

Route Convergence Optimization in the SWITCH Network

Master Thesis

Author: David Carminati

Tutor: Tibor Schneider

Supervisor: Prof. Dr. Laurent Vanbever

Industry Partner: Alexander Gall

Industry Partner: Simon Leinen

April 2021 to November 2021

Abstract

SWITCH - the provider for Swiss universities - experienced a link failure to DE-CIX, a large internet exchange point. The following migration of 250'000 prefixes lasted over two minutes, causing traffic to those destinations to be lost. We analysed SWITCHes configuration and built a minimal model of their topology in GNS3, a network emulation software. Together with experiment data from that model and log data analysis from the real network, it was concluded that the BGP withdrawal messages contributed the most delay to the convergence.

The configuration was then altered so that no next-hop-self would be used in the BGP setup by the border routers and the links to neighbours were added as OSPF passive links. This way only a single link-state update could invalidate all prefixes learned from that IXP. This reduced the convergence time from 65.9 seconds on average to 51.6 seconds in the real network. We now propose to improve the configuration even further by propagating the alternate BGP paths that are already learnt in the network, which would enable sub-second convergence for over 50% of the prefixes learnt at DE-CIX.

Contents

| | | |
|----------|---|-----------|
| 1 | Introduction | 1 |
| 1.1 | Motivation | 2 |
| 1.2 | Task and Goals | 2 |
| 1.3 | Overview | 2 |
| 2 | Background and Related Work | 4 |
| 2.1 | Background | 4 |
| 2.1.1 | Border Gateway Protocol (BGP) | 4 |
| 2.1.2 | Open Shortest Path First (OSPF) | 7 |
| 2.1.3 | GNS3 | 8 |
| 2.2 | Related Work | 8 |
| 2.2.1 | Changes to the BGP Protocol | 8 |
| 2.2.2 | Configuration Optimizations | 9 |
| 2.2.3 | External Changes | 9 |
| 3 | Problem Analysis | 10 |
| 3.1 | Basic Steps of Link Failure Convergence | 10 |
| 3.1.1 | Failure Detection | 11 |
| 3.1.2 | Failure Propagation | 11 |
| 3.1.3 | Routing Recalculation | 12 |
| 3.1.4 | Updating of RIB and FIB | 12 |
| 3.2 | SWITCHes Setup | 12 |
| 3.2.1 | General Overview | 12 |
| 3.2.2 | Router Configuration | 13 |
| 3.3 | Log Data Analysis | 14 |
| 3.3.1 | Preliminary Analysis by SWITCH | 14 |
| 3.3.2 | BMP Log Analysis | 15 |
| 3.4 | Conclusion on the Problem Analysis | 15 |
| 4 | Solution Design | 17 |
| 4.1 | OSPF Link Failure Distribution | 17 |
| 4.2 | BGP Next-Hop Address Tracking | 17 |
| 4.3 | Prefix Independent Convergence (PIC) | 18 |
| 5 | Evaluation | 19 |
| 5.1 | GNS3 | 19 |
| 5.1.1 | Topology | 19 |
| 5.1.2 | Configuration | 20 |

| | | |
|----------|---|-------------|
| 5.1.3 | Measurement Method | 21 |
| 5.1.4 | Side-Effect of the Measurement Method on the Convergence | 22 |
| 5.1.5 | Original Setup | 22 |
| 5.1.6 | OSPF Distribution | 24 |
| 5.1.7 | Effect of a slow Route-Reflector-Client on the Convergence Time | 25 |
| 5.2 | SWITCH Production Network | 25 |
| 5.2.1 | Measurement Method | 25 |
| 5.2.2 | Original Setup | 29 |
| 5.2.3 | OSPF Distribution | 31 |
| 5.3 | Additional Findings | 33 |
| 5.3.1 | The Effect of different Adapter Types on the Bandwidth in GNS3 | 33 |
| 6 | Outlook | 37 |
| 6.1 | Further Practical Improvements for SWITCH | 37 |
| 6.2 | Possible Improvements by Manufacturers | 37 |
| 6.2.1 | Extension of PIC | 38 |
| 6.2.2 | New FIB Architecture | 38 |
| 6.3 | Improving the Convergence for Peers at an IXP | 39 |
| 7 | Summary | 40 |
| | References | 41 |
| A | OSPF Trees for additional Exit Nodes | I |
| B | BGP Session Tracking | IV |
| B.1 | Time of First and Last BGP UPDATE Messages in GNS3 without Flows | IV |
| B.2 | Time of First and Last BGP UPDATE Messages in GNS3 with Flows | VII |
| B.3 | Time of First and Last BGP UPDATE Messages in GNS3 with a Slow Route- Reflector-Client | X |
| C | Flow Graphs | XIII |
| C.1 | GNS3 | XIII |
| C.1.1 | Original Setup per Prefix Graphs | XIII |
| C.1.2 | OSPF Distribution Setup per Prefix Graphs | XVII |
| C.2 | SWITCH Production Network | XXI |
| C.2.1 | Original Setup, additional per Injection Point Graphs | XXI |
| C.2.2 | Original Setup, additional per Prefix Graphs | XXIV |
| C.2.3 | OSPF Distribution, additional per Injection Point Graphs | XXVIII |
| C.2.4 | OSPF Distribution, per Prefix Graphs | XXXI |

Chapter 1

Introduction

The internet is a network of networks, which are called Autonomous Systems (AS). Such an AS can be a provider, an institution or another entity. Each of them is assigned to one or multiple IP prefixes [1]. Every connection between two ASes consists of at least one router on each side and a link between them. To be able to exchange data over the boundaries of an AS, it is necessary that they tell each other which prefixes can be reached through them. This is done with the Border Gateway Protocol (BGP), which glues them together [43]. Both routers and links can fail or be taken down for maintenance at any point, interrupting the connection. The convergence process after such an interruption can be lengthy, depending on the number of prefixes that had been announced by the AS to which the connection was interrupted. This is leading to downtime, lost traffic and therefore lost revenue, even though routing protocols are made to be resilient against such changes. Especially BGP as a distance-vector protocol can take several minutes to converge as it needs to propagate the updates on a per-prefix basis [38, 41, 28].

This convergence process can take so long because BGP needs to recompute the path for each prefix. If it can find one, it will advertise it to its neighbours. If it is not able to find one, it will withdraw it. This message will then be sent individually to each peer, which then needs to recalculate its route again, based on the newly learnt information and propagate it further. The state of the routing table of different routers can differ significantly while this information is floating around and new routes are being calculated. It is therefore common that traffic is black-holed as it is sent to routers that no longer know where to route it to, that loops are formed when for example one router is still using the old route and another one on that path is trying to use a new path that leads through the first router. It is also common that links are congested by looping packets and non-optimal routing paths [19, 44, 35]. Those problems can be even further aggravated if the link that failed was used to connect to an Internet Exchange Point (IXP). The failure of such a link will not only interrupt a single BGP session but many. It is therefore likely that many prefixes were learnt from more than one peer. The border router will consequently have learnt multiple routes to the same prefix and send out unnecessary updates once the best route is deleted, only to delete it as well after a moment as it has been lost as well by the link failure. This increases the number of messages that BGP is exchanging and subsequently the computational expenses of downstream routers.

In this thesis, we will analyse the failure of the link between SWITCH and DE-CIX, which is exactly such a connection to an IXP and the entailing convergence in their network to get a better understanding of the process and its consequences on their customers.

1.1 Motivation

SWITCHlan is the networking part of SWITCH and provides internet connectivity for Swiss universities[11]. SWITCH itself is a non-profit foundation that was founded in 1987 to provide the necessary basis in telecommunication and computing for the teaching and research community in Switzerland[14].

On the 12th of January 2021, SWITCH witnessed by chance the failure of the 100Gbit/s optical link between one of their border routers and DE-CIX. The connection to the prefixes that were lost, was interrupted for over two minutes, according to SWITCHes own analysis of the event. The border router that connected the IXP to their network needed to withdraw 250'000 prefixes, which then had to be processed by the other routers in the network before traffic could fall back to use the default routes via their transit-providers. This means that their customers experienced a lengthy disruption to a significant part of the internet. SWITCH realised that this was a common behaviour of their network that has gone unnoticed until that moment as network monitoring is still limited in its capability to detect every possible problem [29, 42].

It is now important to get a better understanding of the processes during this convergence period to be able to locate the main contributors and provide configuration changes to alleviate them so that the network can converge faster after future failures.

1.2 Task and Goals

The following steps have to be completed to gain a better understanding of the problem and to be able to develop a solution:

- First, we examine the current setup of SWITCHes network. For this, we take a look at their topology, their configurations and get an understanding of how the routes are distributed in their network
- We analyse the logs from a link failure to DE-CIX to gain a deeper understanding of what is happening during the transition period.
- We use the minimal amount of routers to recreate the behaviour of their network in a simulation and verify that the same convergence effects can be seen there
- We research and develop possible solutions and then test them in the simulation to assess their effectiveness.
- If possible, we implement the best solution in SWITCHes network and perform some measurements to verify its effectiveness in the real world

1.3 Overview

This thesis is structured in the following way: Chapter 2 will provide a short overview of for this work relevant parts of BGP and Open Shortest Path First (OSPF), as these are the two main protocols that are relevant for the convergence. We then provide an introduction to our simulation software of choice, GNS3 and present some related work. Chapter 3 is dedicated to the problem analysis, where we first go into more detail about the incident which triggered this investigation before summarising the basic steps that have to be fulfilled in order to converge. We move on to discuss the setup of SWITCHes network and analyse the log data that was recorded during

the link failure. Chapter 4 introduces our solution design and explains how it can improve the convergence problem. In Chapter 5 we go over the measurement setups that have been developed to test the effectiveness of our changes in the simulation as well as in the real network. We carry on by discussing the data that was collected and show interesting insights into the behaviour of a real network during the convergence. In chapter 6 we present additional improvements that can be easily accomplished in the current network, as well as changes that could be implemented by manufacturers to further improve the speed at which traffic can be rerouted after a failure. Finally, we conclude the thesis with a short summary.

Chapter 2

Background and Related Work

In section 2.1 we introduce the mechanisms of BGP as well as OSPF that are relevant for the understanding of this thesis and its results. We will also introduce GNS3, a network simulation and emulation tool. Section 2.2 presents some related work that also aims to improve convergence time.

2.1 Background

The internet is a network of networks. The glue that is bonding those networks together is the Border Gateway Protocol (BGP). Subsection 2.1.1 takes a closer look at BGP and explains the relevant mechanisms. An Interior Gateway Protocol (IGP) is used inside of each of those individual networks to decide where traffic will be routed to in that particular topology [32]. The IGP used by SWITCH is OSPF and subsection 2.1.2 provides an overview of how it works.

2.1.1 Border Gateway Protocol (BGP)

The primary function of BGP is to exchange network reachability information. Or in other words, BGP is the protocol used by ASes to tell each other which prefixes can be reached through them. This is done by sending *UPDATE-messages* containing the prefix(es) that can be reached. They also contain an AS path list, that informs a neighbour which ASes the reachability information has traversed, allowing for loop detection, enforcement of route policies and is used for best path calculations. Those prefixes are viewed as reachable through that advertiser until they are withdrawn or the BGP session goes down. *WITHDRAW-messages* contain a list of the prefix(es) that have to be invalidated for this peer. BGP is also used inside of an AS to tell each router which prefixes a border router has learnt from its neighbours. Those two flavours of BGP are called external BGP (eBGP) and internal BGP (iBGP) and will be explained in the following [37].

We will then go on to explain route-reflectors (RR) that were introduced to improve the scalability problem of iBGP sessions. The last subsection covers the route selection process of BGP.

eBGP

A BGP session that is used to communicate over the boundary of an individual AS to another one is called an external BGP session. Those border routers which maintain such an eBGP session apply their BGP policies onto the routes that they learn to decide which ones they will accept and which ones they choose in case of multiple possible routes to a single prefix.

iBGP

BGP sessions between routers of the same AS are internal BGP sessions. They are used to propagate the routes that are learnt by border routers. The next-hop of those routes will not be altered by default. The problem with that is that internal routers do not know how to reach that next-hop address, as it is not part of the IGP in most configurations. The *next-hop-self* command can solve this by allowing the border router to insert itself as the next-hop before propagating the routes via iBGP. iBGP sessions have to be set up manually for each connection between any two routers. This full mesh approach does not scale well as it leads to n^2 connections. Each time a router is added or removed, it is required to adjust the configuration of all routers. This is not desirable for networks that consist of more than a few routers and the reason why route reflectors were introduced.

Route Reflector

Route reflectors were introduced to BGP, to overcome the scalability problem of its iBGP sessions, which require a full mesh configuration of all routers in a given AS [18]. A route reflector (RR) alleviates this problem by re-advertising routes that are learnt via iBGP, which is not done in standard BGP [37].

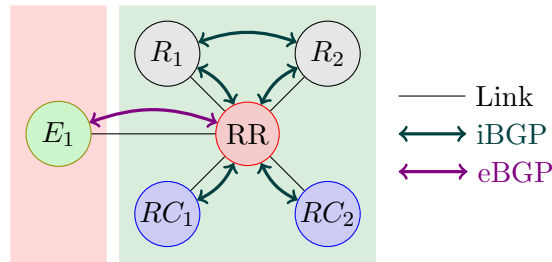


Figure 2.1: Basic topology with one route reflector (red), two client routers (blue) and two non-client routers (grey). The green router is an external border router in a different AS. BGP sessions are indicated by double-sided arrows

Figure 2.1 shows a simple topology with two ASes. One is the red one with router E_1 and the other AS is the green one with a total of five routers in it. In the centre of the green topology is a route reflector, R_1 and R_2 on the top are normal, non-client peers of the route reflector [18]. They, therefore, need to have full mesh iBGP sessions in this part of the network, indicated by the green double arrows running between all of them. The route reflector clients, or client peers, of the route reflector, don't need to run another iBGP session than the one to the route reflector, as they will receive all updates through it. The distinction if a router is a client or a non-client peer of the route reflector is done on the RR itself and only influences how learnt routes are redistributed. This allows running BGP peers that are not compatible with route reflection in the same network.

There are three sources from which a route reflector can learn a route. The originator of the route is marked with a double circle in each case. Standard BGP route distribution is shown by blue arrows and route reflection is indicated by the red arrows. The route reflector operates both as a normal iBGP speaker, where routes that are learnt through iBGP will not be propagated to other iBGP peers, as well as a reflector.

Figure 2.2a shows the case where the route reflector learns a route over iBGP from a non-client peer, here R_1 . R_1 will also send the route to R_2 as they are running a standard iBGP session. The route reflector will send the learnt route to its client peers (RC_1 and RC_2) but not to its non-client

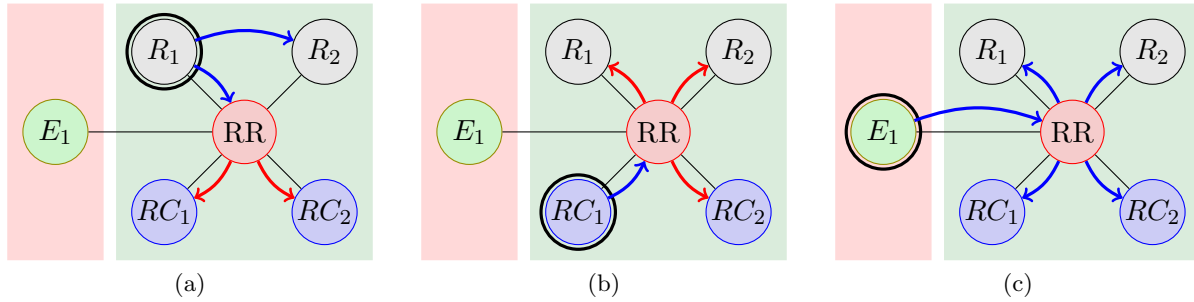


Figure 2.2: The three rules how routes will be propagated by the route reflector. The originator of the route is marked by a double circle. Blue arrows indicate standard BGP route distribution and red arrows the advertisement of reflected routes. (a) shows the distribution of routes learnt from a non-client peer. (b) shows a case of a route learnt from a client peer and (c) shows the propagation of a route learnt via eBGP

peers (R_1 and R_2). The route could also be shared via eBGP with E_1 , but this depends on the export rules and is not part of the route reflection and will not be further discussed here.

The second source that the RR could learn a route from, is from one of its client peers, shown in Figure 2.2b. A route learnt from a client peer, here RC_1 , will be reflected to both client peers and non-client peers. The third and last option is a route learnt via eBGP, in this case from E_1 . If the route reflector elects the learnt route as the best route to that prefix, it will distribute it to all its iBGP peers, client or non-client alike. This is a standard BGP operation as no reflection is happening in this case.

In conclusion, the three rules are

1. routes learnt from non-client peers will be propagated to client peers (Figure 2.2a).
2. routes learnt from client peers are propagated to both client and non-client peers (Figure 2.2b).
3. routes learnt via eBGP are sent to both client and non-client peers, which is standard iBGP (Figure 2.2c).

The disadvantage of a route reflector is, that it represents a single point of failure. If the route reflector goes down, all its clients lose their routing information. It is possible to configure multiple route reflectors in the same cluster to provide some redundancy. Meaning that routers can be clients of multiple RRs. A 4-byte *CLUSTER_ID* can be used to differentiate between multiple route reflectors in the same cluster. This way RRs can discard messages from other route reflectors, avoiding the looping of reflected messages [18].

BGP Route Selection Process

We have now covered how routes can be learnt with BGP. The next step is to decide which routes should be used to route traffic and to advertise to peers. This process can be split into three phases: The first phase calculates the degree of preference for each route, the second phase chooses the best of the available routes and installs it, and the third phase decides which of those routes should be advertised to peers [37].

Phase 1: All incoming routes are checked against the input filters and can be dropped, according to the specifications of the network operator. The routes that are accepted are then rated by the following, descending ordered conditions, meaning that the first condition that makes one route more preferable than the other is used as a decider [43]:

1. **Higher LOCAL-PREF:** *LOCAL-PREF* is an attribute that can be added to routes with an input filter and is freely configurable by the network administrator, routes where the *LOCAL-PREF* is not explicitly set, use the default value, usually 100.
2. **Shorter AS-PATH length:** The *AS-PATH* length is used as an indicator of the distance to the destination and shorter paths are preferred.
3. **Lower MED:** The *MED* or Multi Exit Discriminator is an attribute that can be set by a neighbour to influence which connection should be used if multiple ones are available.
4. **Learned via eBGP over iBGP:**
 - (a) **Lower IGP metric to the next-hop**
 - (b) **Smaller egress IP address:** This is used as a last measure to tie-break if all preceding conditions where equal

Phase 2: The best route is now selected, based on the preferences calculated in phase 1. Routes that would lead to a loop according to the *AS-PATH* or that have unreachable next-hops are discarded in this process [37].

Phase 3: An update message has to be issued if a better route has been selected for a prefix that has already had a route before, to inform peers about the improved route. The route has to be withdrawn if the new route has been learnt via iBGP as those routes must not be redistributed [37]. It will therefore only be known by the border router that has learnt the less preferable route, that an alternate path to that prefix exists. This is important to keep in mind for the discussion of the possible solution methods.

2.1.2 Open Shortest Path First (OSPF)

Open Shortest Path First (OSPF) is classified as an Interior Gateway Protocol (IGP). As such, it is only operating within the bounds of a single AS. OSPF is a *link-state* protocol. Each router maintains a database describing the topology of the AS. The database is identical for all routers belonging to the same OPSF area, aside from inconsistencies during convergence. Each router shares its local state by flooding. This link-state database represents a weighted, directed graph with all routers and edges of the network. Every router runs the same algorithm on this graph to construct the shortest path tree with itself as the root and will then populate its routing table accordingly. A relevant part of OSPF for this thesis is the ability to announce external routing information [33].

External Routing Information

OSPF offers the ability to distribute external routing information, or in other words, it is possible for routers to announce prefixes that can be reached through them. Those prefixes can either be learnt via another protocol or be statically configured. This routing information is then flooded

through the OSPF area unaltered and stored separately from the link-state database. The route selection will be done based on the shortest path tree and will be added to the routing table accordingly [33].

One example of this is the default route that could be distributed this way. It is likely that multiple providers are connected to a network and therefore multiple routers are announcing the default route. Each node in the OSPF area will now add the route to its routing table according to the shortest path tree. Therefore each one is choosing the closest location as the destination for traffic to the default route.

2.1.3 GNS3

GNS3 is software that allows for the testing of networks and their configuration in a virtual environment. It consists of two main components, the GUI and the VM. The GNS3-all-in-one GUI is where one will create the topology and offers the ability to control everything from the server settings, to images and the actual devices in the network. These devices need a server to run on. There are three possible options for the server. It is possible to run a local server that will allow the running of pure software devices. If one wants to run devices that need to be emulated one can either run them in the local GNS3 VM or in a remote GNS3 VM. The VM unlocks the full potential of GNS3 by offering the ability to emulate the hardware of for example a Cisco router so that it can run its native operation system within GNS3 [13].

For a more advanced use, GNS3 can be divided into four parts [8]:

1. **The user interface GUI:** The GUI is the graphical interface where one can create, view and control a network.
2. **The controller:** The controller is at the heart of GNS3. It manages the state of a project and communicates with the computes and the GUI. It can be controlled by the GNS3 API with standard HTTP get and set messages from outside of the GUI environment [8]. This will be used in this thesis to automate some of the configuration and information acquisition.
3. **The computes:** The computes are the servers on which the devices are running. It is possible to run some devices of a network on one server and other devices on a different server or compute.
4. **The emulators:** The emulators are the pieces of software that run on the computes underneath the devices that require emulation of their native hardware.

2.2 Related Work

Slow convergence of BGP in case of a link or node failure is nothing new and has been the topic of many papers and articles over the years. In the following, we will have a look at some of the concepts and alleviation methods that have been proposed.

2.2.1 Changes to the BGP Protocol

The IEEE [20] has outlined that lies or in other words the incorrect information that is floating around in networks after a node failure can contribute a lot of time to the convergence process. Their major method to avoid this was by introducing the *ghost flushing rule*. This rule works by sending a withdrawal message for a destination for which the AS path has gotten worse within the

minRouteAdver time. This avoids the usual count to infinity problem that can usually occur and can therefore speed up the removal of routes that go through failed nodes significantly.

Another approach is to allow BGP update messages to carry the cause that triggered the update to allow recipients of this message to invalidate all routes that are using the same next-hop if the updates were triggered by a link failure [34].

The problem with these solutions is that they require changes to the operating system of the currently deployed hardware. This can only be done by a manufacturer and not by the network administrator, which is out of reach for this thesis.

2.2.2 Configuration Optimizations

Several studies have analysed the Minimal Route Advertisement Interval (MRAI) that is designed to dampen the number of BGP updates that are sent for any prefix [15, 22, 24]. They presented optimal values that still dampen similarly to the original 30s value, but allow for faster convergence speed [15]. They also showed that using different MRAI values across a network can exponentially increase the time it takes to converge after a failure [22].

2.2.3 External Changes

Software-Defined Networking (SDN) is a technology that allows to configure and manage a network in a central location in contrast to traditional networks that require the individual configuration of each node. This aims to decrease manual errors and increase flexibility and performance by using a single global state. There are studies [41] that have shown, that the convergence of BGP can be sped up by using inter-domain SDN.

SWIFT [31] is a technique that can be applied to any existing router by interposing a SWIFT controller and a SDN switch between the router and its BGP peers. It is then able to quickly reroute the traffic after only seeing a fraction of the withdrawal messages in case of a link failure, by accurately predicting which link failed and falling back to precomputed alternate paths.

While such solutions provide good performance improvements over current setups, they also cost money, time and a lot of willingness to learn and use experimental hardware and are therefore not viable.

Chapter 3

Problem Analysis

On the 12th of January 2021, SWITCH was having an all hands-on online-meeting in Zoom with most of their staff. The Zoom servers that are hosted by Oracle are connected to their network through DE-CIX in Frankfurt. All staff members were advised to use the company virtual private network (VPN) and therefore most of them were connected to the Zoom server through this 100Gbit/s optical link as it failed and got subsequently disconnected from the call. The failure triggered the withdrawal of 250'000 prefixes that were learnt at the internet exchange point (IXP) by BA3, the router that was connected to it. It took two minutes and nine seconds for the network to converge on a new route to the servers of Zoom and for SWITCH to be able to proceed with their meeting.

This experience triggered an internal investigation to figure out why this happened. They realised that this process would always lead to such a lengthy disruption of service for their customers. It was short enough for customers not to complain as most users would suspect a problem with their computer or the Zoom server and has therefore gone unnoticed in absence of automated monitoring systems that are able to identify this problem. But it would happen from time to time as links can fail or be taken offline for maintenance.

It is important to get a better understanding of the processes that lead to this convergence behaviour to be able to provide appropriate countermeasures to solve this for future failures. Therefore this chapter will focus on the analysis of the log files and configuration data provided by the network operators.

Section 3.1 goes over the steps that have to be completed in order for BGP to converge after a link failure. The next section 3.2 takes a closer look at the topology and its configuration by SWITCH. The last section 3.3 dives into the analysis of the log data that was captured during the incident and subsequent failures of the same link.

3.1 Basic Steps of Link Failure Convergence

The connection loss between two BGP peers can be caused by the failure of the link that is connecting them or the failure of one of the routers. Routes that utilised this path can obviously not be routed on this link anymore. It is therefore necessary to recompute them and also inform the upstream routers that the old path has been invalidated. The network will only have re-converged once all routers got the updates and withdrawals for all affected prefixes and processed them.

The process can be formally split into four steps that have to be performed for the network to converge after a link failure: Failure detection, failure propagation, routing recalculation and updating of the routing and forwarding tables (RIB & FIB) [27].

3.1.1 Failure Detection

Failure detection on its own is a very complex topic. The detection methods and their resource requirements vary depending on the layer that they should monitor and the targeted detection speed. We will focus on the failure detection for BGP sessions, as this is the failure type that is relevant to this thesis.

BGP on its own has the *HoldTime* [37] that specifies after which time period within which no messages (*KEEPALIVE*, *UPDATE* and/or *NOTIFICATION*) from the peer have been received, a connection is closed. This time is traditionally relative long. The standard value, suggested by BGP in its RFC is 90 seconds. Lowering this value increases the CPU load as more messages have to be generated and can also cause false positives when the CPU can not keep up with the processing of those updates.

