Network Recovery Optimization in SDN

Jozef Papan
University of Žilina, 010 26 Žilina, Slovakia
Jozef.papan@uniza.sk

Slavomir Tatarka
University of Žilina, 010 26 Žilina
slavomir.tatarka@fri.uniza.sk

Dominik Skvarna
University of Žilina, 010 26 Žilina, Slovakia
skvarna2@stud.uniza.sk

Ivana Bridova
University of Žilina, 010 26 Žilina, Slovakia
ivana.bridova@fri.uniza.sk

***Abstract*—Software-Defined Networking (SDN) introduces a centralized and flexible approach to network management, enabling dynamic traffic control and improved scalability. However, fault tolerance remains a critical challenge, as traditional failure recovery mechanisms often suffer from slow convergence and inefficient path selection, leading to significant packet loss and increased latency.**

This paper explores the effectiveness of various Fast Reroute (FRR) mechanisms in SDN environments, focusing on their impact on failure recovery times and network stability. Through extensive simulations, we demonstrate the limitations of existing solutions and highlight the necessity of integrating pre-computed backup paths to reduce service disruptions. Our findings indicate that without FRR mechanisms, packet loss exceeds 50% during multiple link failures, which is an unacceptable outcome for real-world deployments.

Keywords: SDN, FRR, LFA, Mininet, Opendaylight, BFD, DPQoAP

I. Introduction

During the transmission of a data flow, situations may arise in which a connectivity failure between the source node and the destination node inevitably occurs. Restoring connectivity between these nodes in the shortest possible time is crucial. The longer the state of no connectivity between the given nodes persists, the more packets that such nodes send to each other are lost, which negatively affects the Quality of experience (QoE) of the affected users [1].

A topology convergence occurs after detecting a link or node failure. During this time, the routing protocols used react to the resulting connectivity failure and try to find new routing paths that could be used to deliver the given data flow to the target destination without using faulty links in the given topology. However, this takes a certain time, during which there is still no connectivity between the affected nodes in the topology. The length of this time depends on various factors, such as the size of the given topology, the routing protocol used, and the hardware used.

Mechanisms have been developed to shorten the time when there is no connectivity between the given nodes; the task is to provide certain secondary protections intended to shorten this time. These mechanisms of fast network recovery should restore connectivity between the affected nodes in the shortest possible time until the given topology reaches a state of convergence, thanks to the convergence process of the given routing protocol. In this work, we will analyse selected basic mechanisms of fast network recovery, explain the logic of their functioning with examples and list their properties.

This work describes the results of simulations that dealt with this issue to prove the claim that the length of the convergence process in a topology negatively affects the percentage of packet loss sent between the affected nodes. It provides a detailed description of the course of the given simulations and topologies on which these tests were performed. The results obtained from these simulations are analysed and discussed in detail.

Furthermore, this paper describes the results of a study [2] dealing with creating a custom network recovery mechanism in case of a link failure. The dynamic protection mechanism with the quality of alternative paths, abbreviated DPQoAP [2], created by the authors of this study, brought a new perspective on the evaluation of a link failure since it perceived its heavy overload as its failure. Thanks to this change, the DPQoAP mechanism should maintain the same QoE in the topology, even during a link failure on the main routing path. Thus, this paper contains, in addition to a description of the functioning of the DPQoAP mechanism, the results of the tests that were performed in this study and a description of the logic of the functioning of individual algorithms created and used by the authors of the study in obtaining the results of these tests.

A. Network convergence

In this part of the work, we will describe and define the concept of convergence in the network. We will state what factors contribute to the final duration of the convergence time. We will also describe the cause and trigger for starting this process [3], [4].

1) Description

The process of convergence in the network occurs after detecting the failure of a link or a node in a given network. This process can take from a few milliseconds to tens of seconds. During this time, routers update their routing tables to restore node connectivity. However, before that happens, problems such as packet loss, interruption of communication due to the unavailability of end devices, or the creation of routing loops may appear in the network [3].

3) Duration of the convergence period

The parameters that are responsible for the resulting duration of the convergence process in a given network, i.e. the times when not all end devices are available in a given network, are listed below:

- **Detection of line or node failure.** This time can range from a few milliseconds to a few seconds. If the router manages to detect an outage on the physical layer, it will usually detect this outage earlier. The outage detection time can increase to tens of seconds when evaluating a line or node failure based on the Hello packets of the used routing protocol. During this period, no packets will be sent to their destination.
- **The relevant router's response to a line failure.** Under normal circumstances, during this period, the router creates and sends routing updates to the network.
- **Send information to other routers.** This period usually lasts from 10 to 100 milliseconds for one hop [1], [5].
- **Calculation of new routing tables.** Calculating new routing tables on a router that uses a link-state routing protocol using Dijkstra's algorithm takes a few milliseconds. However, this time may vary based on the size of the network. [5].
- **Writing new routing tables.** The duration of this operation depends on the hardware used and the number of prefixes affected by the line failure. Depending on the mentioned factors, the time required to make changes in routing tables can reach hundreds of milliseconds [5],[1].

4) Bidirectional Forwarding Detection

BFD is a network protocol that detects connectivity failures between two network devices. The time during which BFD can detect a line failure varies in milliseconds and, in some implementations, even in microseconds [6].

The three-way handshake method is used when creating a connection between two nodes to meet the needs of BFD operation. At the same time, it is possible to set and use the type of authentication. Valid authentication methods that can be used include plain text, SHA1, or MD5 encryption. The BFD network protocol can also work in two different modes of operation. We will describe these modes in more detail now:

- **Asynchronous mode.** In this mode, both nodes that have the BFD protocol implemented will send each other Hello packets at regular intervals. The length of these time intervals can be changed, and a shorter time causes a faster detection of a connection failure on the line connecting the given nodes.
- **Mode on demand.** In this mode, the nodes do not exchange Hello packets because it is assumed they have another way to prove that their connection is still alive and has not been lost.

Both modes enable the Echo Mode function. When using this function, node A sends echo packets to node B, and the latter resends them to node A. Such behavior guarantees that if node A does not receive the packets it sent from node B, it will assume that an error has occurred and the connection has been interrupted.

