shows some ease of programmability and faster deployment. Hence by efficient use of the Erlang Distributed System, the Fault Tolerant System can be improved further.

**Efficient routing for traffic offloading in Software-defined Network**

Abstract:

The Automatic Re-routing with Loss Detection architecture (ARLD) proposed in this paper works on the assumption that if a packet loss occurs at a link , it is mainly due to congestion. The controller treats this link as a bottleneck link. The Openflow protocol has a stats message which delineates the status of each node and each port to the SDN controller. The Re-routing module of the SDN controller computes an alternate path as a bypass route. The Re-routing module updates the virtual topology by eliminating the node at which the packet loss occurred and finds an alternative route without the switch that dropped the packet. After examining the availability of the path, the module returns the alternate route to controller and controller distributes new forwarding rules to each switch and updates the flow table. In this way the packet loss at the switches due to congestion can be reduced to yield better performance.

Advantages:

· With the help of an Openflow based SDN architecture , the controller detects the packet loss at switches in a shorter time.

· The model reduces the packet loss by providing a better performance.

Issues:

· When dropped packet is routed to bypass route the traffic on that route may cause congestion on that link. This effect would be a serious problem in larger sized network.

· In the proposed architecture, the average loss rate is reduced and this might decrease average latency of each flow, although measuring delay is not conducted due to the low timing realism of mininet emulation.

**Openflow Path Fast Failover Fast Convergence Mechanism**

Abstract:

The paper deals with a fast and efficient failover mechanism for redirecting traffic to more optimal backup path when there is a link failure or congestion problem in SDN. It also proposes a local pre-calculated path dataset mechanism in OpenFlow controller to allow fast network convergence. The central OpenFlow Controller computes the main and the best backup path based on the current network topology. OpenFlow controller is said to have a local dataset of path information and in case of a link failure or congestion in a path, the affected switch sends port-status message to the controller and the controller checks the flow entries affected by the failure. The controller pushes the main and the backup path to the OpenFlow switches and will recalculate the less congested backup path after it is updated periodically by the network. Once the controller get the notification about a link failure, it will perform simple lookup in its local dataset to find whole flow entry that affected by the failure. Finally, the affected entries will be deleted from the flow table and the pre-computed less congestion backup path will be selected. The controller then updates the flow entries of all switches and incorporates the new backup plan.

Advantages:

The single backup path for every main path in only one single switch flow table reduces the possibilities of flow table explosion.

The network traffic is redirected to alternate optimal path.

Issues:

Memory overhead due to the recalculation of less congested path in the controller.

**Detour Planning for Fast and Reliable Failure Recovery in SDN with OpenState**

Abstract:

This paper is based on providing a secure and reliable path if a particular node or a link fails. A protection scheme is given in this paper which calculates the backup paths in prior and the routing used here is MPLS which ensures zero packet loss after the link failure is detected. Also in this paper, the forwarding rules are done autonomously without the use of the controller always.

Advantages:

The recovery is done with the help of tagged indexes making the routing simpler.

The centralized controller is not frequently accessed since few forwarding rules are done independently.

No packets are lost.

Issues:

This mechanism cannot be expanded to multiple link failures.

If the tag indices are lost, then the packet drop numbers will increase.

**A Europe-wide demonstration of fast network resto- ration with OpenFlow**

Abstract:

During the data transmission, there may occur many number of the data failures in the data path. Segment protection is the key feature used to reroute the data in the secondary route which may or may not be the best or optimal path. Independent Transient Plane (IPL) is designed in this paper which reduces the path complexity and maintains the security of the data. This work results in the most efficient protection in the secondary path and optimal solution to the data failure problem.

Advantage:

This designed mechanism deals with the data packets in all paths simultaneously. The data packets which are lost during the transmission in the malicious path are gained by this designed. In some cases those data are sent to the destination in the legitimate path in retransmission phase.

Issues:

Here the regained data packets are not considered as legitimate; therefore there is a possibility of malicious data present in that regained data.

**Relaxed Multiple Routing Configurations:**

**IP Fast Reroute for Single and Correlated Failures**

Abstract:

The paper puts forth the methodology employed to recover from link failures in a network with the help of Relaxed Multiple Routing Configuration (rMRC). The MRC and the rMRC guarantee link or node failure from biconnected topologies. Backup topologies can be constructed using different methods and the number of states required in a router will increase with multiple backup paths. In the rMRC the requirements of the network topologies are relaxed. The difference between rMRC and MRC is that in conventional MRC the instantaneous recovery from the node failure is done by isolating the affected nodes. Instead, rMRC computes the shortest path without the failed link in the backup topology where the detecting node itself is isolated. Using the backup topology where the detecting node is isolated ensures that the traffic cannot loop back to the detecting node but still enables the rMRC forwarding to reach the destination node.

Advantages:

· The presented algorithm can guarantee link and node fault tolerance with fewer backup topologies than MRC.

· As relaxed backup topologies do not isolate all links, there is more flexibility in rMRC than in MRC to decrease the number of backup topologies.

Issues:

· rMRC’s ability to spread traffic over more links can sometimes have a dramatic impact in a sparsely connected network topology.