Bidirectional Forwarding Detection (BFD) [26] is a simple protocol that is designed to significantly increase the speed of fault detections of the bidirectional path between two forwarding engines. It is intended to be implemented in the forwarding engine and can therefore achieve much smaller intervals in-between hello messages than protocols that rely on the control plane to generate those messages. It also alleviates the need for every protocol to run its own proprietary hello mechanism. BFD notifies the overlying protocol once a failure has been detected.

The use of optical communication allows to detect failures at the interfaces level by monitoring the presence of light on the fibre. This means that no protocols or timers are needed to discover if the interface on the other end is operational and therefore allows to detect such a failure immediately.

| Detection Method | Advantages | Disadvantages |
|---------------------------------|--|--|
| BGP HoldTimer | monitoring covers all layers | slow, resource-intensive |
| BFD | fast failure detection (ms), only one hello process needed for all overlying protocols, minimal resource consumption | depending on the implementation only covers the forwarding plane and the link layer, but not the control plane |
| Monitoring of presence of light | instantaneous, no protocol or messages required, no processing overhead | only covers link and interface failures, higher-level failures can only be detected if they also lead to a failure of the interface. |

Table 3.1: Comparison of different failure detection methods

3.1.2 Failure Propagation

Once the failure has been detected, it needs to be forwarded to the other routers in the network. In the case of BGP, this means that the next two steps have to be also done before it can propagate the failure. It first needs to do the routing recalculation and update the RIB before it can send the update or withdrawal message on a per-prefix basis to all its BGP peers [37]. The reason for this is, that BGP being a distance-vector protocol does not report the failure itself, but the subsequent changes to the routes. This does not scale well because it has to do the processing before being able to send out any information about the failure and the information about the interruption is only relayed indirectly on a per-prefix basis which multiplies the number of messages sent dramatically.

3.1.3 Routing Recalculation

When a router gets the information that a prefix is withdrawn or updated by one of its peers, it needs to recalculate the best route for that prefix. If it gets an update, it needs to compare it to already known routes and decide if it is better than the freshly learnt one. In case of a withdrawal, it needs to either fall back to the next best route or mark the prefix to be removed from the RIB and FIB.

3.1.4 Updating of RIB and FIB

The newly calculated route has to be stored in the RIB and the FIB. While updating the RIB is quite straightforward as it is stored in RAM, updating the FIB is not as it is stored in TCAM.

TCAM or ternary content-addressable memory is a special kind of memory that allows to perform a longest prefix match in just one cycle. It is necessary to sort prefixes in the TCAM from longest prefixes to shortest prefixes as the TCAM will return the first match. This means that generally inserting or deleting an entry can result in $O(n)$ operations as the following entries have to be moved up or down to allow for the new entry or to fill the empty hole [16]. Therefore updates of the FIB can take up a lot of resources and contribute considerably to the convergence time.

3.2 SWITCHes Setup

To get a better understanding of what happened during the incident of 12.01.21, where the connection to the DE-CIX went down, it is necessary to analyse the setup of SWITCHes network.

3.2.1 General Overview

Figure 3.1 shows the layer 3 topology of SWITCHes network. Black routers carry the full routing table that SWITCH is using. Note that they only carry around 350'000 prefixes and do therefore not have full routing as there are around a million prefixes at the time of writing [5, 25]. Blue routers have even further reduced routing tables and only forward traffic to the closest backbone router (black routers). This means that they are not relevant for the BGP convergence that is investigated in this thesis as they do not learn any of those external prefixes in the first place.

The thickness of the connection lines visualises the bandwidth of said links. The capacity of those connections currently varies from one to 200 gigabits per second.

While SWITCH is the provider for Swiss universities and research facilities, it also needs providers for itself as it can not interconnect with every network on its own. Those higher level providers or transit providers are Cogent, Telia and Level3. They are connected in different locations by the routers BA3, EZ3 and CE3 respectively. While transit-providers enable connectivity to the whole internet, they are also expensive.

This is why network operators prefer to peer directly with networks with which they exchange a lot of traffic. Internet exchange points or IXPs offer a central location to peer with a lot of other networks as it would be tedious and very expensive to build and maintain direct connections to all desired networks. Peers do not pay each other money for the traffic that they exchange and routes via a peer are therefore preferred over routes from providers. SWITCH is present in the following exchange points: AMS-IX, CIXP, DE-CIX, Equinix and SWISSIX. The IXP most notably here is DE-CIX as this is the exchange point to which the fibre link went down during the here investigated incident.

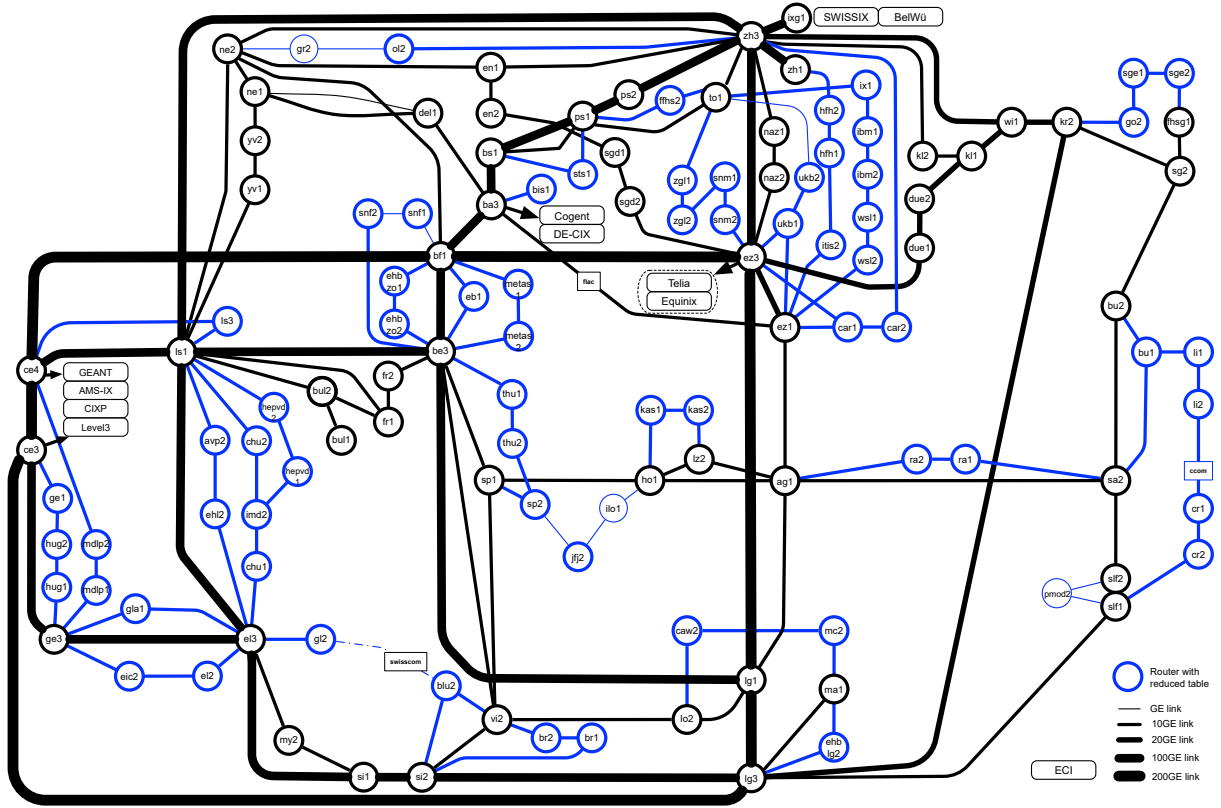


Figure 3.1: Layer 3 view of SWITCHes topology, with routers that carry all their prefixes in black and ones with a reduced routing table in blue

The last external connection is to GEANT [7], a global network that interconnects research and education facilities. In total there are four exit points in SWITCHes network, BA3, EZ3, CE3 and ZH3. Technically CE4 would also be an exit point, but it is only a backup for CE3 and will not carry any traffic as long as CE3 is operational.

3.2.2 Router Configuration

SWITCH is using a route-reflector(RR) setup in their topology as introduced in subsection 2.1.1, to avoid the need for a full mesh iBGP configuration in their network. In total, they are using the three routers LS1, BF1, and LG1 as their RRs. This introduces some redundancy into the configuration as a single route-reflector also poses a single point of failure. Every router in the network is peering with each RR. Please note that the third route-reflector was EZ1 during the incident of 12.01.21 and has been exchanged by BF1 by now as this is a faster router. This is very relevant as the slowest route-reflector lower-bounds the convergence time. Each route-reflector sends a reflected message to every router for each one that it receives. We, therefore, have three entries for each prefix in the table of each client router, which all need to be withdrawn before the route is deleted from the FIB. It is therefore possible that the convergence speed has already been increased by switching out the slowest RR for a faster one.

The default route is distributed with OSPF and the only routers that are configured to be able

to advertise such a route are BA3, CE3, EZ3. This makes sense as those are the three routers that are connected to providers as seen in subsection 3.2.1. They will only do this if they have an entry for 0.0.0.0/0 in their routing table. This ensures that only the routers that have a connection to their respective peer inside of the transit provider are attracting traffic to the default route. All of them set the metric to 10, so the selection of which exit point to use will solely depend on the OSPF weights. There is only a single OSPF area used for the entire network.

Taking a closer look at the configuration of BA3 reveals that the default route if learnt from Cogent, is dropped immediately. Therefore SWITCH is only really using two out of its three providers, which also means that BA3 can not fall back to sending the traffic to its provider once its connection to the peers is lost. Cogent is left unused even though SWITCH is paying for the service, because of previous troubles that subsequently lead to an internal policy to not use it anymore. The only routes that are accepted from Cogent are mainly the ones that originated from within said providers network.

BA3 is also the router that is connected to the DE-CIX and directly peers with a total of 14 other autonomous systems at that exchange point. The routes for the remaining entities that are present at that IXP are learnt through the route servers of the DE-CIX. A route server allows to learn all prefixes that are reachable through that exchange point without the need to peer directly with each participant [9]. BA3 is using the next-hop-self command on all routes learnt through those eBGP sessions.

3.3 Log Data Analysis

Subsection 3.3.1 goes over the preliminary analysis that was done by SWITCH after the incident. Subsection 3.3.2 discusses the information found in the BMP log from the same failure that occurred again on 22.06.21.

3.3.1 Preliminary Analysis by SWITCH

Simon Leinen from SWITCH analysed their log after the incident that he described as follows:

On 12 January 2021 at 11:04:36 MET, we experienced a sudden loss of the 100Gb/s link to DE-CIX in Frankfurt. This caused a two-minute disruption to many ongoing Zoom conferences. The primary bottleneck causing this long duration is the withdrawal of 250'000 DE-CIX routes from our route reflector-based iBGP mesh until the default route took over the traffic from SWITCH to the Zoom servers hosted on Oracle Cloud Infrastructure. [30]

The interface was immediately marked as down and only 6ms later all BGP sessions were marked as down as well. This means that the failure detection and the notification to BGP were done very fast and has no need to be further improved. After 8.2 seconds all IPv6 updates to the FIB were done and 14.9 seconds after the failure all IPv4 ones too. [30].

The peers from inside DE-CIX did not react as quickly and only started to update their routes 2 minutes and 2 seconds after the failure [30]. This has most likely to do with the fact that the links for them on the other side of the switch inside the exchange point did not go down and that they, therefore, relied on the BGP HoldTime to detect the failure.

Looking at Zoom traffic specifically with NetFlow that they collected during the link failure, he could show that traffic only started to flow again after 2 minutes and 9 seconds towards Level 3 and after 2 minutes and 13 seconds towards Telia [30]. This means that the failure propagation,

the subsequent route recalculation and the updating of the FIBs of all routers on the path took significantly longer than for BA3 to converge.

3.3.2 BMP Log Analysis

The BGP Monitoring Protocol (BMP) is a protocol that is used to provide route views from the router that is monitored [40]. It has been used here to capture a BMP log from one of the lab routers of SWITCH. A lab router is a router that is used for testing purposes and as a spare in case of a hardware failure. This means that it is not part of the actual network, but it was set up to peer with the route-reflectors as a client and was therefore learning the same routes as every other client router and seeing all the same updates that were issued in the network. The log that will be analysed in the following was recorded on the 22nd of June 2021, when the same link as in the original incident to DE-CIX was taken down for planned maintenance. The network has not been prepared in any kind for this interruption and had converged as if it was a link failure.

While there actually were two events where the link went down and two when it came back up again, we will only focus on the first link failure as we are interested in optimising the convergence for this case. The notable difference between the incident that SWITCH observed and the failure that is analysed here is that the route-reflector EZ1 was exchanged by BF1 by now, which potentially removed one of the bottlenecks.

Once again, like in the analysis that was done by SWITCH, we saw that all the BGP sessions were marked as down only 6ms after the link failure. The first withdrawal messages reached the lab router already 2 seconds after the link went down and the last one after 22 seconds. This means that by that time BA3 has sent withdraws for all prefixes and that all three route-reflectors had processed them and relayed them. Comparing the RRs also showed that they performed nearly identical. If EZ1 would have been slower and influenced the convergence can not be checked anymore. This is unfortunate from an academic standpoint but is the reality when working with a production network as it has to mainly work as a product.

Also, only around 187'000 prefixes were withdrawn or updated during this incident instead of 250'000 as described by SWITCH. This could be because of changes in the peerings, or just changes in the routes that the peers announce towards SWITCH. This is outside of control and can vary from test to test.

Taking a closer look at the messages that are sent or better said relayed by a single route-reflector, reveals that most prefixes not only see one update or one withdraw but on average 3-4 messages. This multiplication further increases the processing and the amounts of updates that have to be done to the FIB before the network converges. A lot of these unnecessary messages happen as BA3 is deleting its routes and updating the prefix with the second, third or so on route that it has learnt. In the end, it will need to withdraw all routes as all external neighbours are down. This is an artefact of the incremental removal process of every individual route and leads to a lot of overhead as each router in the network has to process all these updates.

A minority of these additional updates are not unnecessary though as these are update messages that are issued by other border routers in the network that announce their route to prefixes, once the best route via BA3 has been withdrawn.

3.4 Conclusion on the Problem Analysis

We have seen that BA3, which is the router that is connected to DE-CIX, notices the link failure immediately as it is an optical interface. It then marks all BGP sessions as down within 6ms, starts to process this failure and finishes to update its FIB within 14.9 seconds. During this period all

traffic that is sent to it is black-holed and will start to loop once the respective prefix is removed from the FIB, while other routers are still using the old path. Such transient loops will form everywhere in the network where one router has converged for a certain prefix and a neighbouring one has not yet done so if the new and the old routes lead in opposing directions. The BMP log shows that the first withdraw reached our lab router that acts as a route-reflector-client within two seconds and the last withdraws are issued only 22 seconds after the failure. We can infer from that, that all three RRs have processed the BGP messages from BA3 and updated their RIB within that period. They all perform very similarly and none is significantly slower than the others. What we don't know is if the original third RR did perform worse. These 22 seconds for the withdrawal of all prefixes are a lot shorter than the 2 minutes and 9 seconds that were recorded by SWITCH during the initial incident until they were seeing the first traffic to Zoom. One can not say with certainty if this large difference is attributable to the different route reflector or to other routers in the network that take longer to converge. As we have no information when traffic was able to leave the network during this second measurement.

In any case, we can say that the main problem is the failure propagation method. While BA3 and even the route-reflectors seem to be able to process this information far quicker than the whole of the network, the sheer bulk of messages that have to be processed by every router in the backbone is unnecessary and leads to a lot of overhead. These messages that indirectly communicate the failure are also only able to be sent by BA3 once it has updated its tables and removed the prefixes, instead of directly notifying the rest of the network about the failure. A third aspect is that all these messages also compete with the looping traffic that is probably saturating some links during the convergence process. This is exaggerated by the number of individual messages that have to be sent.

To summarise this in a single sentence: The main disadvantage of the current setup is, that each prefix has to be withdrawn explicitly while this method carries no information about the link failure that triggered this removal. Therefore the following goals would be desirable to be achieved:

1. Notify the other routers in the network of the link failure before processing any updates to the RIB
2. Minimise the amount of data that has to be transmitted
3. Avoid routing loop where possible

Chapter 4

Solution Design

This chapter will cover the changes to the configuration of SWITCHes network that we propose to decrease the convergence time in case of failure. Section 4.1 covers the main changes that have to be done to implement our solution. Section 4.2 explains how the BGP next-hop address tracking feature helps to avoid unnecessary idle times at the beginning of the converge. Finally, section 4.3 goes over the prefix independent convergence feature of Cisco routers that provide the possibility for fast failover of routes with a backup path.

4.1 OSPF Link Failure Distribution

The basic idea of our solution, is to directly inform every router in the network about the link failure so that each of them can immediately invalidate the next-hops that went offline and start to remove the routes from their FIB. To achieve this we propose the following changes to SWITCHes configuration:

1. Setup interfaces connected to neighbours where a lot of prefixes are learnt as OSPF passive links. These mainly include the interfaces to internet exchange points.
2. Don't use next-hop-self for routes that are learnt via eBGP on these interfaces.

This way the links to peers are added to the IGP and a simple link-state update can notify all routers in the topology about the loss of connection to all neighbours that have been reached through that interface. All routes can therefore be locally and quickly invalidated as the border routers of the neighbours are the next-hop to reach those prefixes instead of SWITCHes own border router.

This shifts the load of the convergence from the failure distribution mechanism to the hardware capabilities of each router. Another benefit is, that we move from routing loops to blackholing traffic that can not be redirected at this point. This is done by invalidating all unreachable next-hops, which can be done much faster than the removal of all prefixes from the FIB. This is beneficial as black holes generate less load on the network and improve the likelihood of new routing information to make it through.

4.2 BGP Next-Hop Address Tracking

Another mechanism that helps to improve convergence is BGP next-hop address tracking. The default BGP scanner polls the RIB every 60 seconds to check whether there have been any changes

to the next-hop reachability since the last polling [3]. This means that in the worst case a change to the next-hop is done right after a polling event and that BGP is not doing anything to find the next best route and to notify its peers about the necessary changes in the routing for nearly a minute.

BGP next-hop tracking combats this by using an event-driven notification system that automatically tracks the next-hops when peering sessions are established. It then reports changes to those next-hops as soon as they are being made to the RIB or after a predefined amount of time to filter noise [3].

This feature is on by default, but could only be used by the border router that detected the failure as the rest of the network did not see any changes in the next-hop with the current configuration. Now BGP next-hop tracking allows all routers in the network to start withdrawing unreachable routes from the FIB and therefore improves the convergence time.

4.3 Prefix Independent Convergence (PIC)

Prefix independent convergence [2] is a feature of Cisco routers that allows for fast failover to an alternate route if present by the use of hierarchical FIBs. This feature only improves the data plane convergence and not the control plane convergence.

It works by pre-computing alternate paths for each prefix if possible and stores them to the RIB, FIB and Cisco Express Forwarding. If a failure happens to one of the prefixes that have a backup path, it can quickly reroute the traffic by invalidating the next-hop that went down. This has to be only done once per next-hop and not for each prefix that is using that next-hop. After that, the prefixes still points to the same primary route but will use the backup route when it notices that the original destination is no longer valid [2].

The limitation of PIC is, that there has to be an alternate for each prefix that should be protected by this feature. Another less problematic limitation is that it can only protect against a single failure at a time as the control plane has to update the forwarding table to use the backup route as the new primary route and find a new backup path to install before being ready for another failure [2].

The problem with SWITCHes current setup is, that they hide alternate paths. So only the exit router that has the cheapest connection will be retained and visible to the rest of the network. The only exceptions are routes that have the same cost at multiple exit points and therefore are selected depending on the internal weights. This means that PIC can currently not do much to help improve the convergence.

Chapter 5

Evaluation

The discussion of the measurements in GNS3 is covered in section 5.1. We will first take a look at how the measurement impacts the convergence. Next, we will compare the differences between the original setup and the improved configuration. We will also discuss the effect of a slow route-reflector-client on the network. Section 5.2 will compare the simulation results to the data that was collected in the real production network of SWITCH during a link failure. Finally, section 5.3 will go over additional interesting findings that have been made during this thesis that are not directly related to the main topic.

5.1 GNS3

All preliminary testing will be done in GNS3 since changing the configuration of routers in a production network has to be done well planned. Introducing failures on purpose can not be done lightly as it interrupts the service for customers. Subsection 5.1.1 goes over the topology and the configuration that has been chosen to test the convergence. The behaviour of this setup is discussed in subsection 5.1.2. All measurements have been limited to IPv4 prefixes to simplify the testing and because IPv6 prefixes only make up roughly 10% of all prefixes that are lost when the connection to DE-CIX goes down.

As introduced in 2.1.3, GNS3 is a simulation and emulation environment for internet networks. As especially emulating uses a lot more resources than running it on real hardware and the fact that the whole network will run on a single computer, it is necessary to scale the topology of SWITCHes production network down to a minimum.

5.1.1 Topology

SWITCHes network can be reduced to the route-reflectors, the route-reflector-clients (RRC) and border routers like BA3, which is connected to the DE-CIX and CE3 that is connected to a provider where it learns the default route. We, therefore, chose to build our topology out of the same four building blocks. Figure 5.1 shows the testing topology. On the left, we can see AS1, which represents SWITCHes network. In the middle of it are the three redundant route-reflectors, labelled as RR1-3. We chose to also use three of them because we wanted to check whether there are any unexpected interactions between them. On the right is BA3 which represents the router with the same name in the original topology. It is connected to AS2 which is substituting as DE-CIX on this link and as a provider on the other link to CE4, which represents a router with a provider connection. BA3 and CE4 are purposely not directly attached so that the traffic has to be rerouted via the network

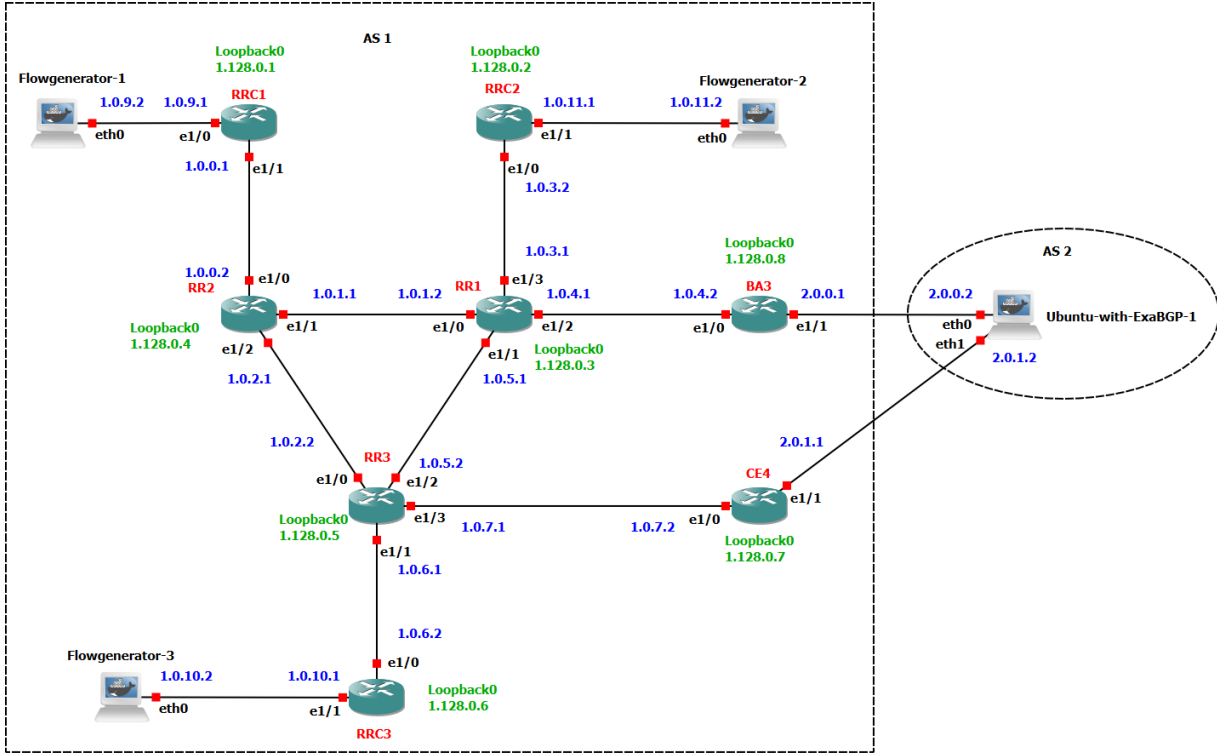


Figure 5.1: Topology in GNS3 to test a minimal version of SWITCHes setup. AS1 represents SWITCHes network with three RRs in the middle. AS2 is acting as DE-CIX to BA3 and as a transit-provider to CE4

once the link to the IXP is interrupted. BA3 also needs to do this in SWITCHes real setup as they are not using their provider connection at that location. The three remaining routers RRC1-3 represent additional route-reflector-clients. Each of them is attached to a host that will be used to inject flows into the network.

All routers are of the type C7200 with IOS 15.2, as images for that platform are well supported and should be sufficient to test the desired setups. The hosts are docker containers that run a bare minimum version of Ubuntu.

5.1.2 Configuration

The following configuration has been chosen this way to be as simple as possible while still representing all the convergence relevant settings of SWITCHes network. All routers are configured with a loopback interface in the 1.128.0.0/24 range and physical interfaces in the 1.0.0.0/16 range for internal interfaces of AS1. Interfaces towards and inside of AS2 are in the 2.0.0.0/16 range. The exact IP of each interface can be seen in figure 5.1. OSPF is turned on on all internal interfaces of AS1 and all routers are in a single area. CE4 is set up to have a static route for 0.0.0.0/0 towards Ubuntu-with-ExaBGP-1 to simulate a provider connection with a default route. It announces the default route via OSPF to the rest of the network to mimic the setup in SWITCHes network.