The BFD protocol works independently of other protocols. This means that even though protocols such as OSPF, EIGRP, or HSRP use their methods to detect the failure of links and nodes in the network, they can be reconfigured to use the BFD protocol for this failure detection. The latter will inform them about possible connectivity failures [7]–[9].

B. Fast network recovery

In this section, we will delve into the issue of fast network recovery mechanisms, so-called FRR mechanisms. We will explain the basic terms associated with this topic and give examples of some selected FRR mechanisms together with an explanation of their logic and functioning [5][4].

1) Operating principle

The goal and importance of FRR mechanisms are to redirect traffic to previously calculated and found backup routing paths in the event of a line or node failure in a given topology. The point is to reduce the time when there is no connectivity between individual nodes.

Calculations to find such a backup routing path occur when the router, which is not currently calculating the shortest paths according to the used routing protocol, is inactive. After detecting a line or node outage, the task of such a router is to redirect traffic to a pre-calculated backup routing path that corresponds to the failure of the given error. A router will route traffic to a given backup routing path until the topology in which this router is located reaches a state of convergence. It follows that if the given FRR mechanism is to be effective, the time required to redirect traffic to the pre-calculated backup routing path must be shorter than the time it takes for the given topology to reach the state of convergence [10], [11].

The effectiveness and efficiency of a given FRR mechanism can be measured by the level of error coverage that such an FRR mechanism guarantees [1]. The paper below describes the definition of the error coverage rate in more detail.

2) Repair coverage

Regarding fast forwarding packets to a backup routing path in case of a line or node failure in a given network, the repair coverage rate represents the success of finding a backup routing path by the given FRR mechanism in all possible scenarios. That is, how successful it is in protecting connectivity in the network during a possible line or node failure. This rate depends on the given FRR mechanism and the topology in which it is used.

Therefore, the degree of coverage of a given FRR mechanism's repair represents its effectiveness in dealing with various scenarios of line and node outages. The greater the degree of repair coverage guaranteed by the given mechanism, the more efficient it is. However, the rule mostly applies that the greater the degree of repair coverage guaranteed by the given FRR mechanism, the more computing resources it uses.

In some cases, the FRR mechanism can cover and correct all errors and outages in the topology. In such a case, the given FRR mechanism would have a repair coverage rate of 100%.

II. ANALYSIS OF EXISTING SOLUTIONS

This section covers FRR mechanisms, including LFA and RLFA, and SDN-based mechanisms such as OpenFlow Groups and Dynamic Protection with Quality of Alternative Paths. It also includes a summary of the analysis.

A. Loop-Free Alternates (LFA)

The Loop-Free Alternates FRR mechanism works by searching for an alternative next-hop router N that will provide an alternate routing path if router E on the main routing path fails. The alternative next-hop router N must be directly connected to the router S on which the given FRR mechanism runs. Only a router that does not cause a routing loop in the topology when it is used in an alternate path can become an LFA next-hop router N [12].

In the event of a failure or outage of the line along which the main routing path goes, the router S will record this failure and redirect the sending of packets to a pre-calculated backup routing path through the next-hop router N. This backup routing path will be used until the routing protocol running in the given topology calculates a new main routing path [13].

LFA enables two types of outage protection. The first is line failure protection, which can deliver packets safely to their destination via a backup routing path only if the line connecting router S to router E fails. If the entire router E fails, line protection cannot deliver packets. The second type of protection that LFA enables is protection against node failure. This protection can safely deliver packets to the destination via a backup routing path even if the entire E router fails.

The Loop-Free Alternates FRR mechanism is divided into two types according to how the backup next-hop router calculates it. A more detailed description of the functioning of individual types of LFA will now be given [1]:

- **Per-link LFA.** This method can only provide line protection. The backup router N found by this logic must be directly connected to the router E. It is a rule that for a given link SE, only one next-hop router N is found for all destinations that can be reached through this link.
- **Per-prefix LFA.** This method can provide both line and node protection. In this case, it does not apply that the backup router N must be directly connected to the main next-hop router E. The backup LFA router N will be calculated for each prefix for the target D on the main router S.

Since the calculation of the alternative path depends on the physical topology of the network connection, there may also be situations in which the LFA mechanism cannot provide a solution for finding a backup routing path. In the same way, the length of the calculation time of the backup routing path, if it exists and can be calculated, also depends on the topology. When comparing the calculation method of per-link LFA and per-prefix LFA, per-prefix LFA uses more computing resources to calculate backup routing paths.

As for the error coverage rate, the LFA mechanism reaches values from 60% to 90% depending on the topology in which it was used [14].

B. Remote LFA

The Remote LFA FRR mechanism uses specific terminology. For this reason, it is first necessary to explain the individual terms [13], [15], [16]:

- P-space = This space consists of all routers concerning the protected line that can be reached from the source router S by the current shortest routing paths without going through the protected line
- Q-space = This space consists of all routers concerning the protected line, from which it is possible to reach router E via the currently shortest routing paths without passing through the protected line
- Node PQ = This is a node that is, at the same time, a member of the P-space of the source router S and the Q-space of the router E concerning the protected line E
- Remote LFA node = This PQ node is suitable for use as an LFA router.

The LFA mechanism is especially effective in topologies with many connections between nodes. However, it has limitations in topologies such as a circle or a star. For this reason, the Remote Loop-free Alternates mechanism was created. This mechanism uses the tunneling technique to create new logical connections between nodes. This has the task of ensuring the possibility of access to remote LFA routers N. Thanks to this, the error coverage rate that this FRR mechanism should achieve should be increased compared to LFA, which does not support the creation of such tunnels. RLFA is used whenever finding a directly connected backup LFA router N for the source router S for a backup routing path to the destination router D is impossible.

The RLFA mechanism assumes that an entire node will never fail during the topology run. In the event of failure of the entire node, a situation may arise in the topology that will result in the creation of routing loops. For this reason, RLFA does not provide a 100% error coverage rate. There may also be situations in which it will not be possible to find any such router simultaneously being a part of P-space and Q-space. Thus, there will be no PQ node in the given topology scenario. This will also result in the failure of the calculation and finding of the backup routing path in the given topology using the RLFA mechanism [16].

As for testing the error coverage rate of the RLFA mechanism, the largest measured value reached 99.7%. Thanks to this, we can say that the RLFA has a significantly higher error coverage rate than the basic LFA [15].