The BGP router-id is set to the loopback0 address for all iBGP sessions. All routers peer with each route-reflector. BA3 has an additional eBGP peering session with Ubuntu-with-ExaBGP-1

and is setting itself as the next-hop when announcing routes that have been learnt this way to the route-reflectors. The route-reflectors maintain standard iBGP sessions with each other and all other routers are set up as route-reflector-clients. The hosts are configured with the IP address on the interfaces that can be seen in figure 5.1 and use the interface of their connected routers (RRC1-3) as the default gateway to be able to send traffic into the network. The host in AS2 is set up with ExaBGP, it is *the BGP Swiss army knife of networking* [6] as they call it. It is able to provide simple translations of BGP messages for Software Defined Networking, it can help protect against DDOS attacks and it can also be used for load balancing applications [6]. But here it is only used to announce the nearly 200'000 prefixes that are learnt via DE-CIX as this is much simpler and easier to scale than having a router with dummy static routes announcing these routes.

5.1.3 Measurement Method

We deploy two different measurement methods. The first one is using flows to addresses equally distributed in the range of at the IXP learnt prefixes. They are injected by the hosts connected to the RRCs. The reason to use such flows is that one can deduct the state of the data-plane by observing where they are routed to and therefore see when a router has fully converged for a given address. The second measurement method is only tracking the BGP UPDATE messages that are sent on every session and does therefore not impact the network at all. While this produces far less detailed information than the first method, it also allows us to test whether the flow injection will influence the convergence.

Flow Injection and Collection

The flows are generated by injecting UDP traffic to dummy port 33'000 with a TTL of 8 to avoid the packets from multiplying excessively during convergence while routing loops are formed. The destination IPs of the packets are ten, equally spaced addresses, where the first address is within the lowest advertised prefix and the last address within the highest prefix. Those prefixes are announced by Ubuntu-with-ExaBGP-1 using ExaBGP and are the same 187'000 prefixes that are learnt by BA3 in SWITCHes real network.

Each packet carries four bytes of data, containing a sequence number so that the packet can later be uniquely identified. A flow is defined by the Flowgenerator that is sending it and the destination address. A packet is sent every 100ms for each flow. The data retrieval is done by packet captures running on each link. These captures are analysed by a Python3 script with Scapy [10].

BGP Session Tracking

For this method we track when each BGP session started to see withdraw messages for the disconnected prefixes and when they stopped on a per-session basis. For this, scapy had to be extended as it is not able to reassemble BGP frames on its own when they have been split up across multiple TCP packets. The BGP Session Tracking can be used independent of the flows and allows to compare the convergence speed with and without the flows present as they introduce additional load to the system.

Filtering

The data that has been collected and saved, still needs to be filtered. Capture data recorded in GNS3 is filtered by only keeping data that has:

- One of the destination links as the last entry in their path (BA3↔ExaBGP-1 or CE4↔ExaBGP-1)
- Or at least one link multiple times in its path, which means the flow is looping

The data lines will be interrupted if no new data point is present within 150ms, meaning that each packet loss would be visible as the packet inter-delay is 100ms. Data for the BGP session tracking does not need to be processed further as it already displays the first and last time that a session saw update packets.

5.1.4 Side-Effect of the Measurement Method on the Convergence

We first needed to check, whether our measurement method of injecting flows into the network has an impact on the convergence. We can do this by only tracking the BGP UPDATE messages that are sent in the network, as explained in 5.1.3 while inducing a link failure. This method does not put any load on the routers, as we only run packet traces on the links, which are operating separate from the routers and are directly handled by GNS3. With that, we get a baseline that we compare to the convergence time when the flows are injected into the network.

The full data set that has been recorded during those measurements can be found in Appendix B under B.1 and B.2. Choosing for example the session between RR1 and RR3 to compare the convergence of the network with and without flows shows that they don't impact it much. When no flows are running we can see the first UPDATE messages 1.37 seconds after the failure and all 187'000 prefixes have been withdrawn at 35.21 seconds. With flows injected it takes longer for the first withdraws to be present at 9.98 seconds after the failure but after 38.45 seconds all prefixes are withdrawn, only taking 9.2% longer than without. This is low enough so that we can be sure that our following tests in GNS3 are not altered much by the measurements.

5.1.5 Original Setup

Figure 5.2 shows the capture data that has been recorded for flows that have been injected into the topology, running the configuration representative of the original configuration of SWITCHes network. The time zero corresponds to the moment that the link between BA3 and Ubuntu-with-ExaBGP-1 fails. This represents the connection between BA3 and DE-CIX in the physical network. The y-axis shows the link on which the traffic is looping or through which border router the packets are routed to the destination. The first thing that is quite obvious is, that the data looks nearly identical for all three injection points. We can also see that all data disappears at the instance the link fails and only reappears again after 3.04 seconds. Taking a look at the raw input data reveals that the flows terminate at BA3 during that time and do not get sent back to RR1. We can deduct from that, that BA3 has not yet converged to use the default route advertised by CE4 and can also not keep on sending it to the original destination as this link has failed. This convergence in BA3 seems to happen within the same 100ms for all prefixes as all of them reappear in the same packet cycle. We can not infer what exactly happens here inside of BA3 and why it takes it three seconds to converge and then suddenly is able to do it at the same time for all prefixes.

The flows are looping between BA3 and RR1 after the 3.04 second mark, as RR1 still has the route via BA3 installed, but BA3 has converged and is trying to fall back to the default route via CE4. We can now see how the prefixes start to be withdrawn and 11.48 seconds after the failure, the first one is now rerouted via CE4. After 38.27 seconds all prefixes have been withdrawn by BA3 and are rerouted to CE4 to but it takes nearly another 14 seconds before CE4 has converged as well and all traffic can flow via the default route. We can see that the inter-delay between

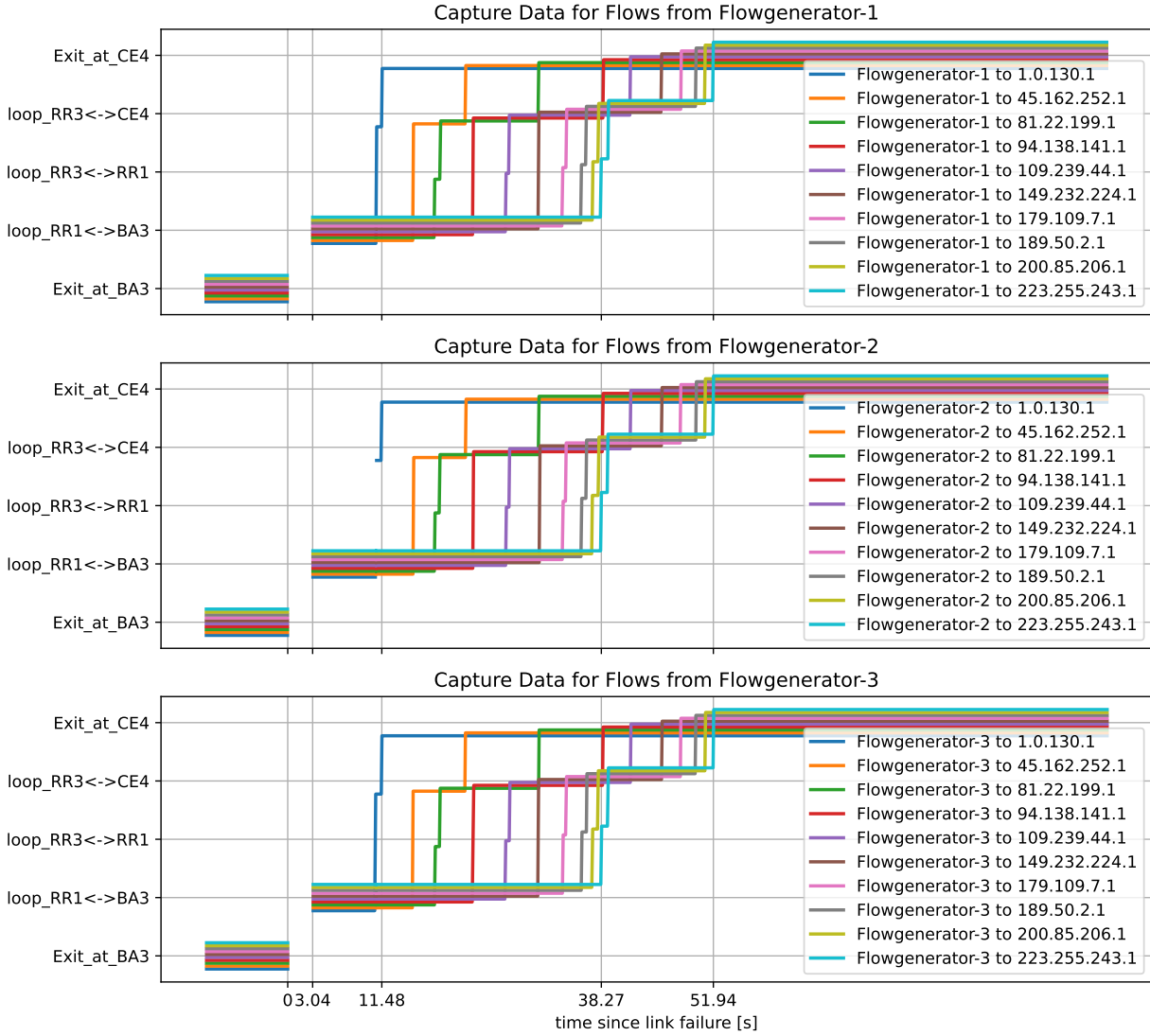


Figure 5.2: Capture data for a link failure in GNS3, showing on which link the flows are looping or where they are able to exit the topology, for the configuration representative of the original setup of SWITCHes network

prefixes converging on RR1 decreases with each prefix that has converges. This has most likely to do with the fact, that the link $RR1 \leftrightarrow BA3$ sees less and less load as fewer flows loop on it and this increases the chance that a BGP withdraw message makes its way through to RR1. This can not be definitively said as this is a simulation and both the data plane as well as the control plane is running on the same machine. RR1 and RR3 seem to converge nearly identical and only minimal times where traffic is looping between them can be observed. This makes sense as RR3 is not being pushed and will be able to process any update that it receives. Notable is that the prefixes are processed in ascending order. Therefore it seems to be desirable to have an IP that is as low as possible to experience as little downtime as possible. Additional plots that compare the convergence on a per-prefix basis for the different injection points can be found in Appendix C.

5.1.6 OSPF Distribution

Most settings stay the same for the OSPF distribution configuration setup. The only router that has to be updated is BA3 where the prefix will be learnt and where the link failure happens. We declare the interface towards Ubuntu-with-ExaBGP-1 as an OSPF passive interface and add it to the same OSPF area as the rest of AS1.

The other change is that we remove the BGP next-hop-self option from BA3 for the peering sessions with the route-reflectors to retain the IP of the Ubuntu-with-ExaBGP-1 host as the next-hop for all prefixes.

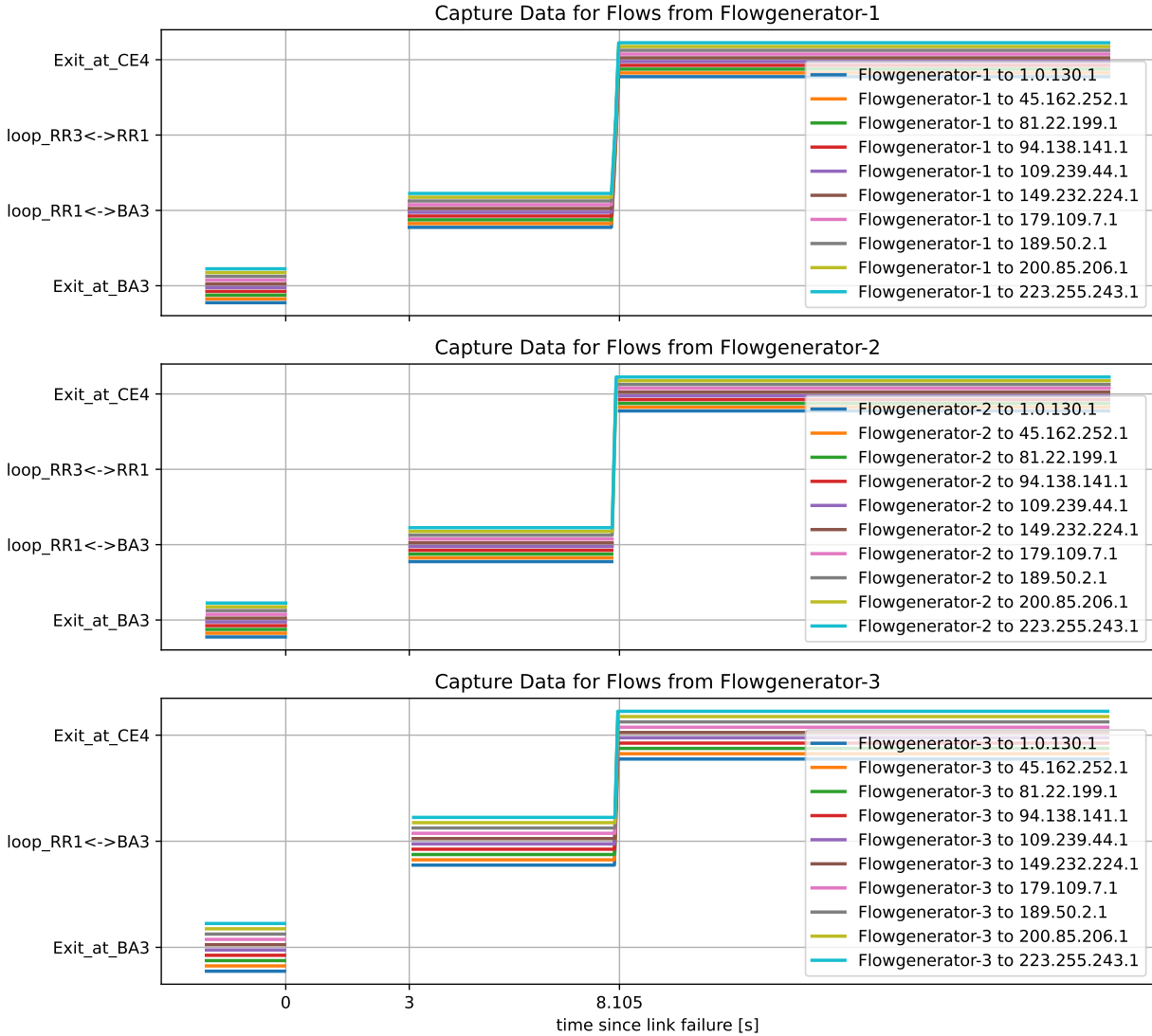


Figure 5.3: Capture data for a link failure in GNS3, showing on which link the flows are looping or where they are able to exit the topology, for the improved configuration where the link failure is distributed with OSPF

Taking a look at the capture data of the convergence behaviour for the improved configuration in figure 5.3, immediately makes it obvious that the process takes a lot less time. The flows do

disappear when the link fails and start to loop after 3 seconds, nearly identical as with the original configuration. But after 8.105 seconds all prefixes have converged over to CE4 and are using the default route. This is faster than what it took in the original setup to withdraw even the first prefix.

The BGP withdraws are still sent to all routers in the network, but they do not impact the routing anymore, as the routes have already been changed to the default route as the next-hop of the BGP paths have been invalidated. This result is promising but has to be taken with a grain of salt as a simulation can behave vastly different than a real network. Additional plots that compare convergence on a per prefix basis for the different injection points can be found in Appendix C under C.1.2

5.1.7 Effect of a slow Route-Reflector-Client on the Convergence Time

The route-reflectors in SWITCHes network have all of their over 120 routers connected as route-reflector-clients. This means that they have to send the messages over 120 times and maintain the state for each connection. Naturally, some routers will be slower to process this information than others and take longer to receive all messages. In the following, we would like to test if this has an impact on the convergence speed of the other routers.

The link to RRC1 was configured to drop 15% of the packets to test whether a slow client can influence the convergence of the rest of the network. This causes TCP to resend the packets and increases the convergence time of that router, simulating a much slower route-reflector-client. All data from the measurement can be found in Appendix B under B.1 and B.3. The convergence time of the sessions between RR1 with RRC1 and RRC2 is 45.98 seconds and 46.35 seconds respectively, while all route-reflector-clients run at full speed. When we now take a look at the measurements where the link to RRC1 was configured to drop 15% of the packets, effectively making it miss some of the packets and also make it receive some packets a second time as the ACKs from RRC1 also experience the same loss. We can see that the session between RR1 and RRC1 now takes 159.98 seconds to converge, yet RR1 with RRC2 still takes 46.26 seconds, nearly identical as before. We can not see any effect on the session between RR1 and RRC2, even though that the one to RRC1 was more than tripled. This is also the case for all other sessions. We can therefore conclude that slow route-reflector-clients do not have any impact on the convergence performance of the rest of the network.

5.2 SWITCH Production Network

The production network of SWITCH is of course a lot more complex than the network that is simulated in GNS3. It is, therefore, necessary to adapt the basic idea on how to measure the convergence.

5.2.1 Measurement Method

The basic idea of injecting artificial flows to selected addresses within prefixes that are affected by the failure is still retained from subsection 5.1.3 but had to be altered to be doable in a real network. The goal of this measurement is to track the convergence of as many routers as we can while keeping the data acquisition as simple as possible. We, therefore, need to get a better understanding of where the flows will traverse, before and after the link failure. Figure 5.4 shows SWITCHes backbone topology as an OSPF tree, with BA3 as the root. It is, therefore, visible which path traffic will take towards DE-CIX for every point of the network before the failure. After

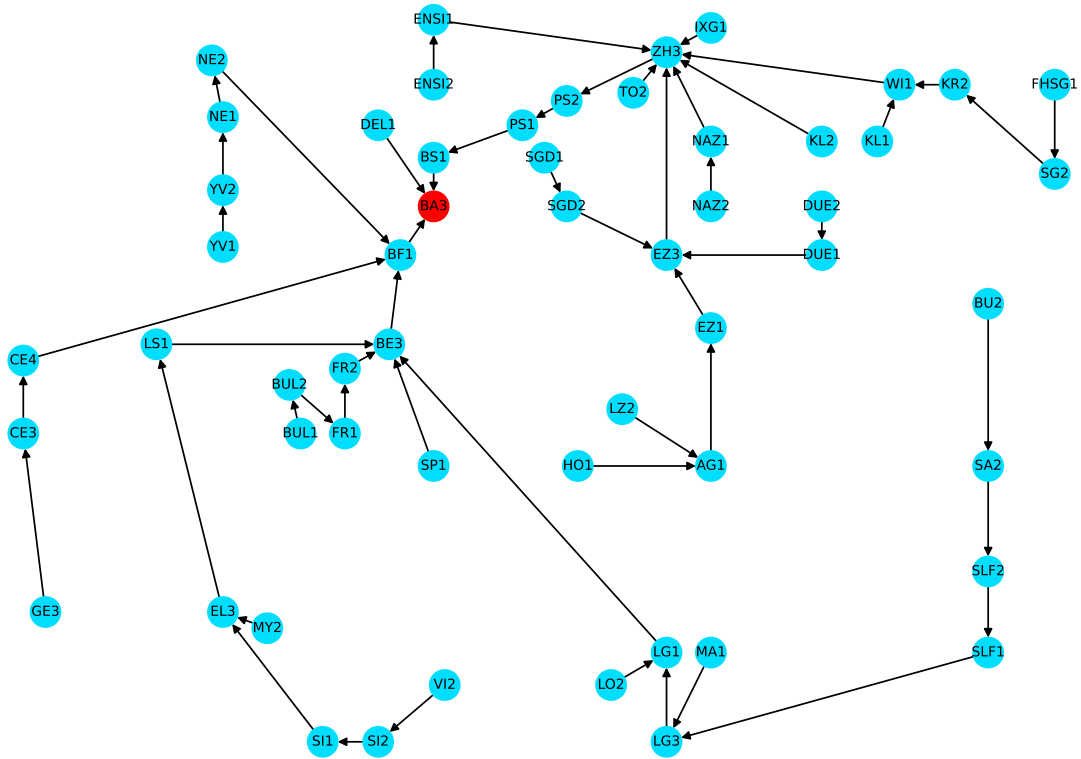


Figure 5.4: OSPF shortest-path tree to BA3 in the backbone of SWITCHes topology, showing how traffic will be routed to BA3

the convergence there are three exit points that the traffic can converge to. We constructed similar OSPF routing trees for these routers, which can be found in Appendix A in figures A.1 to A.3. Figure 5.5 shows the resulting graph when combining all four trees. Black paths remain the same for all exit routers, but red arrows indicate paths that are only present in some trees, but not in others. Obvious injection points for flows are routers that have red arrows only pointing away from them like FR1, SG2, VI2 and ENS1. The other injection points have been selected by comparing the different trees and choosing them so that all routers that have to change their routing table after the link failure, are covered by flows. The injection points are marked in figure 5.5 in turquoise.

The main limitation is that it is not feasible to capture all traffic on every link, as we did in GNS3. It could technically be done but would require a lot of hardware, configuration and time. We will therefore switch over to use another data collection method. Traceroute [12] is a utility that allows to track the path that IP packets take towards a given host. This is done by utilising the time to live (TTL) field of the IP header. It triggers ICMP time exceeded messages from the routers on the path by sending packets with increasing TTL to the host address, effectively getting the path by stringing together the IPs of the answering nodes in order of the respective TTL it took to get that message [12].

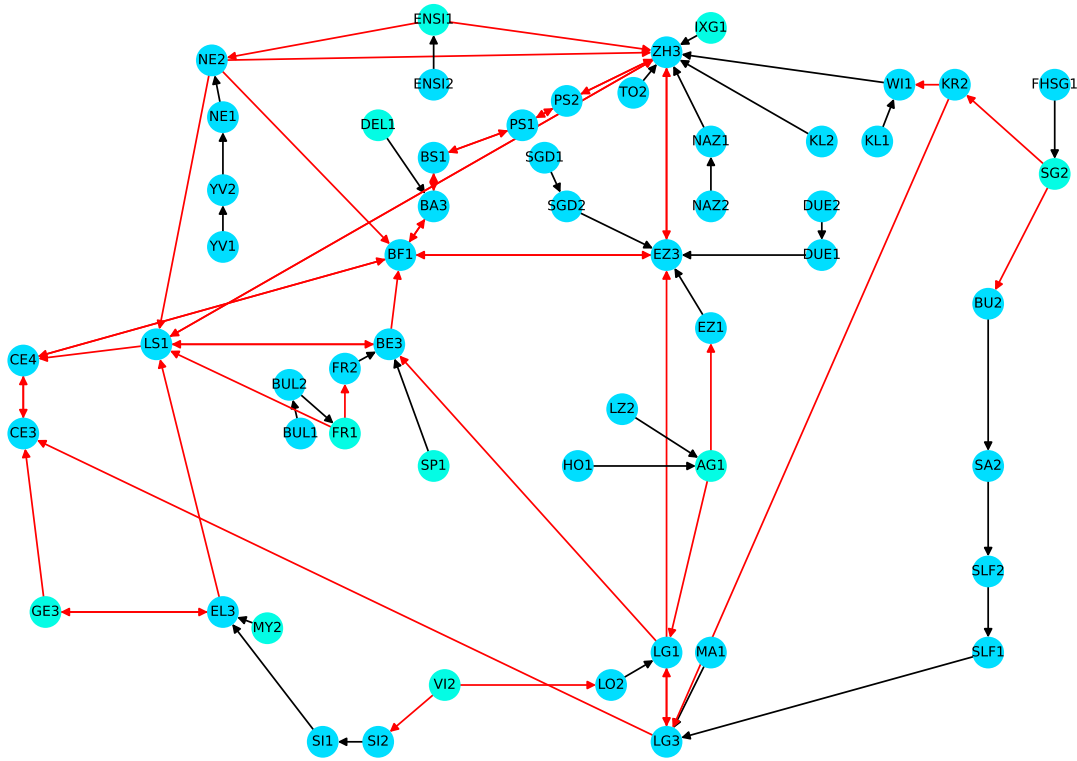


Figure 5.5: Links in the OPSF shortest paths trees that change depending on the exit router in **red** and links that are the same no matter the exit node in **black**. Injection points for flows are marked in **turquoise**

Flow Injection

We only have a single host connected to CE3 that we can use to inject the packets into the network. We therefore configured Generic Routing Encapsulation (GRE) tunnels that terminated at the host at CE3 on the ten injection point routers that have been selected. GRE was first developed by Cisco Systems but later standardized in RFC 2784 and is a simple encapsulation protocol [23]. The packets can be sent into those tunnels by encapsulating them in a GRE header and an IP header addressed to the injection router. Both headers will be decapitated from the packet on the other end of the tunnel. We use the IP address of the CE3 host in the inner IP header so that the ICMP messages are sent back to it for collection.

The packets that are sent also need to be changed to plain IP packets instead of UDP packets, as ICMP time exceeded messages will carry the original IP header and the following 64 bits of data [36]. We need to omit the UDP header and directly add our payload after the IP header to retain the data of our packet to be able to uniquely identify it. Our payload consists of four bytes for the sequence number, one byte for the TTL that the packet started with and one byte to identify the injection point that it was sent from, leaving 2 bytes unused of the available 8 bytes.

To be able to get a proper trace, it is necessary to send multiple packets with increasing TTLs for the same sequence number. The TTL goes up to nine as the longest path in the network is eight

hops long as can be seen in figure 5.4 from router BU2 to BA3. We also space out the sending of the packets for different prefixes in the 100ms interval after which we will send a new trace for each flow, instead of sending a burst of packets and waiting for the next slot. This is done so that the gap between any two packets is as large as possible as the routers on the path need the control plane to generate the ICMP packets. The generation is artificially slowed down by an ICMP rate limit to protect the routers from being overloaded by the construction of those answers. By maximising the inter-delay between any two packets, we also maximise the chance to get as many answers back as possible.

Data Collection

The data collection is done by storing the information of the received ICMP time exceeded messages. All IP addresses are translated to hostnames of the routers in SWITCHes network. An IP corresponds to an external hop if the IP can not be translated to an internal hostname and will therefore be saved as an IP.

Additionally, we also collect all flow packets that leave the network with a tcpdump collected by machines connect to an optical splitter that mirrors the traffic of the outgoing link. This is done on each of the four exit routers (BA3, CE3, ZH3, EZ3). The collected pcap files are processed and the information gets stored.