In Fig. 1 below, the source router S wants to send packets to the destination router D. The SD link connecting these nodes has link protection provided. According to the terminology used in the context of this FRR mechanism, the P-space of the S router belongs to the nodes R1, R2, and R3. This is because the shortest paths from the source router S to other routers that do not pass through the protected link SD have the following form: S -> R1, S -> R1 -> R2, S -> R1 -> R2 -> R3.

In the Q-space for router D are routers R3 and R4. This is because the shortest paths to router D that do not pass through the protected SD link have the following form: R3 -> D, R4 -> R3 -> D. Router R2 is not a member of this Q-space because

there are two equivalent paths from it to router D. These equivalent paths have the following form: R2 -> R1 -> S -> D, R2 -> R3 -> R4 -> D. In case router S creates tunnel to router R2 and would send a packet directed to router D through it, router R2 could decide to send such a packet via R2 -> R1 -> S -> D, and thus a routing loop would be created in the topology.

The only router in the P-space of router S and the Q-space of router D concerning the protected SD link is router R3. For this reason, this router will be chosen as the end of the tunnel that router S will create in case the protected SD link fails. This means that router R3 becomes the RLFA router of router S for the destination routers R4 and D.

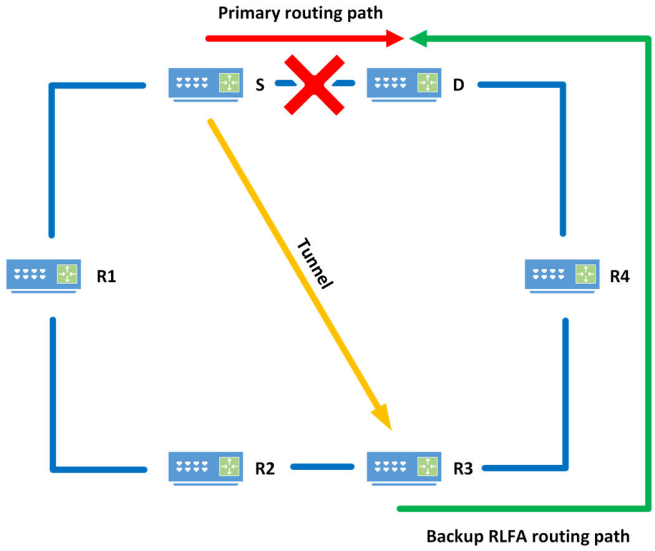

Fig. 1. Figure 1RLFA

C. OpenFlow Groups

The OpenFlow Groups mechanism is only related to SDN topologies.

This mechanism defines specific actions that dictate how network devices should process incoming data flows in the data plane. It organises these actions into groups, where each group consists of special records containing an ID, group type, counters, and a list of operations, referred to as a "bucket." If a group contains multiple "bucket" lists, they are called "action buckets." When a data flow matches a predefined rule, it is assigned to the corresponding group, and the appropriate operation from the assigned "bucket" list is applied [17].

We will now describe the currently existing four types of OpenFlow groups:

- **All Group.** This group is used for multicast or broadcast packets. The packet this group will process will be copied as many times as there are operations in the "bucket" list, and each operation will process exactly one copy of this packet.
- **Selected Group.** The selected group is used for load balancing. The packet selected for this group will be processed using only one list of operations. The switch itself decides which list of operations a given packet will process.
- **Indirect Group.** Only one operation list can exist in this group. The goal is to achieve coverage of the switch's most frequently used operations for the same next-hop when routing packets, which is intended to reduce the load on the switch's computing resources.
- **Fast Failover Group.** The packet selected for this group will be processed using only one list of operations. In this case, the list that will be the first to be in the "live" state will be selected. The watch port method determines whether the given list is in this state.

The Fast Failover group provides the highest error coverage among the different types of OpenFlow groups. It addresses connectivity failures in the data plane using a protection-based approach. Since backup routing paths are pre-established within this group before a link failure occurs, the SDN controller does not need to calculate and find them in real-time. This significantly reduces the time required to reroute data flow to an available backup path, ensuring faster recovery and improved network resilience [18].

We will now describe the procedure for evaluating packet routing:

- **The flow rule table evaluates the incoming packet.** At this point, the switch determines which group will route the packet. In this case, it will be a Fast Failover group.
- **Applying an operation to a given packet.** Based on the current state of the ports, the list that will be the first to be "live" is selected from the list of operations. Subsequently, the packet is sent through the first available output port.

Since backup routing paths apply to a given topology, a backup routing path that could be implemented in a Fast Failover group cannot be found. This FRR mechanism does not provide a 100% protection coverage rate.

D. Fast forwarding of data flow in SDN topologies

This section will give a practical example of solving a line failure and improving connectivity in SDN topology, thanks to the FRR mechanism Dynamic Protection with Quality of Alternative Paths, abbreviated DPQoAP. To simplify the understanding of this solution, the components and principles of operation of SDN topologies will also be explained in this part of the work. The information in this part of the work will be based on the Fault Tolerance in SDN Data Plane Considering Networking and Application-Based Metrics [19].

1) Planes in SDN topologies

Unlike the usual approach to routing in topologies, SDNs provide a different view of the entire topology, thanks to the SDN controller, which centralises the logic of the operation of the given topology in one place. This makes it possible to solve line or node outages in the three different levels found in SDN topologies. Each level is tasked with solving different problems and making different functions available. We will now describe these planes in more detail [20]:

- Application plane. This plane contains all SDN applications used by the given topology. Applications responsible for topology management and security are broadcast here. The most related issues to this layer are SDN application failures.

- Control plane. This part of the SDN network architecture overviews the entire topology and hardware resources for SDN applications. Problems related to this layer include connection failure between the network device in the data layer and the controller.
- The data plane. It is made up of network devices, such as OVS switches, which are responsible for forwarding data flows according to the rules provided to them by the SDN controller and found in their Flow Table routing tables. Problems related to this level include failure of the line or the network device itself.

2) Ways to solve connectivity failure in the data plane

In SDN topologies, line or node failures in the data plane are solved in two ways. It is either restoration or protection. We will now describe the functioning of these approaches to the solution [21]:

- **Recovery.** Using this logic, the network device in the data plane will send a message about this fact to its SDN controller when it detects a link failure. The SDN controller must then pass the information on to the SDN application, which recalculates and finds new routing paths that will not use the faulty line. The SND controller then sends these records of new routing paths to the given network device, adjusting its routing tables according to these records.
- **Protection.** Redirecting the data flow through the topology in this solution follows a similar approach to traditional protection mechanisms. However, in this case, the node detecting the link failure is not responsible for finding backup routing paths. Instead, this task is handled by the SDN application, which communicates with the SDN controller to determine the optimal alternative path.

The protection method is preferred for fast forwarding data sent by the topology, as it can forward the data flow faster. In the case of protection, backup routing paths are already prepared in advance, and there is no need to wait for SDN application calculations.

3) Dynamic Adaptive Streaming over HTTP

DASH is a protocol for streaming and delivering multimedia content such as video or audio through the HTTP protocol. It is commonly used for streaming video content over the Internet. DASH's design enables it to provide users with a high-quality streaming experience. This is achieved because DASH can adapt to the conditions in the topology and the state of the devices in it in real time [22].

E. Dynamic Protection with Quality of Alternative Paths

The authors' main intention in [2] was to create a new FRR mechanism called Dynamic Protection with Quality of Alternative Paths. It should be able to calculate and find the most efficient routing path using knowledge of the congestion of individual lines in a given topology. The result should be the maintenance of the same level of QoS throughout the topology. Such a mechanism should be able to maintain the same level of QoE of users in the event of line outages during video transmission using dynamic adaptive streaming over HTTP, DASH for short.

Subsequently, the effectiveness of this FRR mechanism was tested against several other mechanisms for solving the connectivity failure in the topology.

Although DASH is proposed to improve user QoE during changing topology characteristics, it cannot work well with congested lines. As part of this work, the authors changed the evaluation of errors in the data plane so that a large line overload is also considered a line failure.

1) Data plane design

When designing the data plane, the authors implemented several modules that offer unique functionality. Three of these modules implement different connectivity solutions in the event of a line failure; one is used to detect a line failure, and the last is for sending data streams through the topology. We will now describe the individual modules used by the authors of the study in more detail [2]:

- **DPQoAP module.** The task of this module is to find the best backup routing path concerning the latency parameter. It achieves this by combining solutions for connectivity failures at the data plane with protection and recovery. With the protection method, the time required to redirect the data flow is shorter, as the SDN controller doesn't need to intervene. The authors of the work chose OpenFlow Groups as the FRR mechanism responsible for backup routing paths selected by the protection method. In this case, however, the operation from the Fast Failover group is selected based on information about the status in the topology, which was obtained at the beginning of the data flow transfer. However, since this state constantly changes during the topology, the operation selected in this way may not always be the most effective, even though it can ensure the delivery of the data flow to its destination. For this reason, the authors decided to include a recovery method that considers the current state of the topology in the outage solution, ensuring a more adaptive and efficient response to failures.
- **DASH module.** This module calculates the necessary metrics, buffer level, and bitrate value to determine video quality. These data are further used to compare changes in given metrics in case of failure or line overload. Achieving these values is possible thanks to the widely used Javascript library for measuring client metrics, DASH.js.
- **Line Congestion Detection Module based on BFD.** The authors designed this module to use the method of protection against failure by recovery. They decided to do so because video streaming can withstand connectivity failures in the topology, even for a few seconds, thanks to a mechanism that stores data in a buffer memory. For this reason, it is advantageous to use a recovery method that considers the topology's current state to find a backup routing path. Even though this method takes longer than the protection method, the authors could neglect it in this case, thanks to the already-mentioned use of buffer memory.
- **Static protection module.** The authors used the OpenFlow Groups mechanism in this module to provide backup FRR routing paths. This mechanism enables

efficient traffic redirection by predefining alternative paths that can be used immediately in case of a failure. The module employs a specific algorithm to determine the most suitable backup path, ensuring minimal disruption and improved network resilience.

- **Recovery model.** The recovery model handles connectivity failure resolution in the data plane by following a structured process. The steps performed in this module are as follows:
  - **Sending link failure information to the SDN controller.** Based on the current state of the SDN topology, the controller calculates new routing paths for all flows affected by this outage. The number of hops is the deciding factor when choosing the routing paths. The fewer hops packets have to make, the more likely this routing path will be used.
  - **Sending new routing rules to affected network devices.** In this step, the SDN controller sends new routing records to the network devices affected by the outage.

F. Summary of analysis

SDN brings many advantages and challenges in quickly restoring communication after a node or link failure. The main problem areas include [23]–[25]:

1) Control Latency and Reaction Time

The central control plane (SDN controller) can become a bottleneck since it needs to process many routing updates after a failure.

Communication between switches and the controller (Southbound API) can also affect response time, especially in large-scale topologies.

2) SDN Controller Overload

When multiple nodes or links fail simultaneously, the SDN controller must adapt quickly, which can overload the CPU or memory.

The response speed depends on the efficiency of path computation algorithms (e.g., Dijkstra or Constrained Shortest Path First - CSPF).

3) Limited Scalability

Managing many switches in SDN networks (e.g., WANs, data centers) can be challenging, leading to delays in restoring connectivity.

Decomposing the network into multiple SDN domains can help, but it increases coordination complexity among controllers.

4) Dynamic Flow Reconfiguration

When a failure occurs, SDN switches must immediately update their flow tables. If the table is large, it may take time for changes to be applied.

Some SDN switches have limited storage capacity for flow rules, leading to frequent record swapping and additional latency.

5) Consistency and Coordination in Distributed Control

In hierarchical or distributed SDN (e.g., multi-controller solutions), coordination among controllers can be slow, causing inconsistent routing.

Preventing loops and ensuring routing consistency may require additional mechanisms, such as eventual consistency or state synchronisation.

6) Security Issues in Recovery Mechanisms

After a failure, traffic may be rerouted through less secure or congested paths.

SDN is vulnerable to attack surface expansion. If an attacker manipulates the control plane, they could create false failures or modify flow rules (e.g., via SDN hijacking attacks).