Filtering

The data that has been collected with the optical splitters and tcpdump does not need to be filtered anymore as they only contain valid data. It is only possible for them to record something if the traffic is leaving the network, which is always valid information. Once again the data lines for capture data will be interrupted if a gap of over 150ms is present between successive data points, to expose any packet loss that occurs as they are sent at a 100ms interval. The data from the traces however needs to be filtered to get rid of all incomplete paths. The reason that they occur, is due to the ICMP rate limits at the routers. We filter them by only accepting data that:

- Have an IP as their last hop, meaning they left the network and also have the information about the preceding TTL value so that the border router within SWITCHes network can be traced.
- Or a node is present more than once in the path, which shows that it is looping

Data lines for trace data will only be interrupted if the inter-delay between two successive data points is more than one second. The higher value has been chosen as trace data mostly consists of incomplete traces and this high noise needs to be filtered more aggressively. Therefore data lines are only interrupted if the gap between any two valid data points exceeds ten packets.

Both capture and trace data is shown in the same plots. Capture data is illustrated with solid, non-opaque colours as they do show reliable data. Traces are shown in the same colour for each address, but are semi-translucent and are overwritten by capture data if present. They do complete each other well as capture data is more accurate by showing the exact moment when a packet left the topology, but it can not show internal loops in the network, which traces can do. Combining them results in the accuracy of the capture data and the information range of the traces.

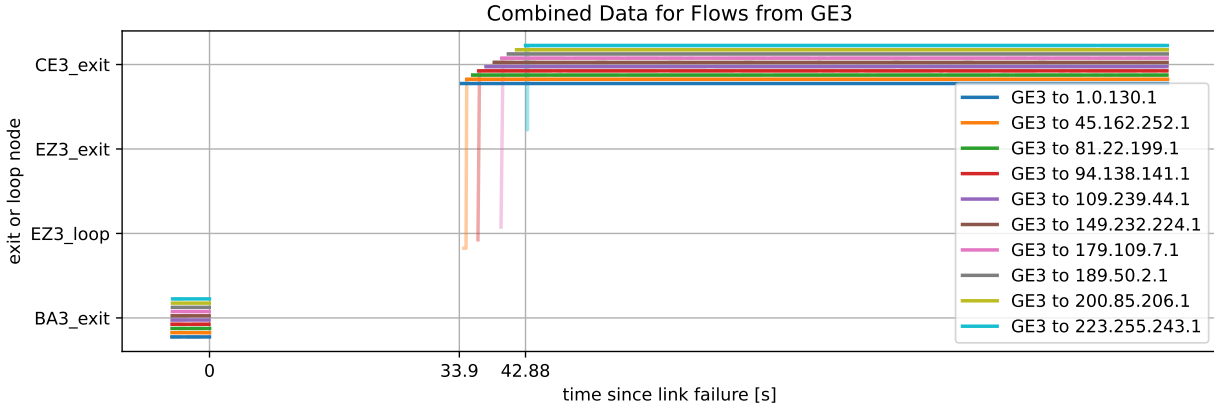


Figure 5.6: Combined trace (semi-transparent) and capture (solid) data, showing the convergence process for flows from GE3 with the original configuration. The y-axis shows where traffic is looping or able to exit the network.

5.2.2 Original Setup

We can not show all collected data here as we use ten injection points in the real network instead of three in the simulation. We will therefore only show some interesting data here and the rest can be found in Appendix C under C.2.1 and C.2.4. Figure 5.6 shows the convergence of flows injected by GE3. On the y-axis, we can see through which router the traffic is able to reach the destination or where it is looping. Loops are of course between two routers, but only the last one in the trace is shown here, as most of the time, not both of them will send ICMP messages during the same trace. We can therefore not directly see between which two routers the loop has formed.

We can see that all traffic is lost in the first 34 seconds. This is very similar for all injection points, as all of them see their first data between 33.89 and 36.61 seconds after the link failure. In the case of GE3 we can see that all prefixes converge rather quickly after that within only 9 seconds. Notice that the prefixes converge in ascending order as we have also seen in the simulation, confirming that lower prefixes are indeed better if one wants to have faster convergence.

But why is there not a single loop visible during the first 34 seconds? Comparing graphs from different injection points shows, that the data lines seem to reappear in the same way for all of them. In 5.2.3 we will see that traffic is black-holed by routers when the next-hop has been invalidated while the routes are still in the FIB and making it impossible to use the default route. Combining the observation of the data reappearance pattern, the black-holing during FIB convergence and some manual checks of the routes that traffic takes during this blank period leads to the conclusion that BA3 is black-holing all traffic in the first 34 to 43 seconds, depending on the prefix. Therefore BA3 does not seem to be updating its FIB for almost 34s and then converges within 9 seconds. This could lead to the assumption that BA3 is not using *BGP Next-Hop Address Tracking*, which would mean that the BGP scanner will poll from time to time to check if next-hops have changed and then update the routes accordingly. But this is disproven by the fact that the BGP log clearly shows a dramatic increase in BGP updates and withdraws from BA3 in less than a second after the failure. We can not say conclusively what the reason is, that BA3 takes so long to start updating its FIB as we have no insight into the implementation that Cisco has chosen. We can only deduce from the data, that the FIB updates are retained for over half a minute, even though there are no dampening mechanisms configured by SWITCH.

The lightly coloured lines for some of the addresses show, that the flows are shortly looping at EZ3 before converging to CE3. We can infer from that, that BF1 converges a bit faster than CE3 and CE4, which leads to the short loop at EZ3, because the closest default route of BF1 is EZ3 before all traffic is flowing via CE3 as that router converges too and uses its own default route instead sending it in the direction of BA3 and therefore BF1. Meaning that all traffic from GE3 has reached its steady-state again after 42.88 seconds.

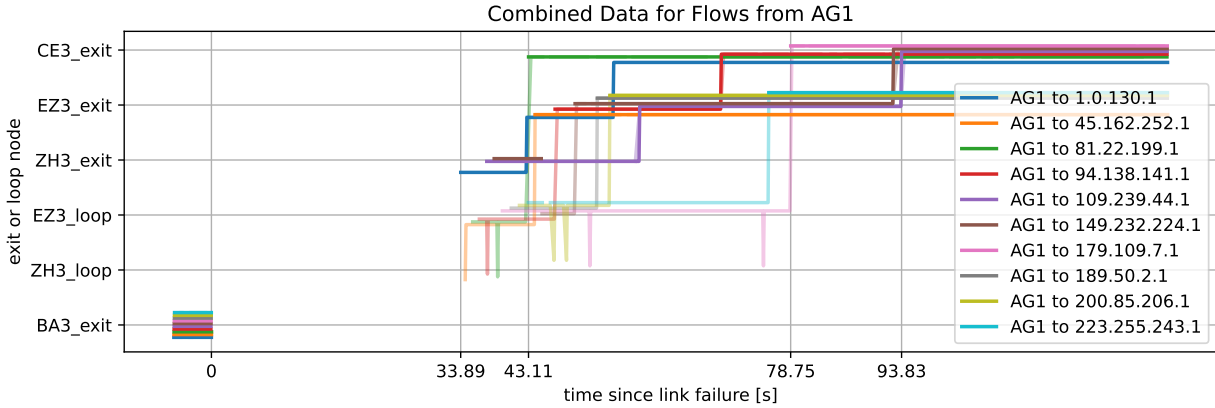


Figure 5.7: Combined trace (semi-transparent) and capture (solid) data, showing the convergence process for flows from AG1 with the original configuration. The y-axis shows where traffic is looping or able to exit the network.

Flows from AG1, seen in figure 5.7, however, do neither converge so fast nor does the process look so clean as for GE3. Here we can see how most flows start to loop between EZ3 and ZH3 when they come back up. ZH3 is much less represented in the plot, as it is one of the routers with a significantly lower ICMP rate limit, therefore answering much more rarely.

A very interesting finding for traffic from AG1 was, that when we look for example at the traffic from *AG1 to 223.255.243.1*, that when it first converges to use EZ3s default route, the internal path looks like the following: *AG1→EZ1→EZ3→ZH3→EZ3→External*. This would make sense if this trace would cover the exact moment when EZ3 converged, but the path looked like this for over 11 seconds. Digging a bit deeper revealed, that EZ3 uses multiple linecards which each have their own FIB. The linecard that receives the packet from EZ1 has not converged yet and is trying to route the packet to BA3 via ZH3, ZH3 has already converged and sends it back to use the default route via EZ3. Now the second linecard gets the packet and sends it to the transit-provider as this one has also already updated its FIB. Leading to this dogleg routing

Figure 5.8 displays the convergence for the address 149.232.224.1 from all injection points. This allows us to more easily see when different paths converge. We can see that all locations are fully connected after 49.44 seconds, but it takes nearly double that time before the final state is reached. This is because SWITCH is hiding BGP additional paths. This is nothing special, but rather the standard operation of BGP. But this leads to this behaviour as in the beginning everyone is falling back to the default route when the path via BA3 is withdrawn. After that CE3 will recalculate its best path and send the route that it has learnt to the rest of the network. Now the rest of the routers start to use CE3 instead of the default route as it advertises a more specific prefix. This is no problem though, as the connection is retained when switching over from one exit node to the other.

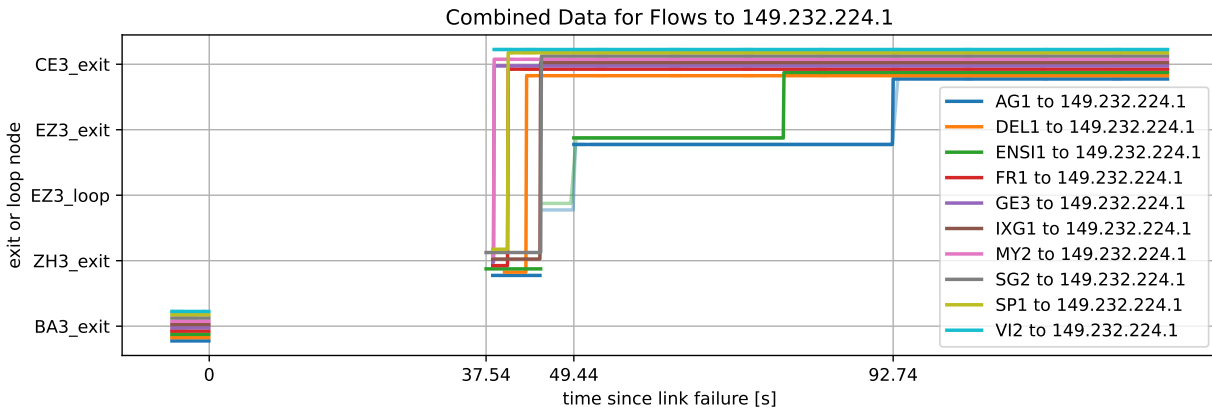


Figure 5.8: Combined trace (semi-transparent) and capture (solid) data, showing the convergence process for flows towards 149.232.224.1 with the original configuration. The y-axis shows where traffic is looping or able to exit the network.

5.2.3 OSPF Distribution

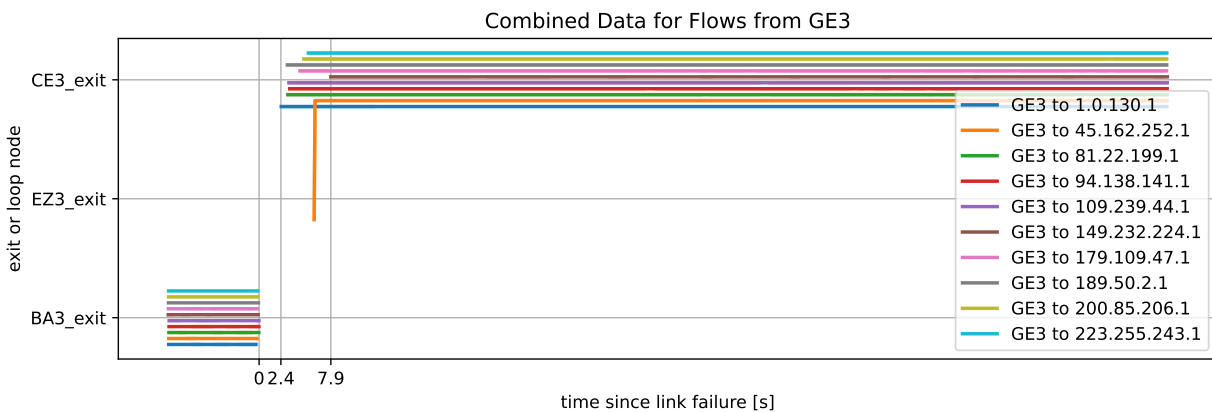


Figure 5.9: Combined trace (semi-transparent) and capture (solid) data, showing the convergence process for flows from GE3 with the OSPF failure distribution. The y-axis shows where traffic is looping or able to exit the network.

Figure 5.9 shows how much the OSPF distribution can do to improve the convergence. It now only takes 7.9 seconds instead of the 42.88 seconds it took to converge with the original configuration. But this is the exception. The data from the other injection points show a convergence time of 42 to 68 seconds. This is still an improvement over the 74 to 75 seconds that we saw with the original convergence, but a lot less drastic than the improvement in the simulation.

There are even two injection points that got worse. MY2 and VI2 both converged within 43.18 seconds with the original configuration and now take 62.03 and 53.84 seconds respectively. Taking a look at figure 5.10 can help us to understand why this can happen. Here we see that the flows for all addresses reappear right around one second after the failure. All of them seem to loop at ENSI1, consulting the input data manually reveals, that all traffic is being black-holed by the

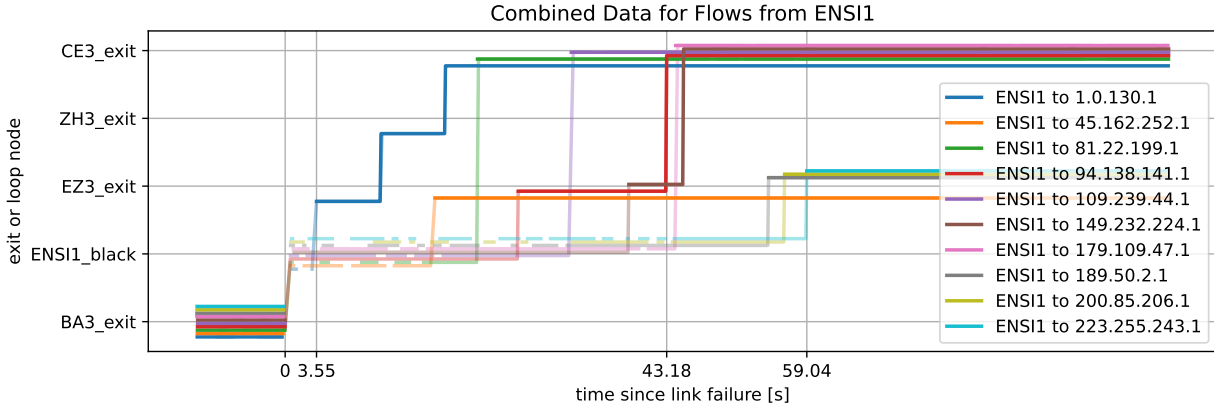


Figure 5.10: Combined trace (semi-transparent) and capture (solid) data, showing the convergence process for flows from ENSI1 with the OSPF failure distribution. The y-axis shows where traffic is looping, being black-holed or able to exit the network.

injection router itself. We introduced the new extension *_black* to the y-axis labels to mark traffic that is being black-holed at a router. This is happening because OSPF informs it that the next-hop for those prefixes that are routed to DE-CIX are no longer reachable. Now the router invalidates those next-hops at a moments notice. The addresses do still match with the routes in the FIB but are then dropped by the router as it can not reach the specified next-hop and it does not seem to be able to fall back to a shorter prefix that would match.

At 3.55 seconds after the failure we can see how the first route has been deleted from the FIB and 1.0.130.1 can now be routed to the default route as it no longer matches with a route to DE-CIX that is no longer reachable. We can see that more and more routes come back up as their respective invalidated route in the FIB is deleted. The convergence is therefore lower bounded by the convergence speed of the FIB of each router on the path. Most injection point routers, except GE3, are located at lower capacity links and therefore also run slower, older hardware. So they generally are the slowest routers in the routing path and this also explains how MY2 and VI2 can be slower to regain connection with the improved configuration.

Comparing their convergence behaviour in the original setup shows that they both converge nearly identical and that all their flows are sent to CE3 (Appendix C, figure C.25 and C.28). Taking the routing paths from these two injection points to BA3 and CE3 into consideration makes it clear how the original setup can converge faster. MY2 and VI2 will still route the same for both locations (figure 5.4 and A.1). The routers that need to converge for the flows to reach CE3 in the original setup are EL3, GE3 and CE3, which are all large, more expensive routers, as they are part of 100 Gigabit paths. But now that OSPF is used to propagate the link failure, it does no longer only depend on these routers, but on MY2 and VI2 themselves as well. This is the reason for the slower converges as it takes those two routers longer until the FIB is updated than it took the faster routers on the route before. Even though their convergence only started after 33.89 seconds after the failure.

Overall the improvements seem to be only minimal with the adjusted configuration that has been proposed. This is vastly different to the behaviour seen in GNS3 and highlights why tests with the production hardware will always be necessary to get a definitive answer if one configuration is superior to another. The reason that the simulation is converging in less than 100ms and the real

hardware is taking 9 to 68 seconds could be that routers in GNS3 are emulated, and in the end, there is no TCAM to store the FIB, but this will be done in RAM. Updating RAM is much faster than TCAM and could explain why the simulation is so fast in comparison.

But this does not mean that the OSPF distribution of the failure can't improve the convergence for SWITCH, as will be shown in chapter 6.

5.3 Additional Findings

This section covers the noteworthy findings that have been made during this thesis which are not directly related to the convergence problem.

5.3.1 The Effect of different Adapter Types on the Bandwidth in GNS3

During initial testing in GNS3, it was discovered that the available bandwidth is very low and also did not seem consistent. Originally the C7200-IO-FE adapter was used to connect the Ubuntu hosts to the routers and the PA-POS-OC3 adapters between routers. The latter choice was made as these are optical interfaces and SWITCHes real network also uses optical interfaces. The choice for the Fast-Ethernet has only been made, as it is the standard first interface of C7200 routers and can be used to connect to the hosts. The optical interfaces are naturally serial interfaces and are incompatible with the hosts out of the box.

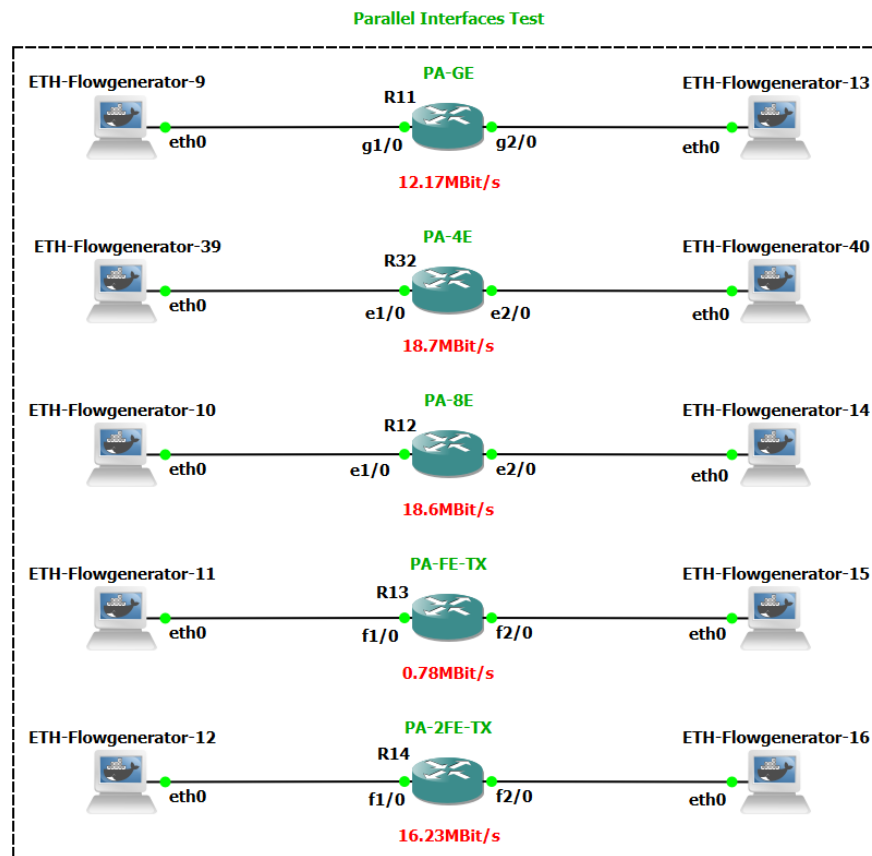


Figure 5.11: Bandwidth test of different parallel adapters available for C7200 routers

We tested that the hosts would not bottleneck the performance of the router by directly connecting them, which resulted in over 1.8 gigabits per seconds of available bandwidth. This is two orders of magnitude faster than any router interface that will be shown in the following. All tests will be run by using iperf on the hosts and testing the bandwidth three times, taking the average over these measurements.

The first test setup, shown in figure 5.11 is intended to test the bandwidth of the parallel adapters available for the C7200 routers. We can see that the four and eight gigabit Ethernet adapters performed best, with a bandwidth of over 18 MBit/s. By far the slowest adapter is the PA-2FE-TX one that only managed 0.78 MBit/s.

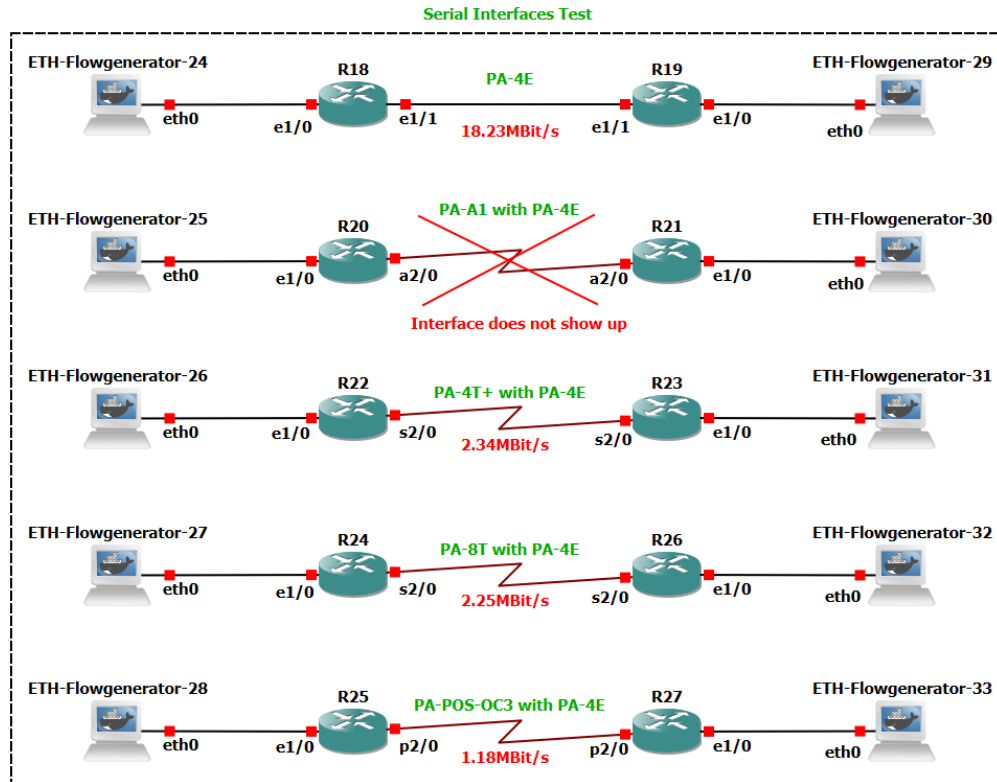


Figure 5.12: Bandwidth test of different serial adapters available for C7200 routers

Figure 5.12 shows the setup to test the serial adapters. A second router had to be introduced as these can not be directly connected to the hosts. Therefore only the link between the two routers will connect the interfaces that should be tested. PA-4E adapters will be used between routers and hosts as these are the fastest adapters that have been tested and will provide the best chance for the serial adapters to reach their limit.

The topmost setup is using PA-4E also on the middle link to verify that no unexpected throttling will be seen by this slightly enlarged testing topology. The speed is very similar with the additional link at 18.23 MBit/s as we can see. PA-A1 did not even show up and could therefore not be tested. PA-8T and PA-4T+ did both only deliver a bit over 2 MBit/s and are considerably slower than the PA-4E adapter. Even worse performs the PA-POS-OC3 adapter that has been chosen in the beginning. This type of interface limits the bandwidth to 1.18 MBit/s.

The first slot in C7200 routers can only be equipped with adapters that are unique for this slot. Figure 5.13 shows the setup for this test. For single interface adapters, it was necessary to use

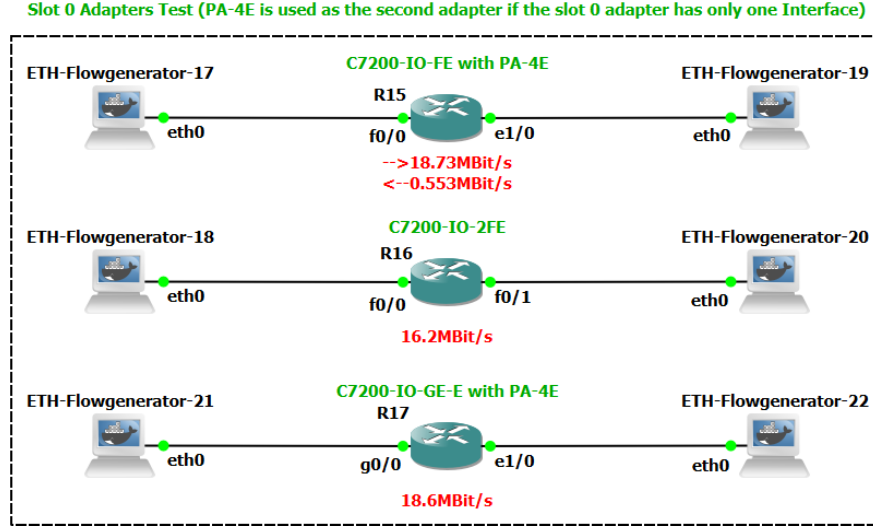


Figure 5.13: Bandwidth test of different adapters for slot0 available for C7200 routers

PA-4E ones on the second link as it is not possible to use two slot0 adapters.