7) Lack of Immediate Reactive Mechanisms

OpenFlow and similar SDN protocols are primarily designed for centralised control, not ultra-fast failure recovery.

Mechanisms like Fast Reroute (FRR) or precomputed backup paths can help, but increase the burden on switch capacity.

8) Integration with Traditional Network Mechanisms

Hybrid networks (SDN + traditional networks) may experience incompatibilities between dynamic SDN policies and classic routing protocols (e.g., OSPF, BGP).

Mechanisms such as BGP-SDN hybrid routing can help, but they introduce additional latency in reconfiguration.

9) Potential Solutions for Faster Communication Recovery

Proactive Backup Path Installation – Precompute alternative routes and pre-install flow rules on switches.

Local Reconfiguration (Fast Failover Groups in OpenFlow) – Switches can autonomously reroute traffic upon failure detection without controller intervention.

Hierarchical or Distributed Controllers – Distribute control across multiple SDN domains to reduce latency.

AI-Based Failure Prediction – Machine learning models can predict failures and allow faster reconfiguration.

Network Function Virtualization (NFV) and SDN Integration – Dynamically reroute traffic through virtualised network functions for better failure recovery.

III. SIMULATIONS

This section will focus on testing SDN topology in Mininet against link failure. Simulations will show unacceptably long convergence duration in SDN networks, in which no FRR mechanism was used to reduce the duration of connection failure. We have made several topologies and tests but will only present one topology in this paper.

A. Mininet

Mininet is free software that can simulate complex topology connections within SDN networks. In these connections, we can use the building blocks that the Mininet program offers, which are, among other things, end devices, OVS switches, and controllers. For the topology to be complete, these individual building blocks, i.e. nodes, must be interconnected by lines [26].

Mininet gives us a choice of two variants for creating our topology. The topology can be created by programming a script in Python, where we define individual nodes, their properties, and the connections between them. Or we can use the second option, the graphic editor MiniEdit, which contains Mininet.

Similar programs that also serve to simulate network traffic and are widely used are, for example, GNS3 or Cisco Packet Tracer. However, we chose the Mininet program for the needs of our work, and the reasons why we did so are listed below:

1. Specializes in working with SDN networks. This is a great advantage since this work tests the work of the SDN network.

2. It enables the simple setting of parameters for individual nodes, lines, and topology. Without this, the simulation itself would be impossible.

3. It supports easy operation with the Linux operating system. Since the other programs we used for simulations also run on Linux operating system distributions, it was important that the simulation program could also run on these distributions.

B. *OpenDaylight*

The OpenDaylight driver is an open-source Java virtual machine application that can be run from any operating system that supports Java. It collects information from the SDN topology and uses it in its analysis algorithms. Applications that cooperate with the controller create routing rules for a given topology [27].

The driver exposes an open northbound API that applications use to communicate with. These applications contain the logic of algorithms for creating routing rules. Depending on the implementation, these may differ and thus change the routing of packets in the topology. These routing rules are subsequently delivered to the OVS switches via the southbound API, where multiple protocols are supported to deliver these rules. The protocol used in this work is OpenFlow 1.0.

Other SDN drivers used are Ryu, Floodlight, and ONOS drivers. In this work, however, we decided to use the OpenDaylight driver in version Boron 0.5.0. This is because it is an open-source solution that is easily accessible, and manuals with information on its configuration and use are easy to find.

For the correct functioning of the SDN driver OpenDaylight Boron 0.5.0, which was used in this work, it was necessary to install the following extensions in its interface: odl-restconf, old-l2switch-switch, odl-mdsal-apidocs, and odl-dlux-all. The command used to install the extension has the following form, and we enter it in the SDN controller interface after its initialization: "feature: install x", where x is the name of the extension we want to install.

C. *Testing topology*

A topology comprises eight end devices, ten OVS switches and one ODL controller. In this case, every two end devices connect to one switch in the access layer. The switches in this layer are then redundantly connected to the switches in the access layer. In Fig. 2 below, we see this is not an each-to-each type of connection. Still, the redundant connections are divided in the topology as if into left and right sides by an imaginary vertical line running through the center of the topology. The remaining switches in the distribution layer are subsequently connected redundantly to both switches in the main layer, which also contain a connection between them. All switches are subsequently connected to the ODL controller.

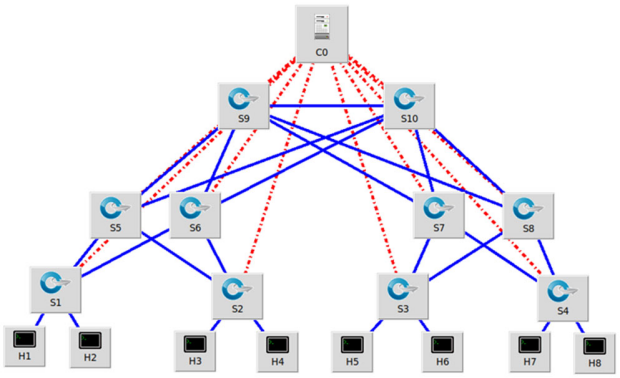

Fig. 2. Testing topology

D. *Main routing path*

Fig. 3 below shows the main routing path from node H1 to node H2 found by the ODL controller after running the topology in the Mininet simulation program. We found the route of the routing path thanks to the entries in the Flow table on the individual OVS switches.

The selected routing path is H1 -> S1 -> S5 -> S9 -> S8 -> S4 -> H8.

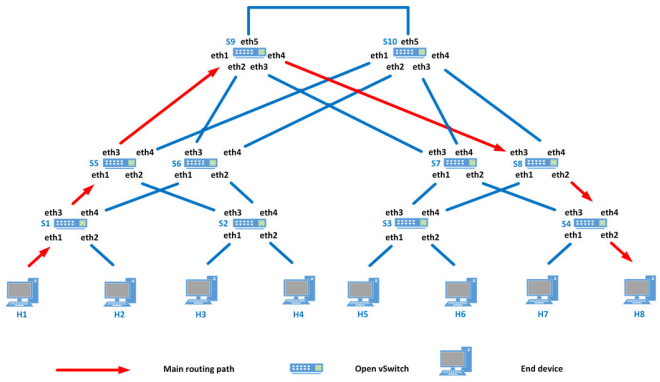

Fig. 3. Main routing path

E. *New routing paths after line failure*

Fig. 4 below shows the routing path from node H1 to node H8 found by the ODL controller after a link failure between switches S5 and S9 that was part of the main routing path. The selected routing path looks like this: H1 -> S1 -> S5 -> S10 -> S9 -> S8 -> S4 -> H8.