A very strange behaviour can be seen by the C7200-IO-FE adapter that has been used initially in GNS3. In the outgoing direction, it seems like PA-4E is bottlenecking the bandwidth of this interface. But when testing in the other direction, we can see that only 0.553 MBit/s can be sent through that adapter. According to this testing, it seems like the two initially selected adapters are in fact the two slowest possible ones of them all.

The last test is done to see if there is a difference between using multiple interfaces on the same adapter or if it is advantageous to only use a single interface on them. Figure 5.14 shows the setup for this. The first topology is using two interfaces on the same adapter and the second setup uses only a single interface but two adapters to connect to the hosts. We can see that there is not much of a difference between the two. Using multiple interfaces from the same adapter is only slightly advantageous over using different adapters.

The lower two topologies do test the same thing again but with four routers as this is the longest chain of routers that has been used to simulate SWITCHes setup. We can see that the difference between the two configurations has grown more noticeable in this scenario. Both of them lost a bit of bandwidth, but using different adapters seems to cause more degradation than using multiple interfaces on the same adapter.

In the end, we chose to use PA-4E adapters for all tests during this thesis and tried to use all interfaces on an adapter before using a second one.

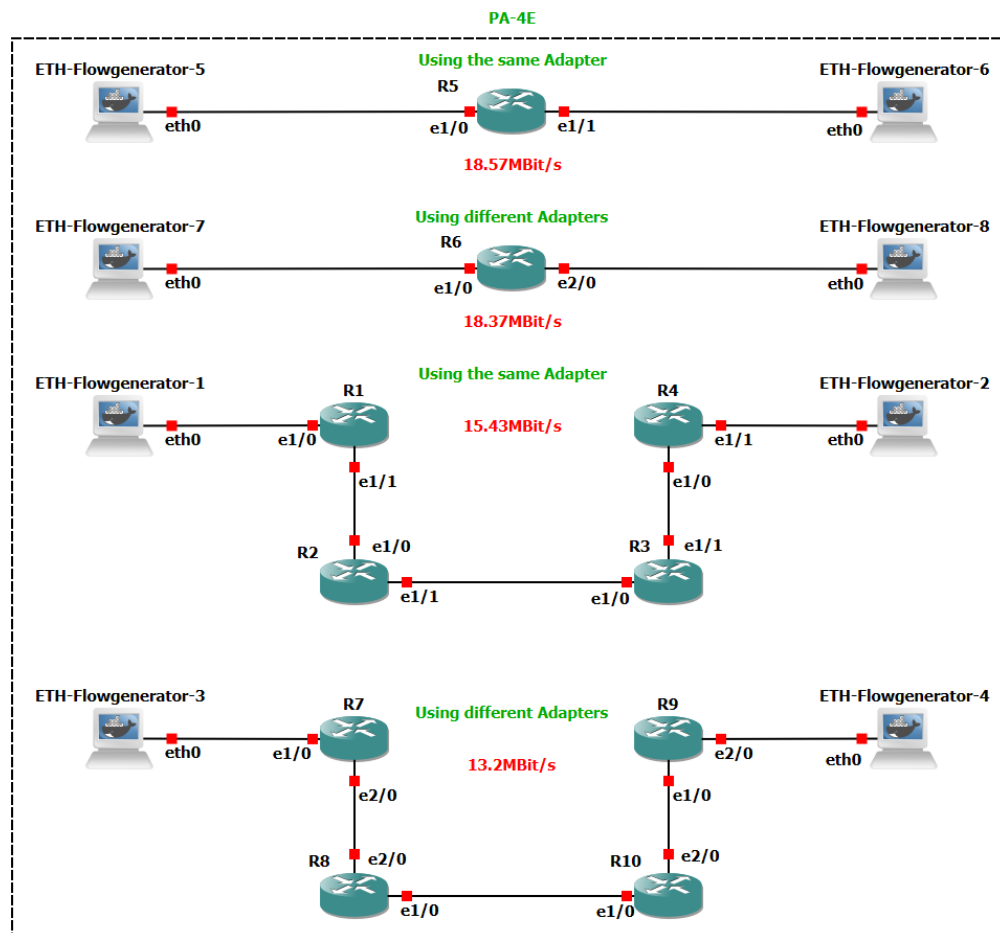


Figure 5.14: Bandwidth test of different topologies and configurations of the PA-4E adapter for C7200 routers

Chapter 6

Outlook

The setup of SWITCH has been converted from BGP propagating the failure, which does not scale well as it works on a per-prefix level, to a configuration that immediately notifies all routers in the network about which next-hops can not be reached anymore. This means each router can converge as fast as it is possible for that hardware. While this did indeed decrease the convergence time on average, it also increased it for two out of ten injection points. We will in the following go over some improvements that could improve the convergence further.

6.1 Further Practical Improvements for SWITCH

SWITCHes current setup can not benefit from PIC as already explained in section 4.3. This is because they are operating in standard BGP mode, which means that only the best paths will be announced. The disadvantage is that only the border router that learns an alternate path via an external peer knows about this route. PIC can therefore only improve the converge for this single router.

Analysing the BMP log that was recorded during the outage of the 22nd of June 2021, shows that SWITCH actually knows an alternate path for 50.4% of the prefixes that are learnt via DE-CIX. Simply enabling the advertisement of such routes will decrease the convergence time for them to less than 2.5 seconds according to the data that was recorded with the OSPF distribution setting. Most fast routers will be able to converge in less than one second as only the single OSPF link state update has to be processed.

Due to the time constraints of this thesis, it was not possible to implement and test this yet, but it is easy to configure, does not produce any extra cost and only requires some additional space in RAM, but should deliver considerable improvements.

6.2 Possible Improvements by Manufacturers

The limiting factor now is the time it takes to remove the invalidated routes from the FIB so the data plane is able to fall back to the default route for prefixes that can no longer be routed via DE-CIX. In the following, we would like to touch on two possible solutions to that problem which could be implemented by manufacturers.

6.2.1 Extension of PIC

Prefix Independent Convergence or PIC is a feature that is implemented in Cisco routers for over 20 years now. But it is still not standardized and only an informational internet draft exists [17]. Even that document does not specify how one can obtain a backup next-hop for a prefix that should be protected. Ciscos documentation states that BGP Fast Reroute is providing the backup or alternate path to PIC to be installed [4]. While no definitive answer could be found on how BGP Fast Reroute calculates this path, the little available information pointed to the assumption that it only provides an alternate path to the exact same prefix. While this is useful for prefixes that are learnt in the same length at at least a second location, it won't allow to reroute via a less-specific prefix.

We would therefore like to propose an algorithm that tries to remedy the assumed limitation of PIC, being that it can only protect prefixes that have an alternate route for the exact same prefix learnt via BGP. It is loosely oriented on an algorithm that was proposed in a technical report about PIC [39].

1. Run normal BGP decision process and install routes
2. With lower priority in the background:
 - (a) Exclude all routes that use the same next-hop and the same BGP extended community from consideration
 - (b) Treat the prefix as an address and find the next-hop that would be taken to route that address
 - (c) Install this next-hop, if found, as a backup

Excluding not only routes with the same next-hop, but also with the same BGP extended community, allows the network operator to bundle routes that use the same link, the same adapter, or even the same router. This means he can choose between the trad-off of having more routes available for the backup calculation and having protection against more severe kinds of failures.

Treating the prefix as an address that needs to be routed in the absence of the excluded routes allows us to find the next-longest prefix that can route the traffic. It even allows for the default route as a backup for prefixes that do not have a less specific counterpart in the routing table.

This algorithm provides an advantage over standard PIC, even if the assumption about the current limitation of it being only able to provide a fast data-plane convergence for prefixes with an alternate path to the exact same prefix is wrong. The presented backup calculation is not only limited to routes that are learnt via BGP but can also utilize ones that are learnt by other protocols as it uses the whole routing table for its calculation.

Having said this, if PIC could use less specific prefixes for its backup computation in its current implementation, it would be possible to improve the convergence of SWITCHes network even further, by adding the default route distribution to BGP as well, as this would make it available for usage for the prefixes that are only learnt in a single location.

6.2.2 New FIB Architecture

The main reason why updating a FIB takes so long, is because it needs to be ordered from longest prefixes to shortest prefixes as the TCAM where the FIB is stored always returns the first matching entry. An insertion or a deletion of an entry in the FIB is of the complexity $O(n)$. This could be remedied if manufacturers would implement a new TCAM architecture as proposed by Reza Avazeh

and Nasser Yazdani from the University of Tehran. Their architecture achieves a complexity of $O(1)$ by not requiring the TCAM to be ordered. It is therefore possible to delete or add an entry at an arbitrary location, which only requires a single operation [16].

6.3 Improving the Convergence for Peers at an IXP

While previously suggested improvements help to reduce the time for SWITCHes network to converge on a new path to reroute their traffic, it still does not fully solve the convergence problem. Because even if SWITCH is able to reach their destinations again, it still does not guarantee that they will get any data back from that location. This is because the reverse-path may still lead through the failed link as the peers on the other side did not notice that SWITCH is no longer reachable through DE-CIX. The solution for peers that they peer directly with, is to set up BFD. But peers that are connected through a route server are not as easy to inform about the link failure.

A current internet-draft [21] is trying to remedy this by implementing a new mechanism that allows the route server of an IXP to automatically provision BFD sessions between its clients and have them report the status of those connections back to it. This has the potential to significantly decrease the time it takes for route-server-clients to learn about connection disruptions of other clients.

Chapter 7

Summary

During the *Route Convergence Optimization in the SWITCH Network* we went over the basic functionality of protocols that control the current convergence behaviour of the network and explained the different steps that have to be processed in order to regain connectivity. A thorough analysis of the BMP log that was captured during the link failure has been conducted and combined with automatically extracted information from the configuration of the routers to identify the failure propagation as the main problem. A migration to an OSPF distribution has been proposed and tested in GNS3 as well as the real network with self-developed measurement methods. The current improvement in convergence time is small, but a simple solution to decrease this to sub-second or worst case less than 3 seconds convergence for over 50% of their prefixes has been presented.

Bibliography

- [1] Autonomous system numbers. <https://www.arin.net/resources/guide/asn/>, accessed 06.11.21.
- [2] *BGP PIC (Prefix Independent Convergence) Edge for IP and MPLS-VPN*. https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/iproute_bgp/configuration/xe-3s/asr903/17-1-1/b-irg-xe-17-1-asr903/b-irg-xe-17-1-asr903_chapter_01.pdf, accessed 26.10.21.
- [3] *BGP Support for Next-Hop Address Tracking*. https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/iproute_bgp/configuration/15-mt/irg-15-mt-book/irg-nexthop-track.pdf, accessed 26.10.21.
- [4] Chapter: Bgp pic edge for ip and mpls-vpn. https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/iproute_bgp/configuration/xe-3s/irg-xe-3s-book/irg-bgp-mp-pic.html, accessed 02.11.21.
- [5] Cidr report for 24 oct 21. <https://www.cidr-report.org/as2.0/>, accessed 24.10.21.
- [6] Exabgp. <https://github.com/Exa-Networks/exabgp>, accessed 27.10.21.
- [7] GÉant networks. <https://www.geant.org/Networks>, accessed 24.10.21.
- [8] Gns3 api. <https://api.gns3.net/en/2.2/general.html>, accessed: 29.09.2021.
- [9] Route servers. <https://www.swissix.ch/infrastructure/routeserver>, accessed 24.10.21.
- [10] Scapy. <https://scapy.net/>, accessed 27.10.21.
- [11] Switchlan: the smart data network. <https://www.switch.ch/services/network/>, accessed: 17.09.2021.
- [12] *Traceroute for Linux*. <https://manpages.debian.org/bullseye/traceroute/traceroute.1.en.html>, accessed 30.10.21.
- [13] What is gns3? <https://docs.gns3.com/docs/#what-is-gns3>, accessed: 29.09.2021.
- [14] Working for a better digital world. <https://www.switch.ch/about/foundation/>, accessed: 17.09.2021.
- [15] ALABDULKREEM, E., AL-RAWESHIDY, H., AND ABBOD, M. Mrai optimization for bgp convergence time reduction without increasing the number of advertisement messages. *Procedia Computer Science* 62 (2015), 419–426. Proceedings of the 2015 International Conference on Soft Computing and Software Engineering (SCSE’15).

- [16] AVAZEH, R., AND YAZDANI, N. A new tcam architecture for ip routing with update complexity equal to $O(1)$. *Canadian Journal of Electrical and Computer Engineering* 43, 4 (2020), 207–217.
- [17] BASHANDY, A., FILSFILS, C., AND MOHAPATRA, P. Bgp prefix independent convergence. Internet-Draft draft-ietf-rtgwg-bgp-pic-17, IETF Secretariat, October 2021. <https://www.ietf.org/archive/id/draft-ietf-rtgwg-bgp-pic-17.txt>.
- [18] BATES, T., CHANDRA, R., AND CHEN, E. Bgp route reflection - an alternative to full mesh ibgp. RFC 2796, RFC Editor, April 2000.
- [19] BONAVENTURE, O., FILSFILS, C., AND FRANCOIS, P. Achieving sub-50 milliseconds recovery upon bgp peering link failures. *IEEE/ACM Transactions on Networking* 15, 5 (2007), 1123–1135.
- [20] BREMLER-BARR, A., AFEK, Y., AND SCHWARZ, S. Improved bgp convergence via ghost flushing. In *IEEE INFOCOM 2003. Twenty-second Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE Cat. No.03CH37428)* (2003), vol. 2, pp. 927–937 vol.2.
- [21] BUSH, R., HAAS, J., SCUDDER, J., NIPPER, A., AND DIETZEL, C. Making route servers aware of data link failures at ixps. Internet-Draft draft-ietf-idr-rs-bfd-09, IETF Secretariat, September 2020. <http://www.ietf.org/internet-drafts/draft-ietf-idr-rs-bfd-09.txt>.
- [22] FABRIKANT, A., SYED, U., AND REXFORD, J. There’s something about mrai: Timing diversity can exponentially worsen bgp convergence. In *2011 Proceedings IEEE INFOCOM* (2011), pp. 2975–2983.
- [23] FARINACCI, D., LI, T., HANKS, S., MEYER, D., AND TRAINA, P. Generic routing encapsulation (gre). RFC 2784, RFC Editor, March 2000. <http://www.rfc-editor.org/rfc/rfc2784.txt>.
- [24] GILL, R., PAUL, R., AND TRAJKOVIĆ, L. Effect of mrai timers and routing policies on bgp convergence times. In *2012 IEEE 31st International Performance Computing and Communications Conference (IPCCC)* (2012), pp. 314–323.
- [25] HUSTON, G. Ipv6 cidr report for 24 oct 21. <https://www.cidr-report.org/v6/as2.0/>, accessed 24.10.21.
- [26] KATZ, D., AND WARD, D. Bidirectional forwarding detection (bfd). RFC 5880, RFC Editor, June 2010.
- [27] KOPKA, M. Ip routing fast convergence, 2013. https://www.cisco.com/c/dam/global/cs_cz/assets/ciscoconnect/2013/pdf/T-SP4-IP_Routing_Fast_Convergence-Miloslav_Kopka.pdf, accessed: 23.10.21.
- [28] LABOVITZ, C., AHUJA, A., BOSE, A., AND JAHANIAN, F. Delayed internet routing convergence. *IEEE/ACM Transactions on Networking* 9, 3 (2001), 293–306.
- [29] LEE, S., LEVANTI, K., AND KIM, H. S. Network monitoring: Present and future. *Computer Networks* 65 (2014), 84–98.
- [30] LEINEN, S. De-cix outage 12 january 2021, January 2021.

- [31] MAO, P., BIRKNER, R., HOLTERBACH, T., AND VANBEVER, L. Boosting the bgp convergence in sdxes with swift. In *Proceedings of the SIGCOMM Posters and Demos* (New York, NY, USA, 2017), SIGCOMM Posters and Demos '17, Association for Computing Machinery, p. 1–2.
- [32] MISRA, S., AND GOSWAMI, S. *Interior Gateway Protocols*. 2014, pp. 131–157.
- [33] MOY, J. Ospf version 2. STD 54, RFC Editor, April 1998. <http://www.rfc-editor.org/rfc/rfc2328.txt>.
- [34] PEI, D., AZUMA, M., MASSEY, D., AND ZHANG, L. Bgp-rcn: improving bgp convergence through root cause notification. *Computer Networks* 48, 2 (2005), 175–194.
- [35] PEI, D., ZHAO, X., MASSEY, D., AND ZHANG, L. A study of bgp path vector route looping behavior. In *24th International Conference on Distributed Computing Systems, 2004. Proceedings.* (2004), pp. 720–729.
- [36] POSTEL, J. Internet control message protocol. STD 5, RFC Editor, September 1981. <http://www.rfc-editor.org/rfc/rfc792.txt>.
- [37] REKHTER, Y., LI, T., AND HARES, S. A border gateway protocol 4 (bgp-4). RFC 4271, RFC Editor, January 2006. <http://www.rfc-editor.org/rfc/rfc4271.txt>.
- [38] SAHOO, A., KANT, K., AND MOHAPATRA, P. Improving bgp convergence delay for large-scale failures. In *International Conference on Dependable Systems and Networks (DSN'06)* (2006), pp. 323–332.
- [39] SCHRIECK, V. V. D., BONAVENTURE, O., FRANCOIS, P., FILSFILS, C., MOHAPATRA, P., BETTINK, J., DHARWADKAR, P., VRIENDT, P. D., AND TSIER, Y. Bgp prefix independent convergence (pic) technical report. Tech. rep., Université catholique de Louvain (UCL), November 2007. https://dial.uclouvain.be/pr/boreal/object/boreal%3A166710/datastream/PDF_01/view.
- [40] SCUDDER, J., FERNANDO, R., AND STUART, S. Bgp monitoring protocol (bmp). RFC 7854, RFC Editor, June 2016.
- [41] SERMPEZIS, P., AND DIMITROPOULOS, X. Can sdn accelerate bgp convergence?— a performance analysis of inter-domain routing centralization. In *2017 IFIP Networking Conference (IFIP Networking) and Workshops* (2017), pp. 1–9.
- [42] TAMMANA, P., AGARWAL, R., AND LEE, M. Distributed network monitoring and debugging with switchpointer. In *15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18)* (Renton, WA, Apr. 2018), USENIX Association, pp. 453–456.
- [43] VANBEVER, P. D. L. Lecture in communication networks, Feb 2020.
- [44] WANG, L., SARANU, M., GOTTLIEB, J. M., AND PEI, D. Understanding bgp session failures in a large isp. In *IEEE INFOCOM 2007 - 26th IEEE International Conference on Computer Communications* (2007), pp. 348–356.

Appendix A

OSPF Trees for additional Exit Nodes

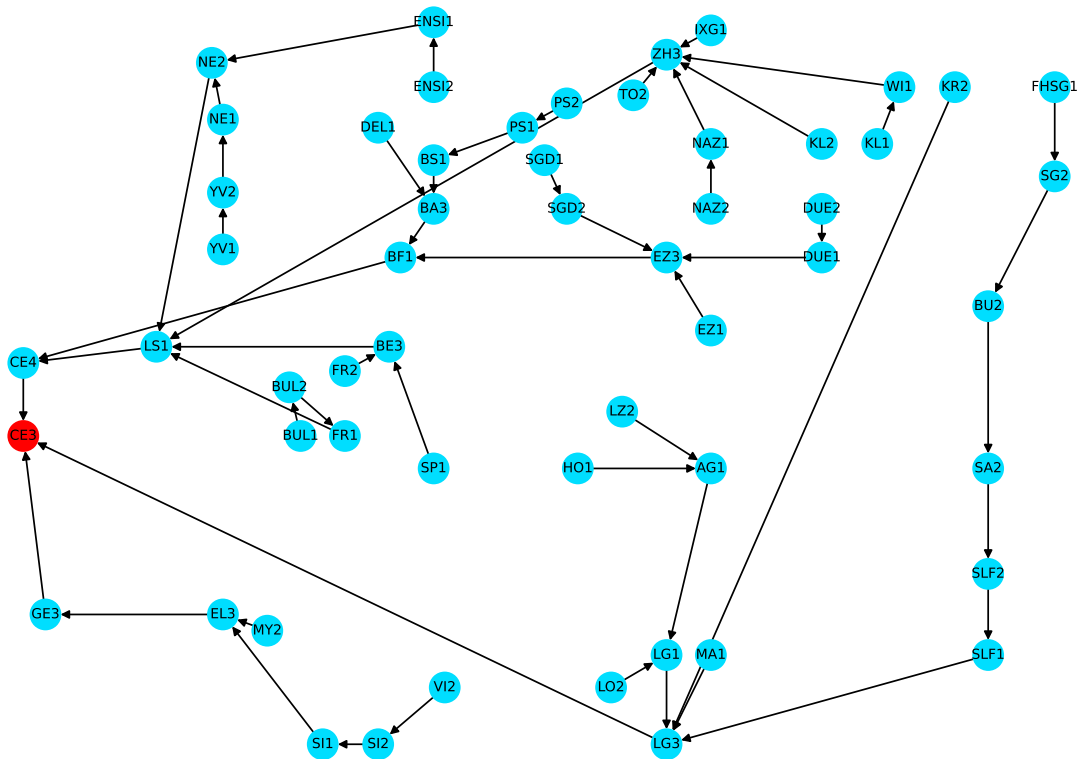


Figure A.1: OSPF paths to BA3 in the backbone of SWITCHes topology

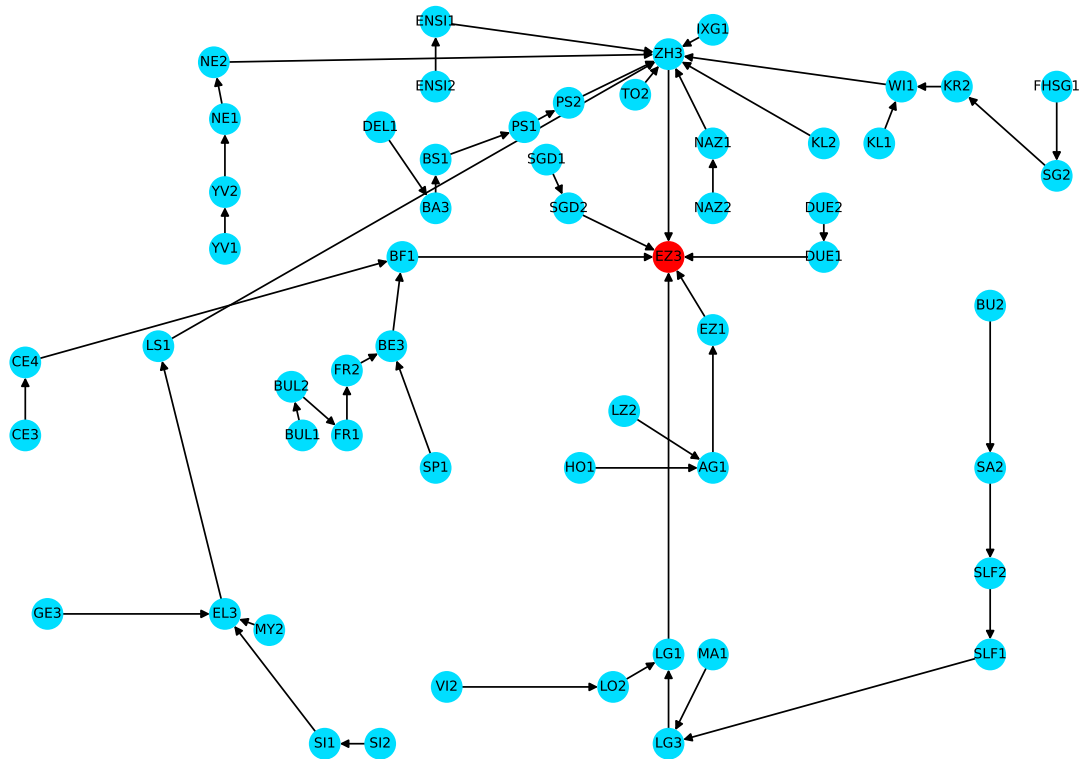


Figure A.2: OSPF paths to BA3 in the backbone of SWITCHes topology

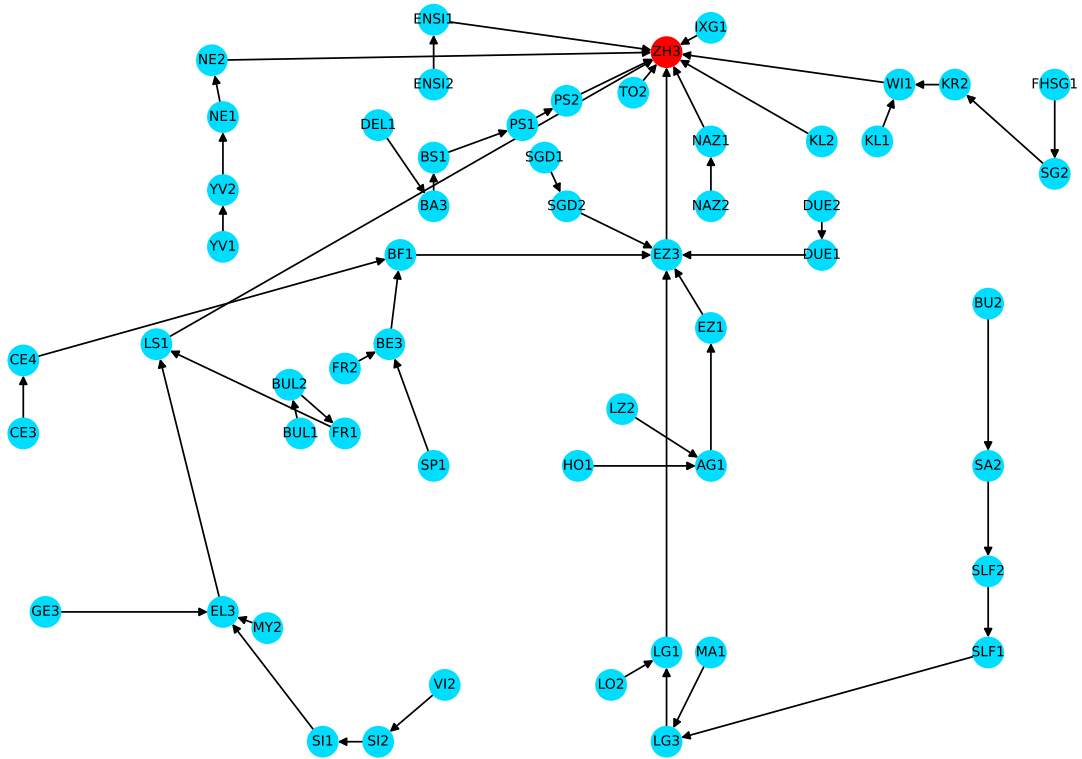


Figure A.3: OSPF paths to BA3 in the backbone of SWITCHes topology

Appendix B

BGP Session Tracking

B.1 Time of First and Last BGP UPDATE Messages in GNS3 without Flows

The state of the link changed at:

epoch time: 2021-11-04 15:11:10.647196

Session BA3_to_RR1:

link_RR1_to_BA3:

started after: 1.0933722 seconds

and ended after: 27.9624302 seconds

Session BA3_to_RR2:

link_RR1_to_BA3:

started after: 1.1042572 seconds

and ended after: 26.8016922 seconds

link_RR2_to_RR1:

started after: 1.1159202 seconds

and ended after: 26.8100842 seconds

Session BA3_to_RR3:

link_RR1_to_BA3:

started after: 1.1049052 seconds

and ended after: 26.8927042 seconds

link_RR3_to_RR1:

started after: 1.1159102 seconds

and ended after: 26.8939522 seconds

Session RR1_to_BA3:

link_RR1_to_BA3:

started after: 1.3782952 seconds

and ended after: 35.9169662 seconds

Session RR1_to_CE4:

link_RR3_to_CE4:

started after: 1.3816872 seconds

and ended after: 46.6562422 seconds

link_RR3_to_RR1:

started after: 1.3787202 seconds

and ended after: 46.6461882 seconds

B.1. TIME OF FIRST AND LAST BGP UPDATE MESSAGES IN GNS3 WITHOUT FLOWSV

Session RR1_to_RR2:

link_RR2_to_RR1:

started after: 1.3579162 seconds

and ended after: 34.9488732 seconds

Session RR1_to_RR3:

link_RR3_to_RR1:

started after: 1.3684562 seconds

and ended after: 35.2117492 seconds

Session RR1_to_RRC1:

link_RR2_to_RR1:

started after: 1.3686392 seconds

and ended after: 45.9747402 seconds

link_RRC1_to_RR2:

started after: 1.3788322 seconds

and ended after: 45.9803612 seconds

Session RR1_to_RRC2:

link_RR1_to_RRC2:

started after: 1.3679472 seconds

and ended after: 46.3523182 seconds

Session RR1_to_RRC3:

link_RR3_to_RR1:

started after: 1.3685622 seconds

and ended after: 46.5449332 seconds

link_RR3_to_RRC3:

started after: 1.3713972 seconds

and ended after: 46.5542692 seconds

Session RR2_to_BA3:

link_RR1_to_BA3:

started after: 1.3782432 seconds

and ended after: 35.2559312 seconds

link_RR2_to_RR1:

started after: 1.3660422 seconds

and ended after: 35.2505012 seconds

Session RR2_to_CE4:

link_RR2_to_RR3:

started after: 1.3683092 seconds

and ended after: 39.3804652 seconds

link_RR3_to_CE4:

started after: 1.3714732 seconds

and ended after: 39.3813872 seconds

Session RR2_to_RR1:

link_RR2_to_RR1:

started after: 1.3795682 seconds

and ended after: 34.7413382 seconds

Session RR2_to_RR3:

link_RR2_to_RR3:

started after: 1.3808172 seconds

and ended after: 28.7440662 seconds

B.1. TIME OF FIRST AND LAST BGP UPDATE MESSAGES IN GNS3 WITHOUT FLOWSVI

Session RR2_to_RRC1:

link_RRC1_to_RR2:

started after: 1.3677372 seconds

and ended after: 36.4128052 seconds

Session RR2_to_RRC2:

link_RR1_to_RRC2:

started after: 1.3680362 seconds

and ended after: 41.6969342 seconds

link_RR2_to_RR1:

started after: 1.3659242 seconds

and ended after: 41.6960272 seconds

Session RR2_to_RRC3:

link_RR2_to_RR3:

started after: 1.3678932 seconds

and ended after: 38.3470332 seconds

link_RR3_to_RRC3:

started after: 1.3713322 seconds

and ended after: 38.3503082 seconds

Session RR3_to_BA3:

link_RR1_to_BA3:

started after: 1.4090112 seconds

and ended after: 35.3896602 seconds

link_RR3_to_RR1:

started after: 1.4023192 seconds

and ended after: 35.3861582 seconds

Session RR3_to_CE4:

link_RR3_to_CE4:

started after: 1.4023612 seconds

and ended after: 35.5299522 seconds

Session RR3_to_RR1:

link_RR3_to_RR1:

started after: 1.3918972 seconds

and ended after: 34.5697962 seconds

Session RR3_to_RR2:

link_RR2_to_RR3:

started after: 1.3923112 seconds

and ended after: 28.7073952 seconds

Session RR3_to_RRC1:

link_RR2_to_RR3:

started after: 1.4026272 seconds

and ended after: 36.4173592 seconds

link_RRC1_to_RR2:

started after: 1.4118742 seconds

and ended after: 36.4235062 seconds

Session RR3_to_RRC2:

link_RR1_to_RRC2:

started after: 1.3987652 seconds

and ended after: 39.3585812 seconds

```

link_RR3_to_RR1:
    started after: 1.3919582 seconds
    and ended after: 39.3504372 seconds
Session RR3_to_RRC3:
    link_RR3_to_RRC3:
        started after: 1.3920312 seconds
        and ended after: 35.4376022 seconds

```

B.2 Time of First and Last BGP UPDATE Messages in GNS3 with Flows

```

The state of the link changed at:
    epoch time: 2021-10-21 16:12:20.186601
Session BA3_to_RR1:
    link_RR1_to_BA3:
        started after: 9.7240276 seconds
        and ended after: 34.6025796 seconds
Session BA3_to_RR2:
    link_RR1_to_BA3:
        started after: 9.7137186 seconds
        and ended after: 29.3011866 seconds
    link_RR2_to_RR1:
        started after: 9.7159666 seconds
        and ended after: 29.3045776 seconds
Session BA3_to_RR3:
    link_RR1_to_BA3:
        started after: 9.7138366 seconds
        and ended after: 28.5453006 seconds
    link_RR3_to_RR1:
        started after: 9.7164546 seconds
        and ended after: 28.5481296 seconds
Session RR1_to_BA3:
    link_RR1_to_BA3:
        started after: 10.0129116 seconds
        and ended after: 38.9087616 seconds
Session RR1_to_CE4:
    link_RR3_to_CE4:
        started after: 10.0077616 seconds
        and ended after: 51.1960036 seconds
    link_RR3_to_RR1:
        started after: 10.0013996 seconds
        and ended after: 51.1894926 seconds
Session RR1_to_RR2:
    link_RR2_to_RR1:
        started after: 9.9775776 seconds
        and ended after: 38.1001626 seconds

```

Session RR1_to_RR3:

link_RR3_to_RR1:

started after: 9.9897056 seconds

and ended after: 38.4521236 seconds

Session RR1_to_RRC1:

link_RR2_to_RR1:

started after: 9.9890676 seconds

and ended after: 50.7406956 seconds

link_RRC1_to_RR2:

started after: 9.9910886 seconds

and ended after: 50.7420516 seconds

Session RR1_to_RRC2:

link_RR1_to_RRC2:

started after: 9.9909396 seconds

and ended after: 49.2286296 seconds

Session RR1_to_RRC3:

link_RR3_to_RR1:

started after: 10.0012226 seconds

and ended after: 50.6379926 seconds

link_RR3_to_RRC3:

started after: 10.0076156 seconds

and ended after: 50.6409196 seconds

Session RR2_to_BA3:

link_RR1_to_BA3:

started after: 10.0777106 seconds

and ended after: 38.5457966 seconds

link_RR2_to_RR1:

started after: 10.0726446 seconds

and ended after: 38.5373696 seconds

Session RR2_to_CE4:

link_RR2_to_RR3:

started after: 10.0727336 seconds

and ended after: 38.7504886 seconds

link_RR3_to_CE4:

started after: 10.0791236 seconds

and ended after: 38.7769556 seconds

Session RR2_to_RR1:

link_RR2_to_RR1:

started after: 10.0521146 seconds

and ended after: 37.6438176 seconds

Session RR2_to_RR3:

link_RR2_to_RR3:

started after: 10.0521966 seconds

and ended after: 31.0095836 seconds

Session RR2_to_RRC1:

link_RRC1_to_RR2:

started after: 10.0524586 seconds

and ended after: 38.6086346 seconds

Session RR2_to_RRC2:

link_RR1_to_RRC2:

started after: 10.0672526 seconds

and ended after: 38.8778006 seconds

link_RR2_to_RR1:

started after: 10.0623996 seconds

and ended after: 38.8722756 seconds

Session RR2_to_RRC3:

link_RR2_to_RR3:

started after: 10.0624836 seconds

and ended after: 38.8416246 seconds

link_RR3_to_RRC3:

started after: 10.0688986 seconds

and ended after: 38.8493536 seconds

Session RR3_to_BA3:

link_RR1_to_BA3:

started after: 9.9020066 seconds

and ended after: 38.7535366 seconds

link_RR3_to_RR1:

started after: 9.8863246 seconds

and ended after: 38.7458776 seconds

Session RR3_to_CE4:

link_RR3_to_CE4:

started after: 9.8864466 seconds

and ended after: 38.9529906 seconds

Session RR3_to_RR1:

link_RR3_to_RR1:

started after: 9.8863606 seconds

and ended after: 37.5859956 seconds

Session RR3_to_RR2:

link_RR2_to_RR3:

started after: 9.8867846 seconds

and ended after: 30.9769966 seconds

Session RR3_to_RRC1:

link_RR2_to_RR3:

started after: 9.8868136 seconds

and ended after: 38.8803536 seconds

link_RRC1_to_RR2:

started after: 9.8889656 seconds

and ended after: 38.8826536 seconds

Session RR3_to_RRC2:

link_RR1_to_RRC2:

started after: 9.9028676 seconds

and ended after: 38.8881396 seconds

link_RR3_to_RR1:

started after: 9.8965786 seconds

and ended after: 38.8804266 seconds

Session RR3_to_RRC3:

link_RR3_to_RR3:
started after: 9.8863986 seconds
and ended after: 38.8492706 seconds

B.3 Time of First and Last BGP UPDATE Messages in GNS3 with a Slow Route-Reflector-Client

The state of the link changed at:
epoch time: 2021-11-04 14:55:55.089706
Session BA3_to_RR1:
link_RR1_to_BA3:
started after: 1.768134 seconds
and ended after: 27.772251 seconds
Session BA3_to_RR2:
link_RR1_to_BA3:
started after: 1.210196 seconds
and ended after: 27.741970 seconds
link_RR2_to_RR1:
started after: 1.221760 seconds
and ended after: 27.745932 seconds
Session BA3_to_RR3:
link_RR1_to_BA3:
started after: 1.220612 seconds
and ended after: 27.459191 seconds
link_RR3_to_RR1:
started after: 1.221864 seconds
and ended after: 27.468442 seconds
Session RR1_to_BA3:
link_RR1_to_BA3:
started after: 1.848849 seconds
and ended after: 34.305203 seconds
Session RR1_to_CE4:
link_RR3_to_CE4:
started after: 1.850408 seconds
and ended after: 45.302058 seconds
link_RR3_to_RR1:
started after: 1.849288 seconds
and ended after: 45.297429 seconds
Session RR1_to_RR2:
link_RR2_to_RR1:
started after: 1.849430 seconds
and ended after: 33.861051 seconds
Session RR1_to_RR3:
link_RR3_to_RR1:
started after: 1.849314 seconds
and ended after: 33.883119 seconds

B.3. TIME OF FIRST AND LAST BGP UPDATE MESSAGES IN GNS3 WITH A SLOW ROUTE-REFLECTOR

Session RR1_to_RRC1:

link_RR2_to_RR1:

started after: 1.849401 seconds

and ended after: 159.985654 seconds

link_RRC1_to_RR2:

started after: 2.471795 seconds

and ended after: 159.970205 seconds

Session RR1_to_RRC2:

link_RR1_to_RRC2:

started after: 1.848896 seconds

and ended after: 46.257028 seconds

Session RR1_to_RRC3:

link_RR3_to_RR1:

started after: 1.849260 seconds

and ended after: 44.648821 seconds

link_RR3_to_RRC3:

started after: 1.850349 seconds

and ended after: 44.651616 seconds

Session RR2_to_BA3:

link_RR1_to_BA3:

started after: 1.545268 seconds

and ended after: 34.513281 seconds

link_RR2_to_RR1:

started after: 1.538720 seconds

and ended after: 34.512649 seconds

Session RR2_to_CE4:

link_RR2_to_RR3:

started after: 1.539674 seconds

and ended after: 42.465108 seconds

link_RR3_to_CE4:

started after: 1.546630 seconds

and ended after: 42.473701 seconds

Session RR2_to_RR1:

link_RR2_to_RR1:

started after: 1.516678 seconds

and ended after: 33.135718 seconds

Session RR2_to_RR3:

link_RR2_to_RR3:

started after: 1.527805 seconds

and ended after: 29.706547 seconds

Session RR2_to_RRC1:

link_RRC1_to_RR2:

started after: 2.791662 seconds

and ended after: 303.712675 seconds

Session RR2_to_RRC2:

link_RR1_to_RRC2:

started after: 1.535073 seconds

and ended after: 45.378448 seconds

B.3. TIME OF FIRST AND LAST BGP UPDATE MESSAGES IN GNS3 WITH A SLOW ROUTE-REFLECTOR

link_RR2_to_RR1:
 started after: 1.527403 seconds
 and ended after: 45.372770 seconds

Session RR2_to_RRC3:
 link_RR2_to_RR3:
 started after: 1.539196 seconds
 and ended after: 43.000275 seconds
 link_RR3_to_RRC3:
 started after: 1.546502 seconds
 and ended after: 43.008583 seconds

Session RR3_to_BA3:
 link_RR1_to_BA3:
 started after: 1.524880 seconds
 and ended after: 34.662908 seconds
 link_RR3_to_RR1:
 started after: 1.515876 seconds
 and ended after: 34.659530 seconds

Session RR3_to_CE4:
 link_RR3_to_CE4:
 started after: 1.505504 seconds
 and ended after: 41.487470 seconds

Session RR3_to_RR1:
 link_RR3_to_RR1:
 started after: 1.515976 seconds
 and ended after: 32.991289 seconds

Session RR3_to_RR2:
 link_RR2_to_RR3:
 started after: 1.516295 seconds
 and ended after: 29.669228 seconds

Session RR3_to_RRC1:
 link_RR2_to_RR3:
 started after: 1.718977 seconds
 and ended after: 203.651576 seconds
 link_RRC1_to_RR2:
 started after: 1.723427 seconds
 and ended after: 204.663149 seconds

Session RR3_to_RRC2:
 link_RR1_to_RRC2:
 started after: 1.514549 seconds
 and ended after: 45.145870 seconds
 link_RR3_to_RR1:
 started after: 1.505261 seconds
 and ended after: 45.139343 seconds

Session RR3_to_RRC3:
 link_RR3_to_RRC3:
 started after: 1.505372 seconds
 and ended after: 42.586282 seconds

Appendix C

Flow Graphs

C.1 GNS3

C.1.1 Original Setup per Prefix Graphs

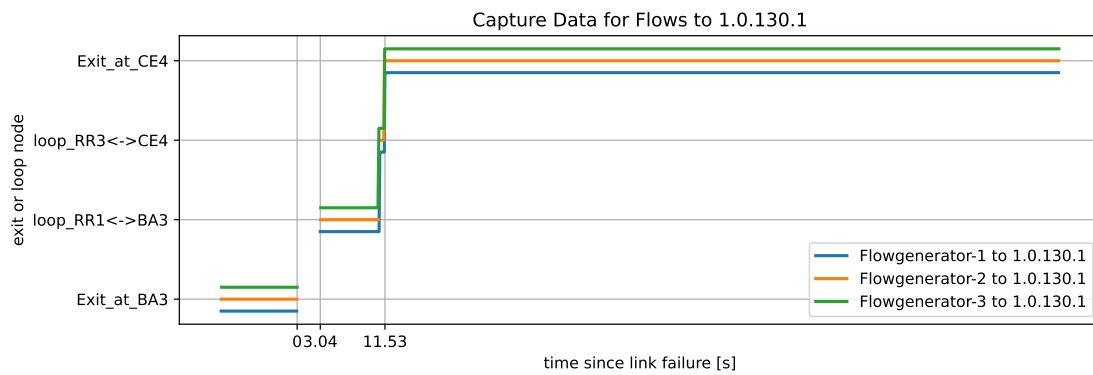


Figure C.1: Capture data for a link failure in GNS3 showing the convergence for 1.0.130.1 from different injection points

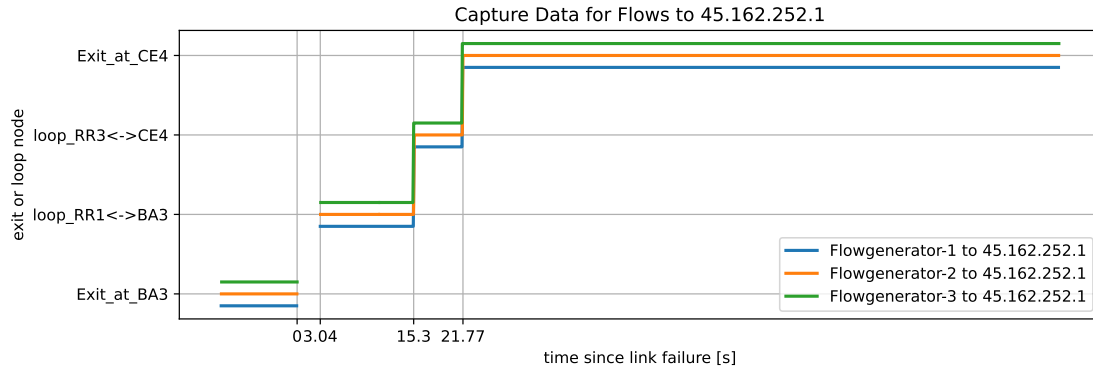


Figure C.2: Capture data for a link failure in GNS3 showing the convergence for 45.162.252.1 from different injection points

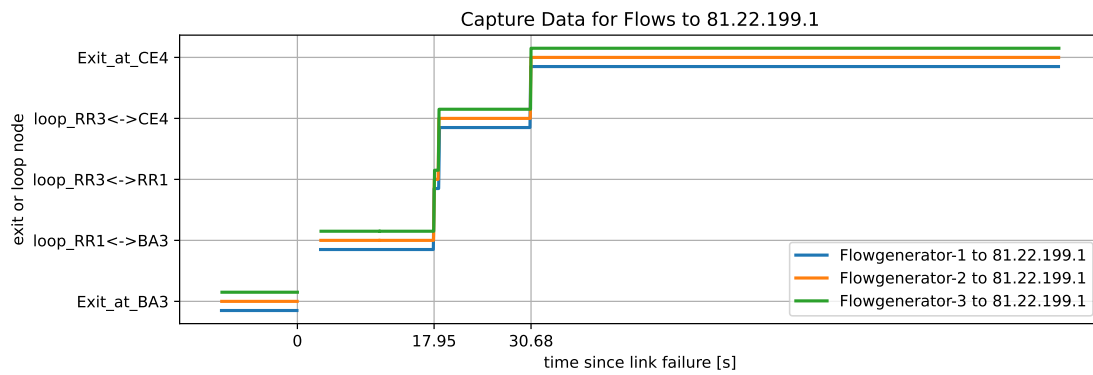


Figure C.3: Capture data for a link failure in GNS3 showing the convergence for 81.22.199.1 from different injection points

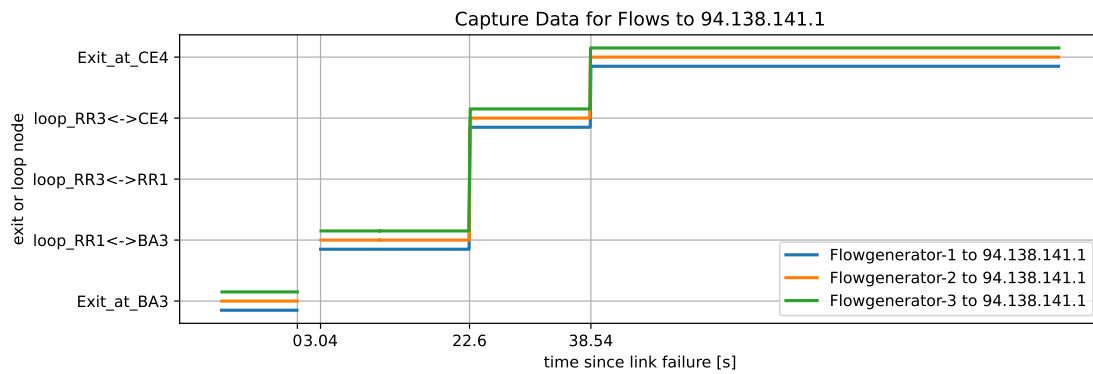


Figure C.4: Capture data for a link failure in GNS3 showing the convergence for 94.138.141.1 from different injection points

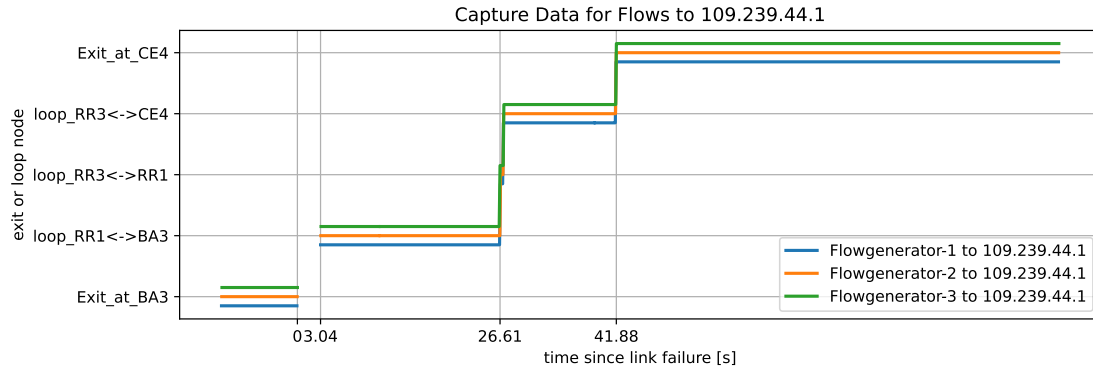


Figure C.5: Capture data for a link failure in GNS3 showing the convergence for 109.239.44.1 from different injection points

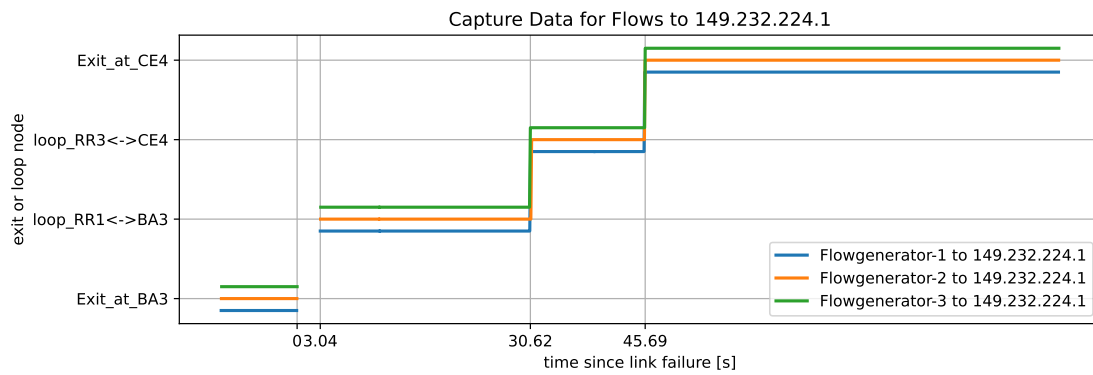


Figure C.6: Capture data for a link failure in GNS3 showing the convergence for 149.232.224.1 from different injection points

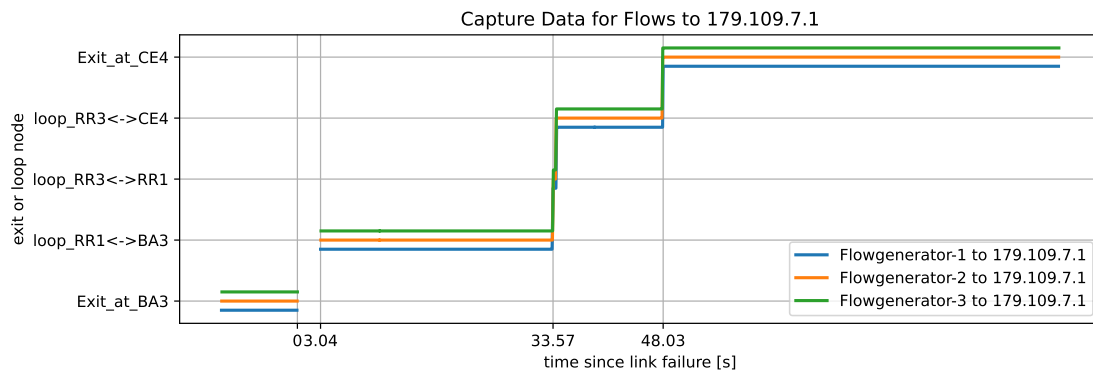


Figure C.7: Capture data for a link failure in GNS3 showing the convergence for 179.109.7.1 from different injection points

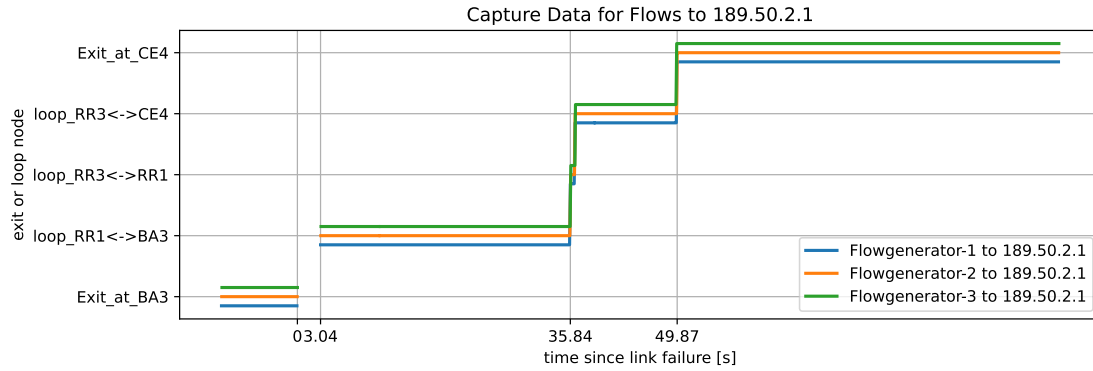


Figure C.8: Capture data for a link failure in GNS3 showing the convergence for 189.50.2.1 from different injection points

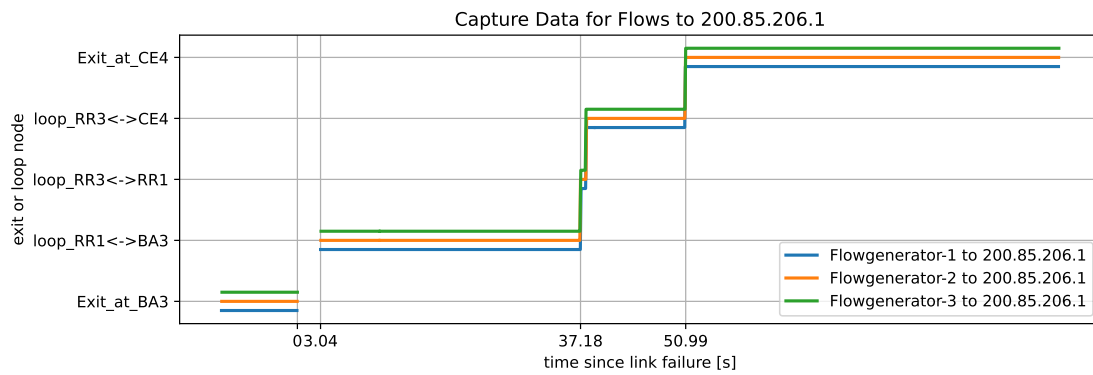


Figure C.9: Capture data for a link failure in GNS3 showing the convergence for 200.85.206.1 from different injection points

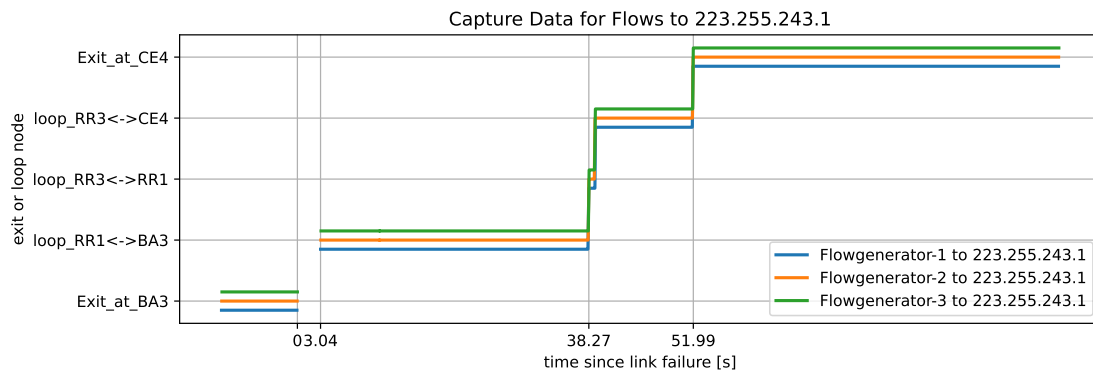


Figure C.10: Capture data for a link failure in GNS3 showing the convergence for 223.255.243.1 from different injection points

C.1.2 OSPF Distribution Setup per Prefix Graphs

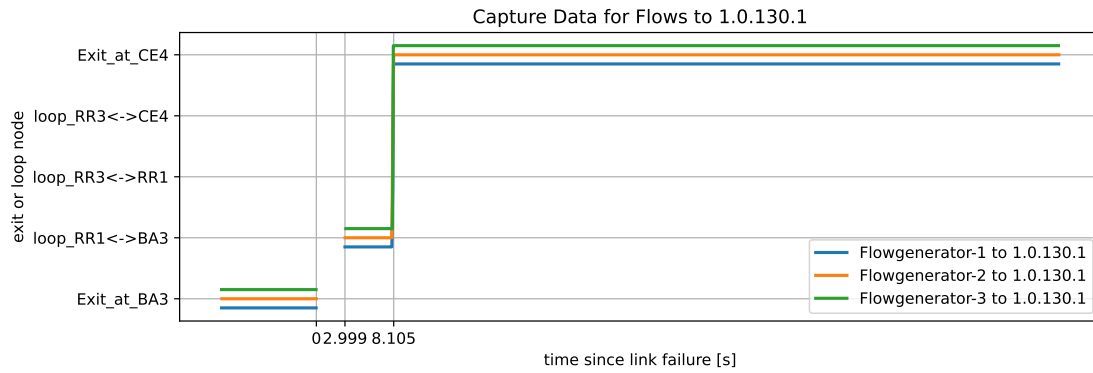


Figure C.11: Capture data for a link failure in GNS3 showing the convergence for 1.0.130.1 from different injection points

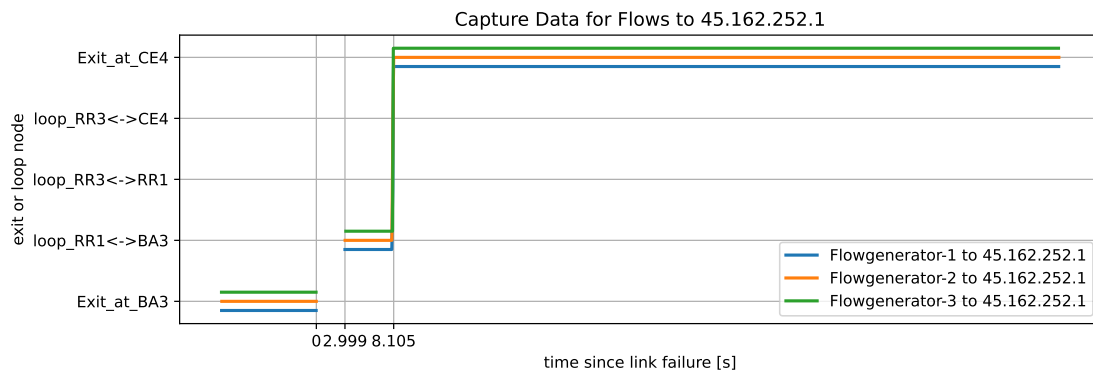


Figure C.12: Capture data for a link failure in GNS3 showing the convergence for 45.162.252.1 from different injection points

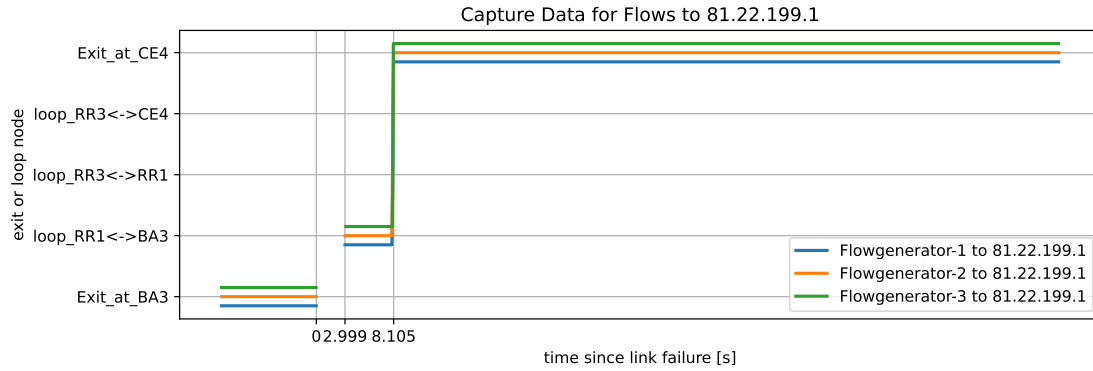


Figure C.13: Capture data for a link failure in GNS3 showing the convergence for 81.22.199.1 from different injection points

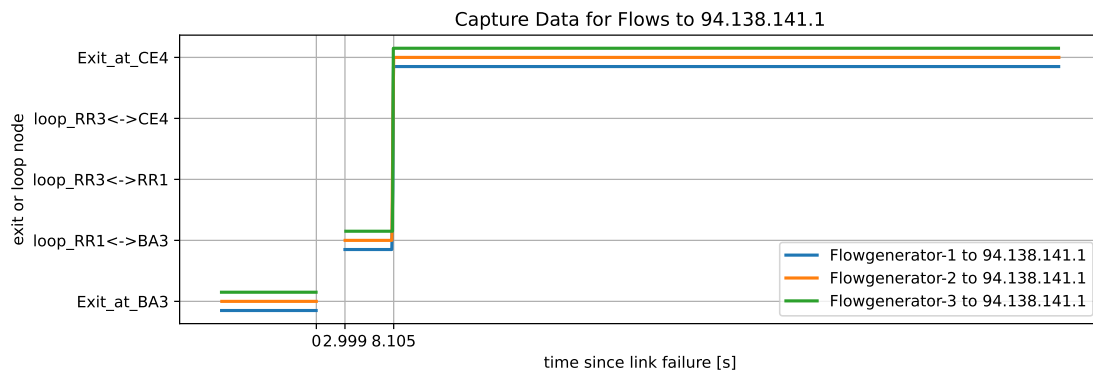


Figure C.14: Capture data for a link failure in GNS3 showing the convergence for 94.138.141.1 from different injection points

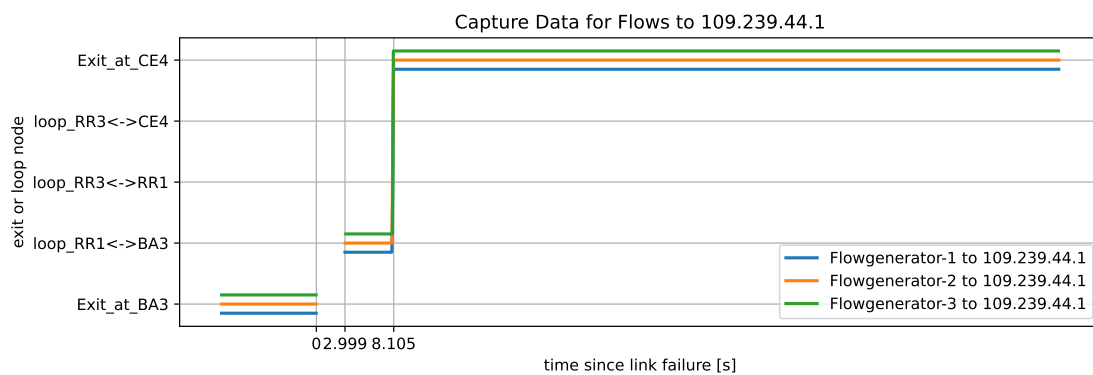


Figure C.15: Capture data for a link failure in GNS3 showing the convergence for 109.239.44.1 from different injection points

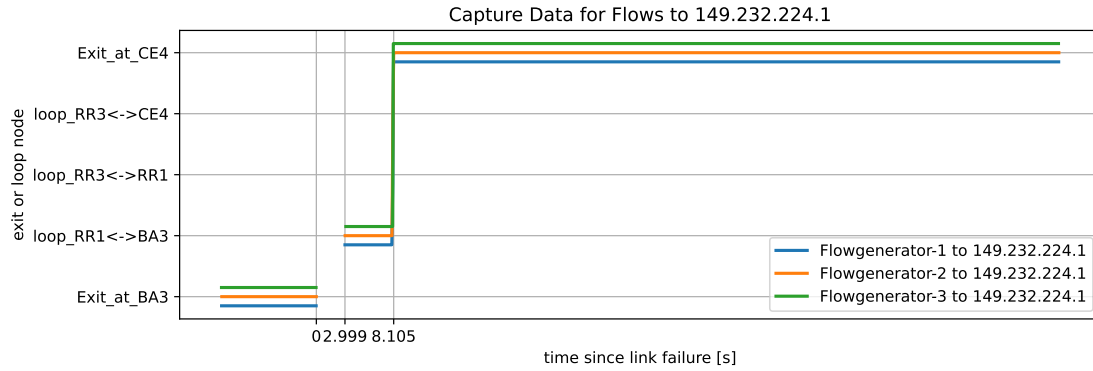


Figure C.16: Capture data for a link failure in GNS3 showing the convergence for 149.232.224.1 from different injection points

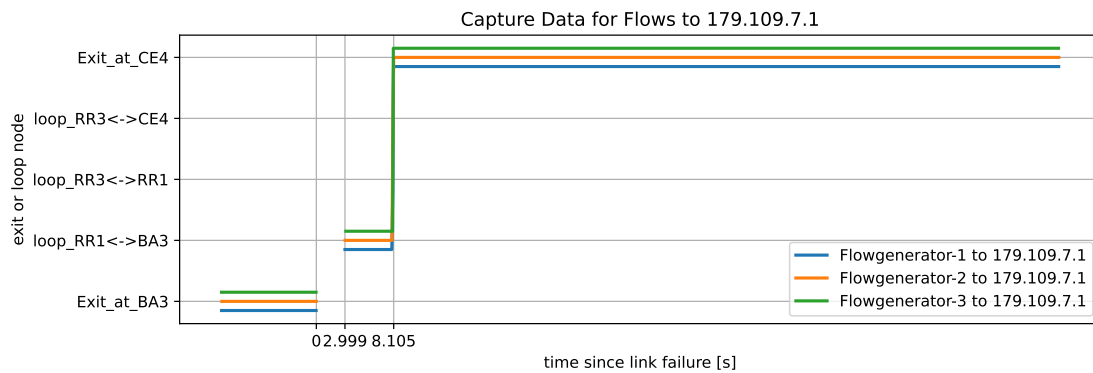


Figure C.17: Capture data for a link failure in GNS3 showing the convergence for 179.109.7.1 from different injection points

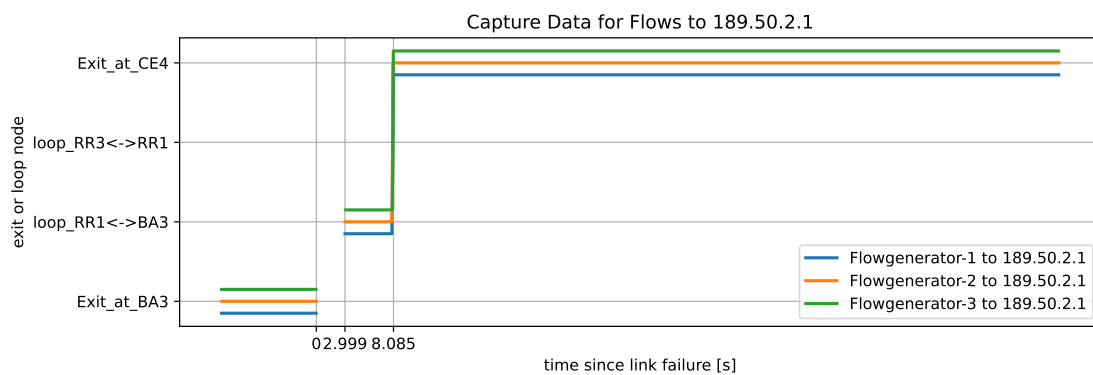


Figure C.18: Capture data for a link failure in GNS3 showing the convergence for 189.50.2.1 from different injection points

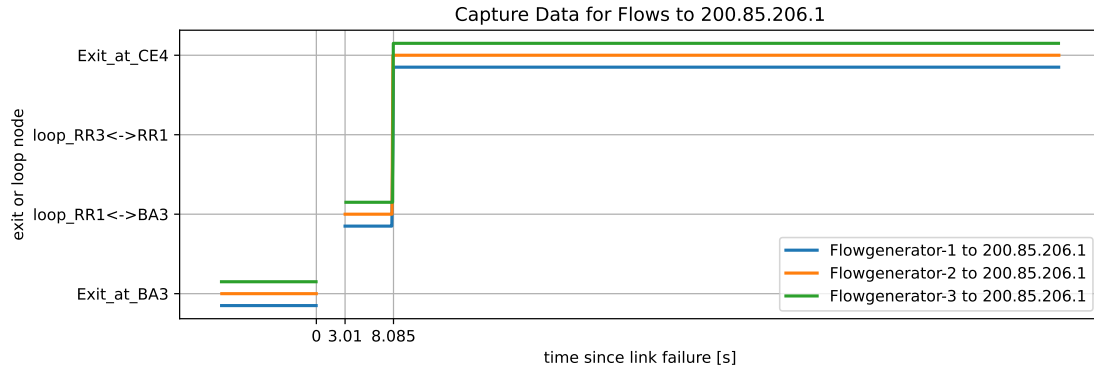


Figure C.19: Capture data for a link failure in GNS3 showing the convergence for 200.85.206.1 from different injection points

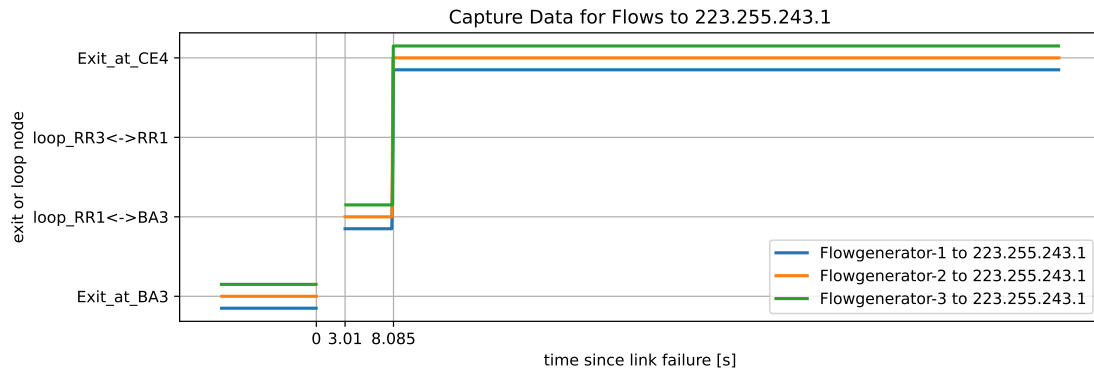


Figure C.20: Capture data for a link failure in GNS3 showing the convergence for 223.225.243.1 from different injection points