Interestingly, even though all lines are identical in terms of their properties, the ODL controller did not choose the shortest path from switch S10 to node H8 through the eth4 line. This line would send packets directly to switch S8, from where they could be forwarded to switch S4 and directly to their destination. However, he decided to send these packets via the eth5 interface to the S9 switch, from where they were subsequently forwarded to the S8 switch. From there, they reached their destination via the route described above.

Since no FRR mechanism is used in this topology, there is no pre-calculated backup routing path, and the controller must run calculations to find a new routing path. The duration of these calculations in the middle topology, i.e., the convergence time, ranges from 3.964 to 6.169 seconds, with an average value of 4.490 seconds. Whenever a line or node in the currently used routing path fails, connectivity from node H1 to H8 will be lost for an average of 4.490 seconds. Since the UDP protocol is used for packet transmission, which does not verify packet delivery, all packets sent during this period are lost.

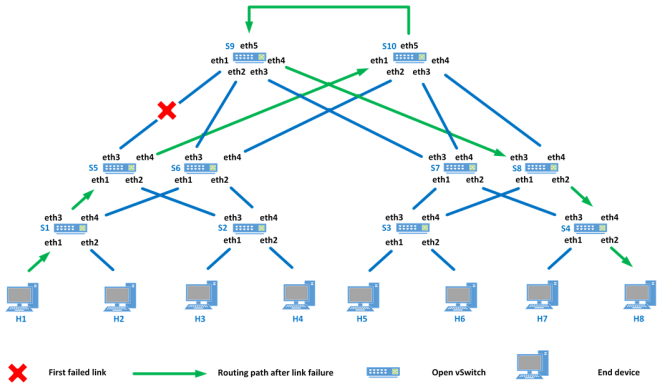

Fig. 4. Routing path after one line failure

In Fig. 5, we can see that after the links between switches S5 and S9 and between S9 and S10 went down, the ODL controller finally decided to choose the routing path to H8, which was indeed the shortest this time. The routing path shown in the figure has the following form: H1 -> S1 -> S5 -> S10 -> S8 -> H8.

At the same time, this routing path applies to all measurements conducted in the paper. These measurements reflect the results of sending and receiving packets, with the occurrence of two line failures, each causing a loss of connectivity in the topology between nodes H1 and H8 for an average of 4.490 seconds, during which a new routing path had to be found.

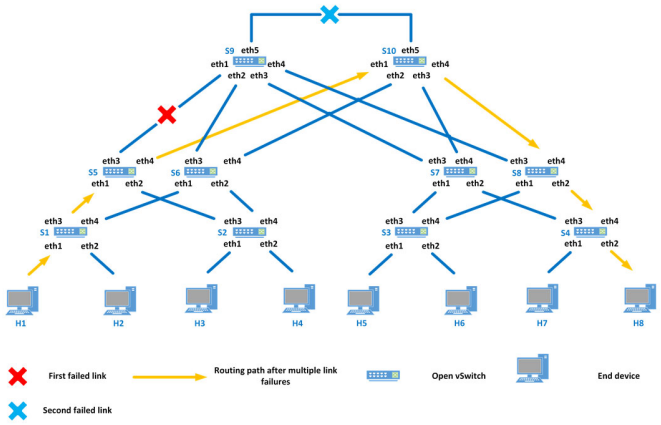

Fig. 5. Routing path after the outage of two lines

F. Simulation results

The Table I below shows the results of individual simulation replications performed on the topology. These results are almost indistinguishable from those achieved by the small topology. In both cases, the values of sent, received, and lost packets move in almost the same intervals and reflect a similar handling of the situation in the event of multiple line failures in the topology. These are, therefore, equally unacceptable values, which would cause enormous packet losses and great unreliability in data transmission in the used topologies when the current solution to line and node outages is implemented.

TABLE I. SIMULATION RESULTS

Replication	Sent Packets	Received Packets	% Received Packets	Lost Packets	% Lost Packets
1.	21758	10680	49,09	11078	50,91
2.	21851	9868	45,16	11983	54,84
3.	21768	10258	47,12	11510	52,88
4.	21781	10174	46,71	11607	53,29
5.	21984	10228	46,52	11756	53,48
6.	21752	10596	48,71	11156	51,29
7.	21753	10489	48,22	11264	51,78
8.	21641	10317	47,67	11324	52,33
9.	22193	10986	49,52	11208	50,85
10.	21621	9829	45,46	11792	54,54

The graph below shows the percentage of received and lost packets in each test replication. The X-axis represents the replications (1st to 10th), while the Y-axis shows the percentage of received and lost packets.

- The yellow line represents the % of received packets, ranging between approximately 45% and 49.5%.

- The orange line represents the % of lost packets, ranging between 50.85% and 54.84%.

The graph (Fig. 6) indicates a significant packet loss (above 50%) in all cases, with some variations between replications. Some replications show higher losses than others (e.g., the 2nd and 10th replications).

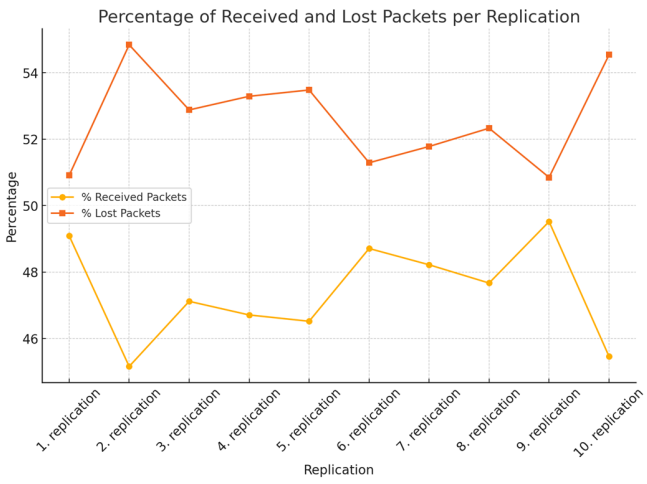

Fig. 6. Percentage of received and lost packets per replication (test round)

The results of the simulations demonstrate that in the event of multiple line outages, more than 50% of the transmitted

packets were lost due to a loss of connectivity to the destination. During 15 seconds of packet generation, out of approximately 22,000 sent packets, no more than 11,000 successfully reached their destination. This packet loss level is unacceptable and would cause severe issues in real-world deployments, affecting data transmission, user QoE, and overall network reliability.

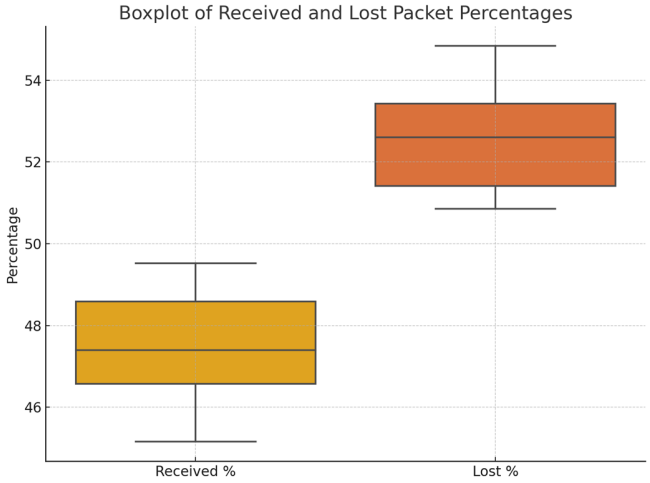

Fig. 7. Boxplot of Received and Lost Packet Percentages

To further the analytical part of the evaluation, we created a boxplot (Fig. X) that shows the dispersion of the received and lost packet rates across all ten simulation replications.

The boxplot shows the following:

- Received %: Most values ranged between ~45% and 49.5%, with some variability and a few lower values as potential outliers.
- Lost %: The packet loss rate was above 50%, consistent with the observations in the table.

These findings emphasise the critical need for Fast Reroute (FRR) mechanisms in practical network topologies. FRR mechanisms can significantly reduce the time required to forward packets to a new routing path by precomputing and establishing backup paths in advance. This proactive approach ensures that network connectivity is maintained despite failures, preventing prolonged service disruptions and improving network resilience.

IV. Future work

The results presented in this paper highlight several persistent limitations in current FRR mechanisms across both SDN environments and traditional IP-based networks. While many existing solutions focus on reactive failover strategies—triggered after a failure has already occurred—these often suffer from significant convergence delays, high packet loss, or limited scalability in complex topologies.

Motivated by these challenges, we are developing a new conceptual framework for a proactive FRR mechanism. Unlike conventional methods, this approach envisions leveraging neural networks to anticipate potential failures based on historical and real-time telemetry data. The goal is to shift from a purely reactive paradigm toward a predictive and preemptive one, where backup paths can be dynamically selected or adjusted before failures occur, minimising their impact on service continuity.

Importantly, this concept is being designed with cross-domain applicability in mind, aiming to integrate seamlessly into SDN architectures and traditional network infrastructures where dynamic routing protocols (e.g., OSPF, BGP) are used. The envisioned system considers how AI-based failover logic could be embedded into existing control frameworks or realised through lightweight, modular enhancements to controller logic.

V. Conclusion

This paper emphasises the necessity of efficient failure recovery mechanisms in SDN environments. Traditional approaches often struggle to balance rapid failover with optimal path selection, leading to increased packet loss and latency during network failures. Analysing various Fast Reroute (FRR) mechanisms and SDN-based recovery solutions, such as OpenFlow Groups and Dynamic Protection with Quality of Alternative Paths (DPQoAP), we demonstrated the benefits of integrating proactive and reactive strategies to improve fault tolerance.

Our findings highlight the importance of tailoring failure recovery mechanisms specifically for SDN environments, where centralised control and dynamic traffic management offer new opportunities for optimisation. As part of our future research, we aim to design an AI-enabled proactive FRR mechanism. This approach will reduce failover time, optimise backup path selection, and improve overall network stability in SDN or IP architectures.

VI. Acknowledgement

This research was funded by the Slovak Grant Agency VEGA project Fast Reroute, No. 1/0316/24.

VII. References

[1] M. Shand and S. Bryant, “IP Fast Reroute Framework,” RFC5714, 2010, http://www.rfc-editor.org/rfc/rfc5714.txt, ISSN: 2070-1721.

[2] B. Yamansavascilar, A. C. Baktir, A. Ozgovde, and C. Ersoy, “Fault Tolerance in SDN Data Plane Considering Network and Application Based Metrics,” Dec. 2019, doi: 10.1016/j.jnca.2020.102780.

[3] A. Malik, B. Aziz, M. Adda, and C. H. Ke, “Smart routing: Towards proactive fault handling of software-defined networks,” *Computer Networks*, vol. 170, p. 107104, Apr. 2020, doi: 10.1016/J.COMNET.2020.107104, ISSN: 1389-1286.

[4] R. Petija, M. Michalko, F. Jakab, and P. Fecilak, “Convergence of Routing Protocols in Real and Simulated Environments,” in *ICETA 2018 - 16th IEEE International Conference on Emerging eLearning Technologies and Applications, Proceedings*, 2018, pp. 425–430, doi: 10.1109/ICETA.2018.8572184, ISBN: 9781538679142.

[5] A. Kamisinski, “Evolution of IP fast-reroute strategies,” in *Proceedings of 2018 10th International Workshop on Resilient Networks Design and Modeling, RNDM 2018*, 2018, pp. 1–6, doi: 10.1109/RNDM.2018.8489832, ISBN: 9781538670309.

[6] D. Katz and D. Ward, “Bidirectional Forwarding Detection,” RFC5880, 2010, doi: 10.1017/CBO9781107415324.004, ISBN: 9788578110796.

[7] S. M. Kim, G. Yang, C. Yoo, and S. G. Min, “BFD-based link latency measurement in software defined networking,” in *2017 13th International Conference on Network and Service Management, CNSM 2017*, 2018, vol. 2018-Janua, pp. 1–6, doi: 10.23919/CNSM.2017.8256023, ISBN: 9783901882982.

[8] C. Pignataro, D. Ward, N. Akiya, M. Bhatia, and S. Pallagatti, “Seamless Bidirectional Forwarding Detection (S-BFD),” RFC Editor, 2016, ISSN: 2070-1721.

[9] Di. Siqueira, T. Pinheiro, J. Dantas, and P. MacIel, “Dependability evaluation in a convergent network service using BGP and BFD

protocols," *Conf Proc IEEE Int Conf Syst Man Cybern*, vol. 2019-October, pp. 2378–2383, Oct. 2019, doi: 10.1109/SMC.2019.8914368, ISBN: 9781728145693.

[10] M. Gjoka, V. Ram, and X. Yang, "Evaluation of IP Fast Reroute Proposals," in *2007 2nd International Conference on Communication Systems Software and Middleware*, 2007, pp. 1–8, doi: 10.1109/COMSWA.2007.382443, ISBN: 1-4244-0613-7.

[11] S. Cevher, M. Ulutas, and I. Hokelek, "Performance evaluation of Multiple Routing Configurations," in *2013 21st Signal Processing and Communications Applications Conference (SIU)*, 2013, pp. 1–4, doi: 10.1109/SIU.2013.6531353, ISBN: 978-1-4673-5563-6.

[12] A. A. Eluheshi, M. Mansour, and N. Ben Saud, "Optimizing Network Resilience with Segment Routing: A Comparative Study of SR TI-LFA and rLFA," *2024 IEEE 4th International Maghreb Meeting of the Conference on Sciences and Techniques of Automatic Control and Computer Engineering, MI-STA 2024 - Proceeding*, pp. 488–493, 2024, doi: 10.1109/MI-STA61267.2024.10599752, ISBN: 9798350372632.

[13] P. Sarkar, S. Hegde, C. Bowers, H. Gredler, and S. Litkowski, "Remote-LFA Node Protection and Manageability," RFC8102, 2017, ISSN: 2070-1721.

[14] S. Litkowski, B. Decraene, C. Filsfils, K. Raza, M. Horneffer, and P. Sarkar, "Operational Management of Loop-Free Alternates," RFC7916, 2016, ISSN: 2070-1721.

[15] S. Bryant, C. Filsfils, S. Previdi, M. Shand, and N. So, "Remote Loop-Free Alternate (LFA) Fast Reroute (FRR)," RFC7490, 2015, ISSN: 2070-1721.

[16] L. Csikor and G. Rétvári, "On providing fast protection with remote loop-free alternates," *Telecommun Syst*, vol. 60, no. 4, pp. 485–502, Dec. 2015, doi: 10.1007/s11235-015-0006-9, ISSN: 1018-4864.

[17] A. M. Bahaa-Eldin, E. E. E. Eldessouky, and H. Dag, "Protecting openflow switches against denial of service attacks," *Proceedings of ICCES 2017 12th International Conference on Computer Engineering and Systems*, vol. 2018-January, pp. 479–484, Jul. 2017, doi: 10.1109/ICCES.2017.8275355, ISBN: 9781538611913.

[18] M. Chiesa, A. Kamisinski, J. Rak, G. Retvari, and S. Schmid, "A Survey of Fast-Recovery Mechanisms in Packet-Switched Networks," *IEEE Communications Surveys and Tutorials*, vol. 23, no. 2, pp. 1253–1301, Apr. 2021, doi: 10.1109/COMST.2021.3063980, ISSN: 1553877X.

[19] W. Braun and M. Menth, "Scalable resilience for Software-Defined Networking using Loop-Free Alternates with loop detection," in *Proceedings of the 2015 1st IEEE Conference on Network Softwarization (NetSoft)*, 2015, pp. 1–6, doi: 10.1109/NETSOFT.2015.7116180, ISBN: 978-1-4799-7899-1.

[20] A. H. Abdi *et al.*, "Security Control and Data Planes of SDN: A Comprehensive Review of Traditional, AI, and MTD Approaches to Security Solutions," *IEEE Access*, vol. 12, pp. 69941–69980, 2024, doi: 10.1109/ACCESS.2024.3393548, ISSN: 21693536.

[21] N. Khan, R. bin Salleh, A. Koubaa, Z. Khan, M. K. Khan, and I. Ali, "Data plane failure and its recovery techniques in SDN," *Journal of King Saud University - Computer and Information Sciences*, vol. 35, no. 3, pp. 176–201, Mar. 2023, doi: 10.1016/J.JKSUCI.2023.02.001, ISSN: 22131248.

[22] G. Xie, X. Jin, L. Xie, and H. Chen, "A Quality-driven Bit Rate Adaptation Method for Dynamic Adaptive Streaming over HTTP," *2018 10th International Conference on Wireless Communications and Signal Processing, WCSP 2018*, Nov. 2018, doi: 10.1109/WCSP.2018.8555579, ISBN: 9781538661192.

[23] N. Khan, R. bin Salleh, A. Koubaa, Z. Khan, M. K. Khan, and I. Ali, "Data plane failure and its recovery techniques in SDN: A systematic literature review," *Journal of King Saud University - Computer and Information Sciences*, vol. 35, no. 3, pp. 176–201, Mar. 2023, doi: 10.1016/J.JKSUCI.2023.02.001, ISSN: 1319-1578.

[24] J. Chen, J. Chen, J. Ling, J. Zhou, and W. Zhang, "Link Failure Recovery in SDN: High Efficiency, Strong Scalability and Wide Applicability," *Journal of Circuits, Systems and Computers*, vol. 27, no. 6, Jun. 2018, doi: 10.1142/S0218126618500871, ISSN: 02181266.

[25] T. Semong *et al.*, "A review on Software Defined Networking as a solution to link failures," *Sci Afr*, vol. 21, p. e01865, Sep. 2023, doi: 10.1016/J.SCIAF.2023.E01865, ISSN: 2468-2276.

[26] "Mininet: An Instant Virtual Network on Your Laptop (or Other PC) - Mininet." [Online]. Available: https://mininet.org/. [Accessed: 03-Mar-2025].

[27] "OpenDaylight." [Online]. Available: https://www.opendaylight.org/. [Accessed: 03-Mar-2025].