C.2 SWITCH Production Network

C.2.1 Original Setup, additional per Injection Point Graphs

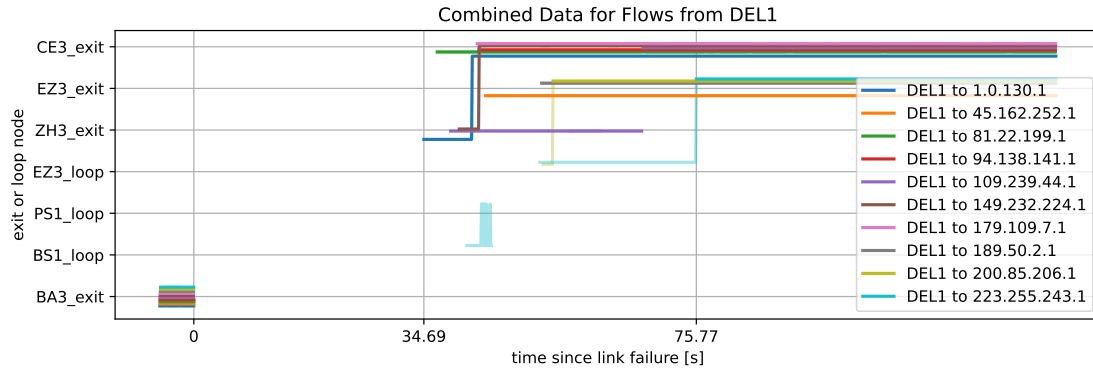


Figure C.21: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows from DEL1

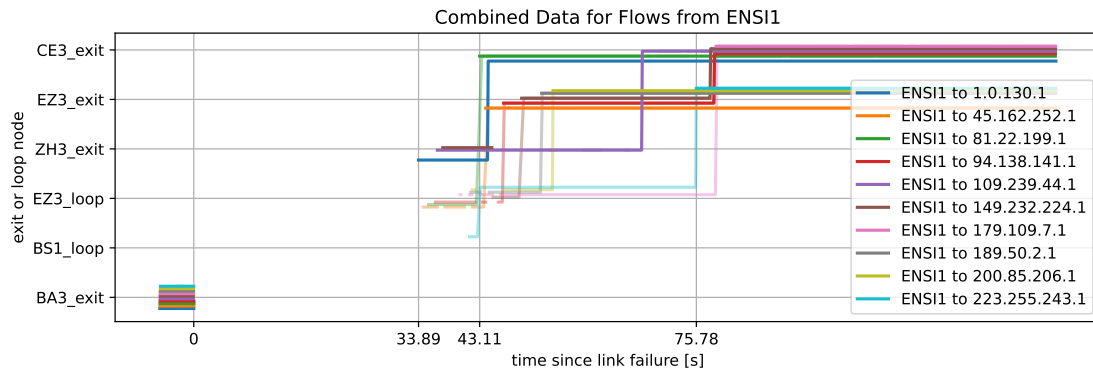


Figure C.22: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows from ENSI1

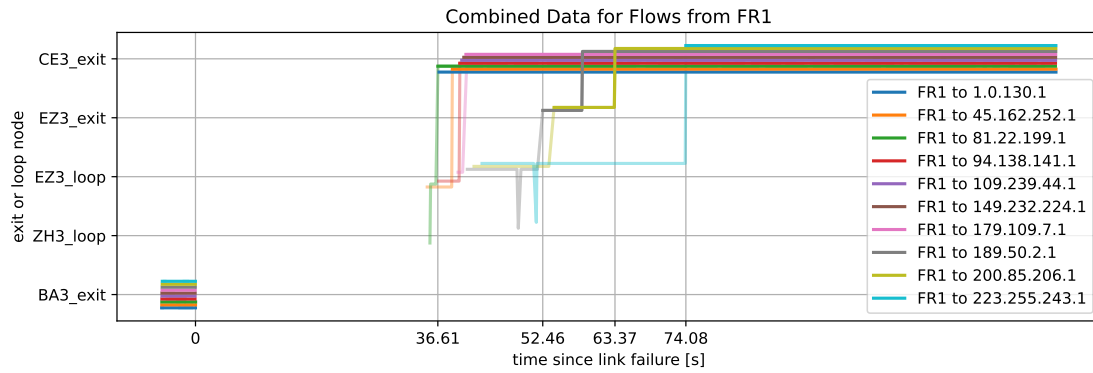


Figure C.23: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows from FR1

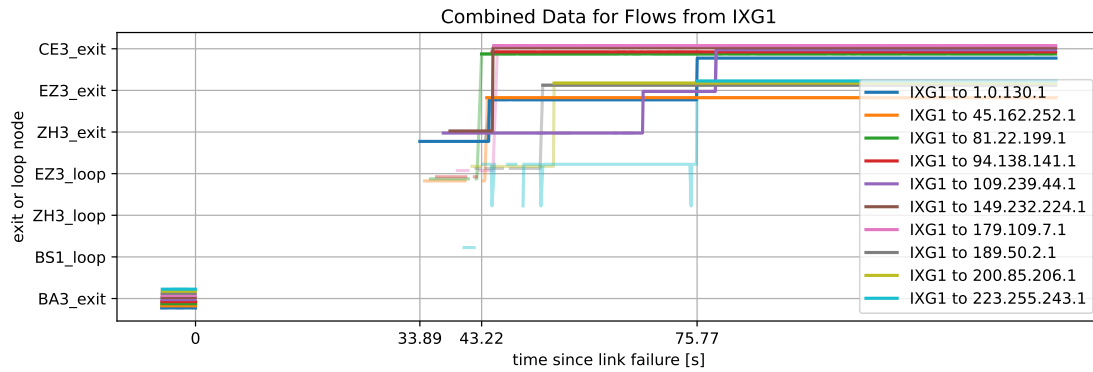


Figure C.24: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows from IXG1

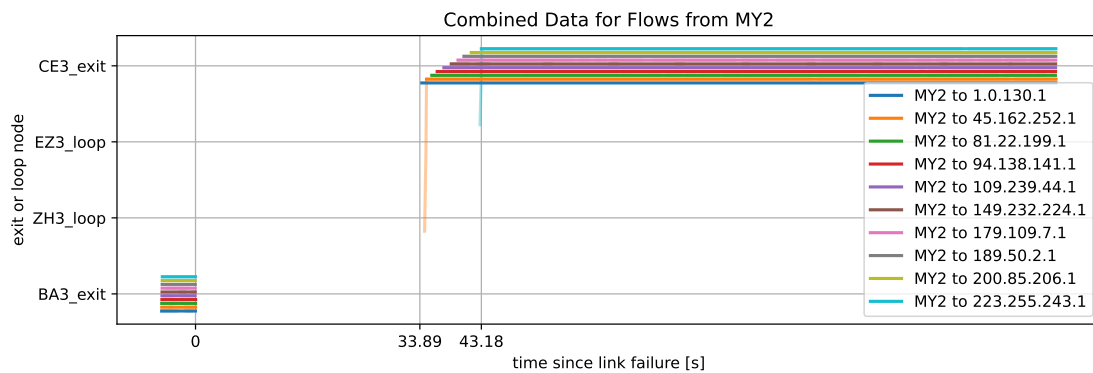


Figure C.25: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows from MY2

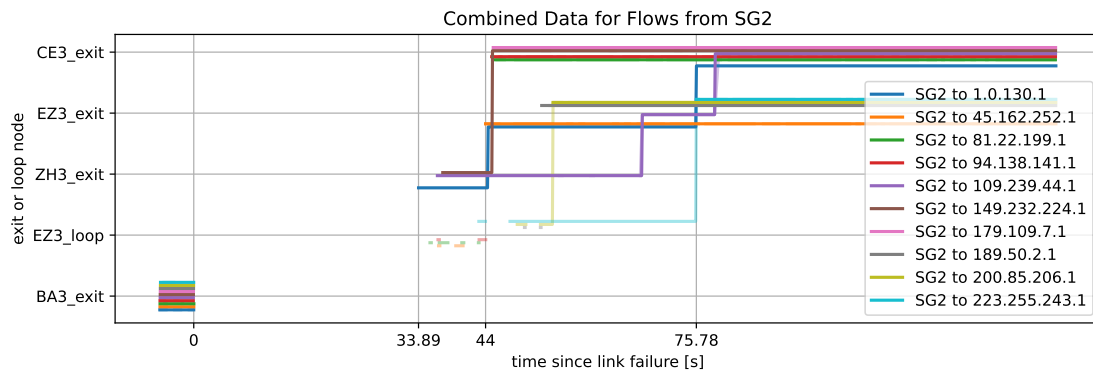


Figure C.26: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows from SG2

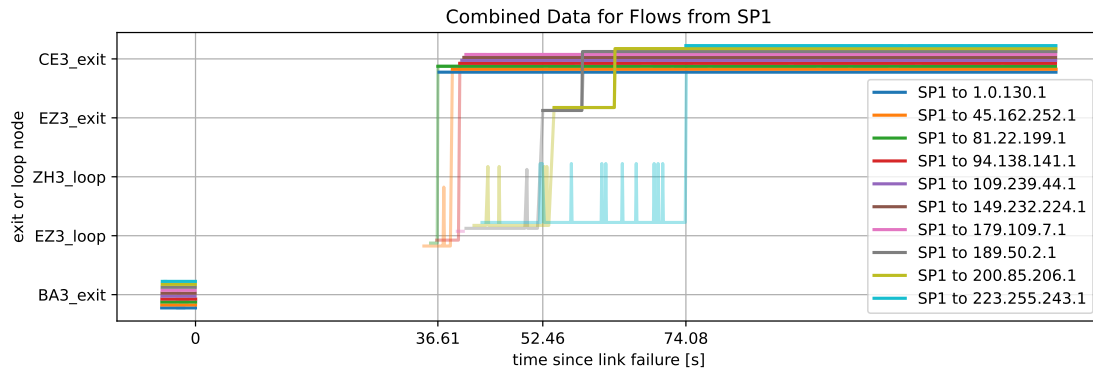


Figure C.27: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows from SP1

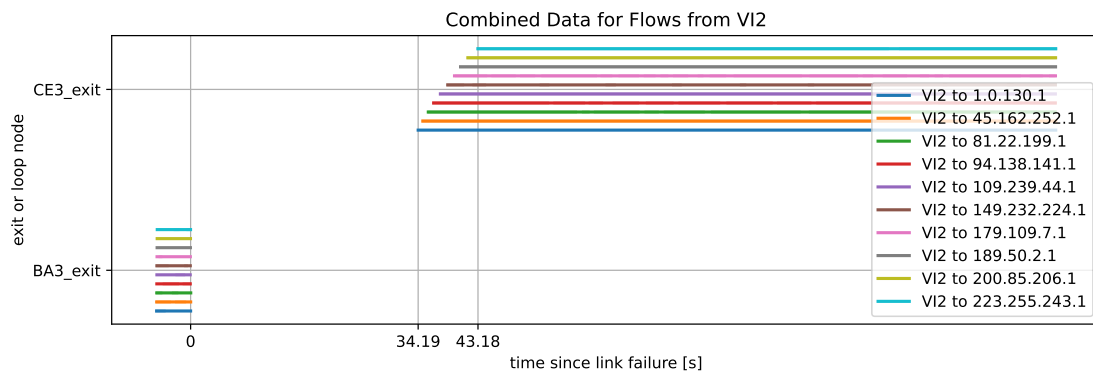


Figure C.28: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows from VI2

C.2.2 Original Setup, additional per Prefix Graphs

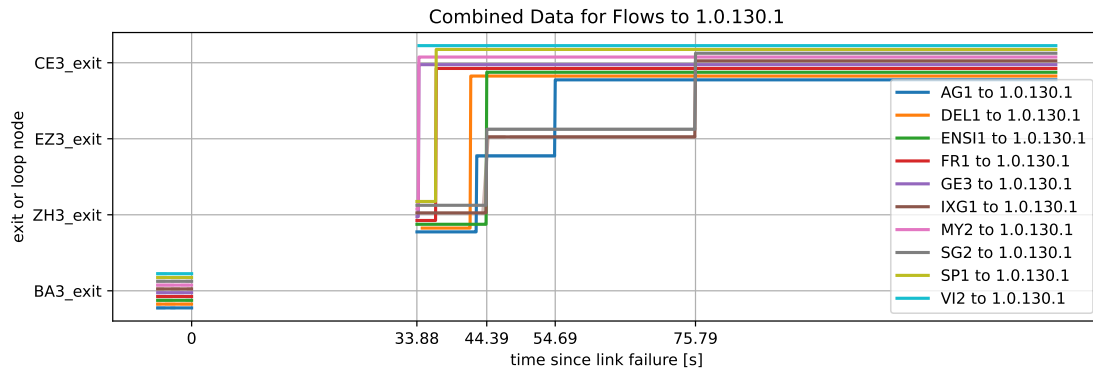


Figure C.29: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows towards 1.0.130.1

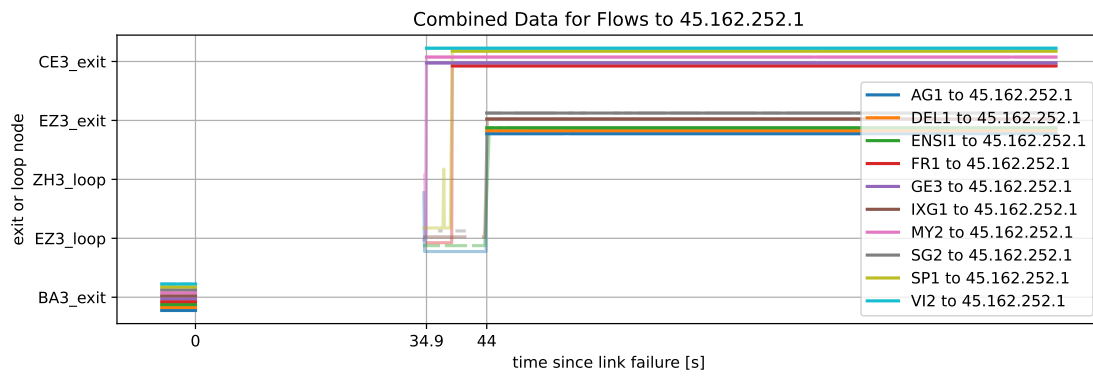


Figure C.30: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows towards 45.162.252.1

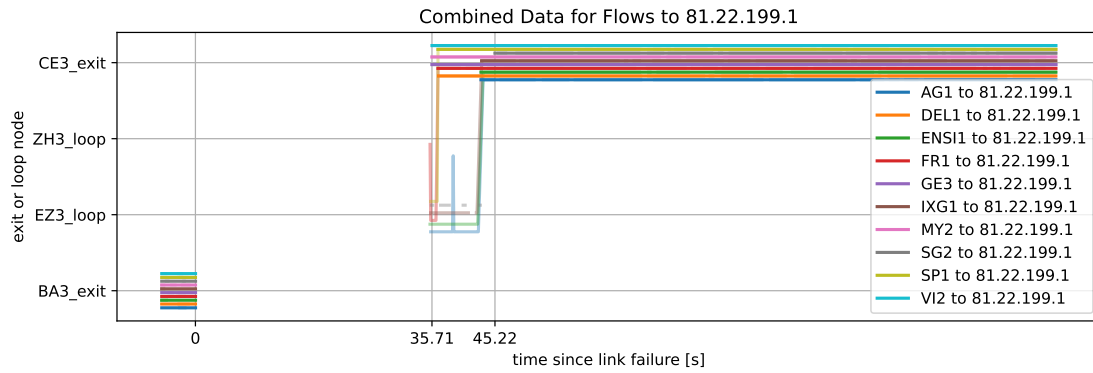


Figure C.31: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows towards 81.22.199.1

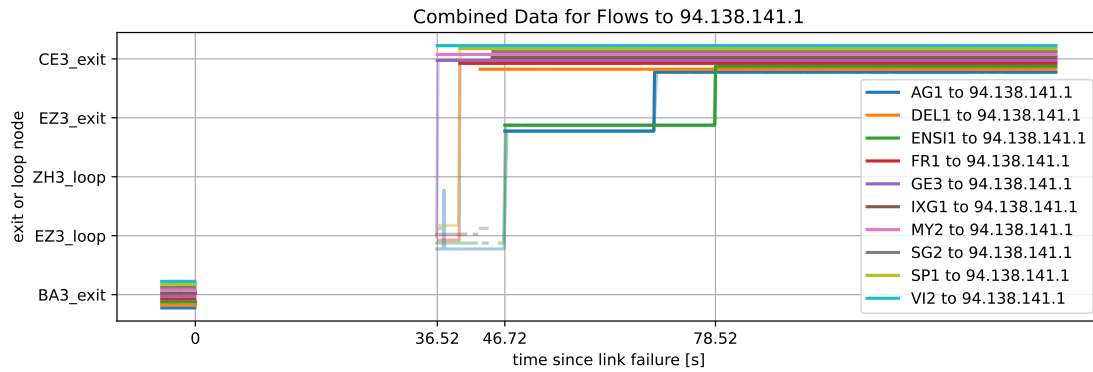


Figure C.32: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows towards 94.138.141.1

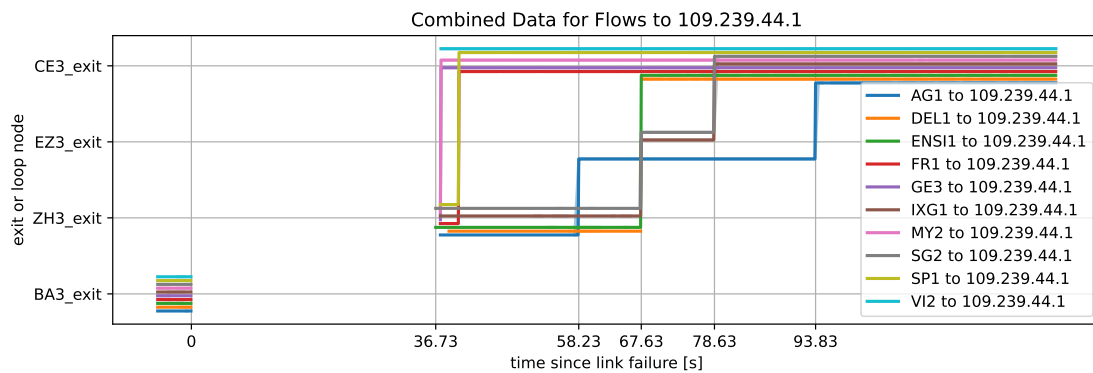


Figure C.33: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows towards 109.239.44.1

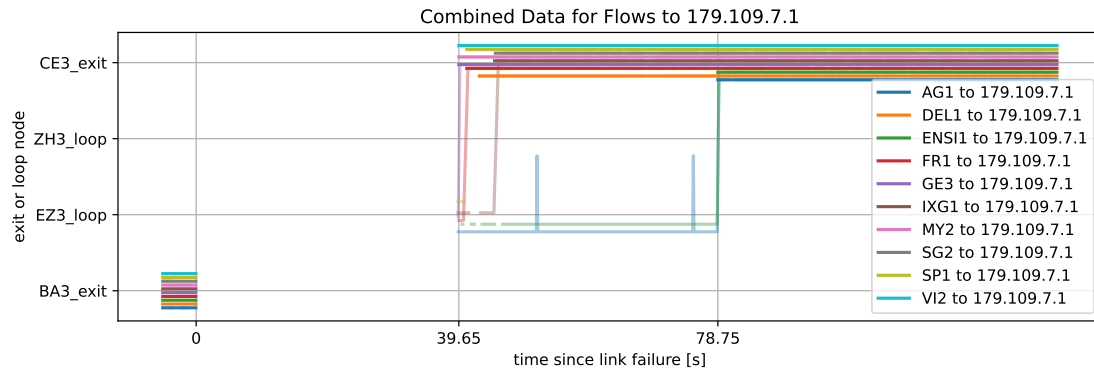


Figure C.34: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows towards 179.109.7.1

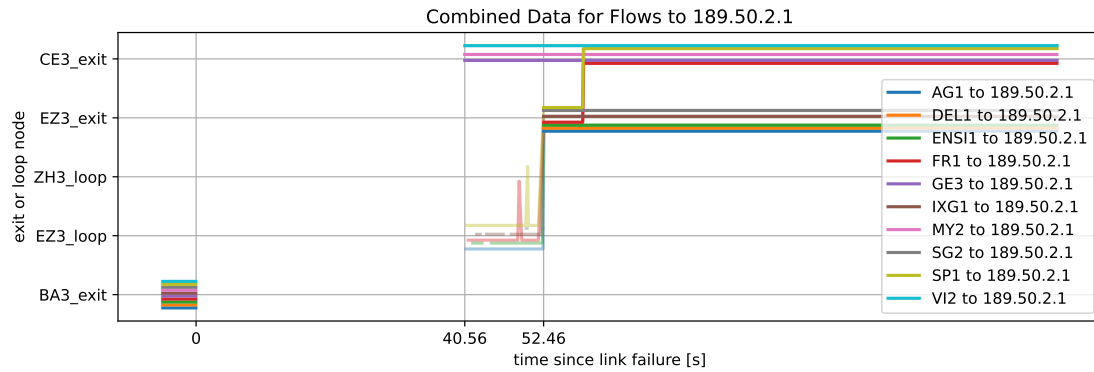


Figure C.35: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows towards 189.50.2.1

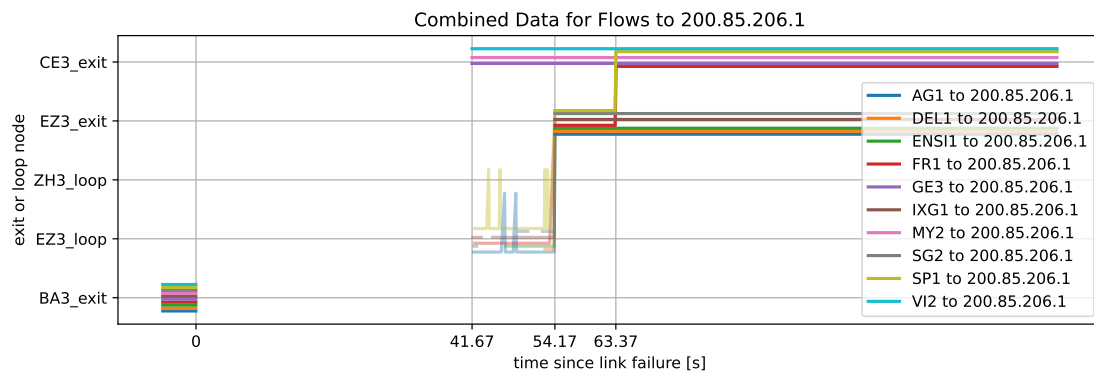


Figure C.36: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows towards 200.85.206.1

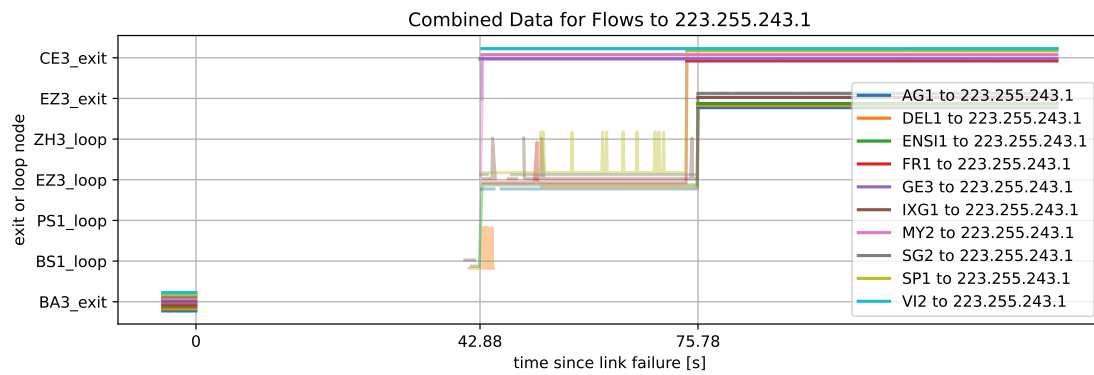


Figure C.37: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows towards 223.255.243.1

C.2.3 OSPF Distribution, additional per Injection Point Graphs

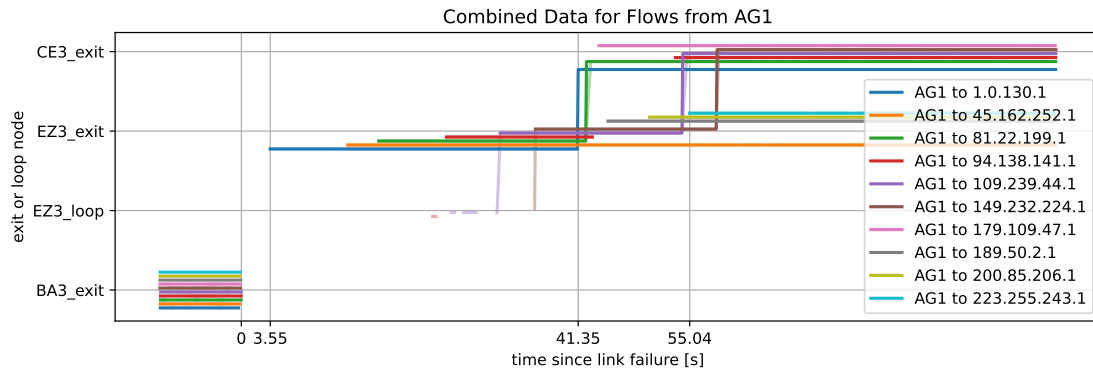


Figure C.38: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows from AG1

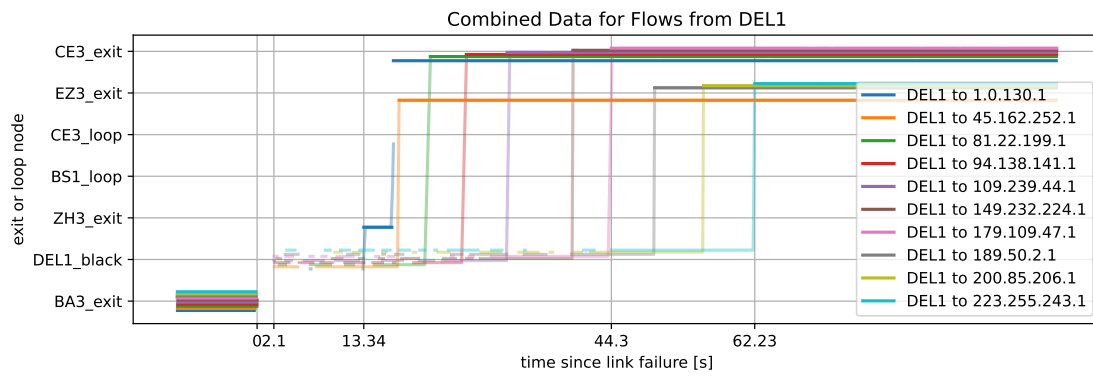


Figure C.39: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows from DEL1

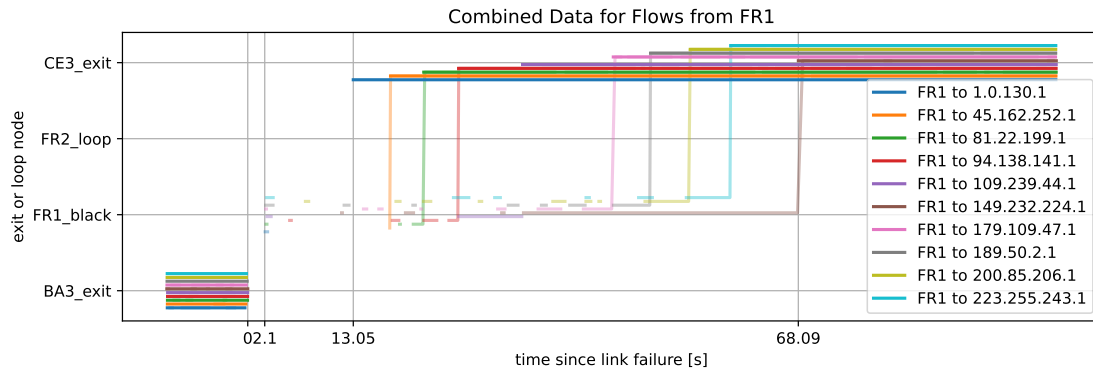


Figure C.40: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows from FR1

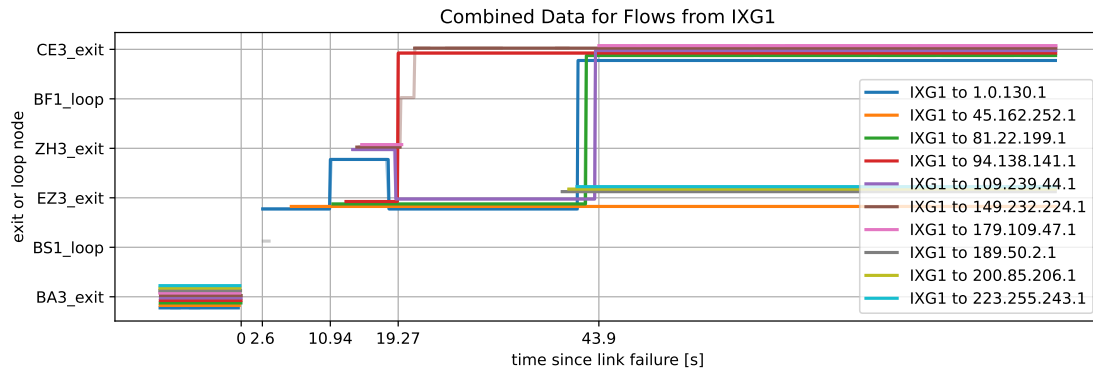


Figure C.41: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows from IXG1

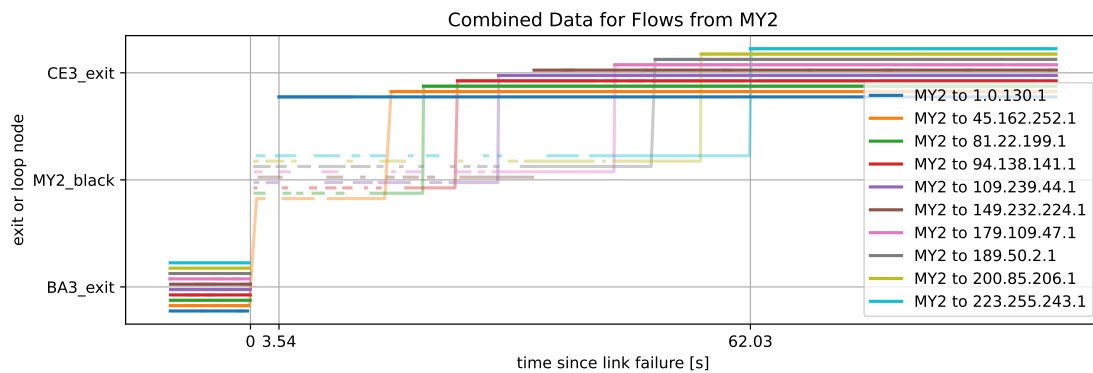


Figure C.42: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows from MY2

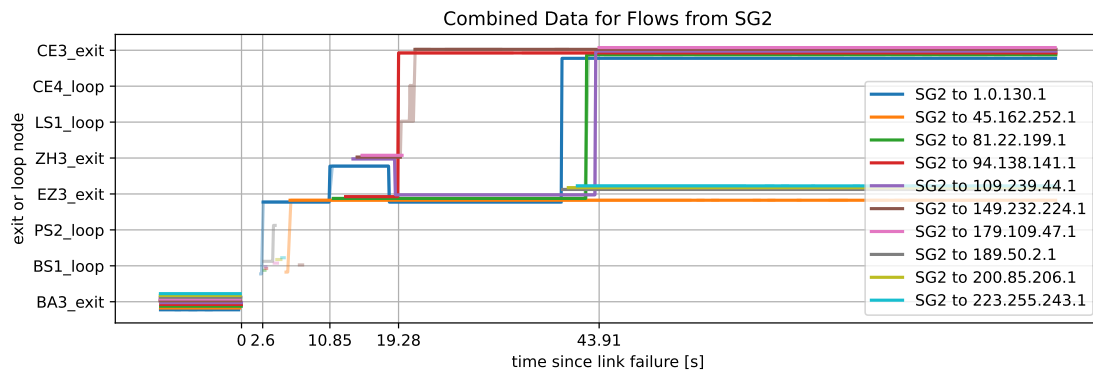


Figure C.43: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows from SG2

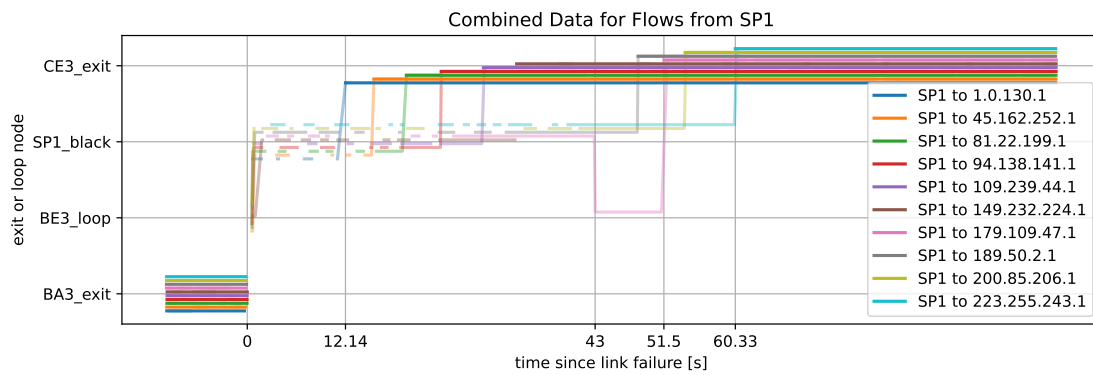


Figure C.44: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows from SP1

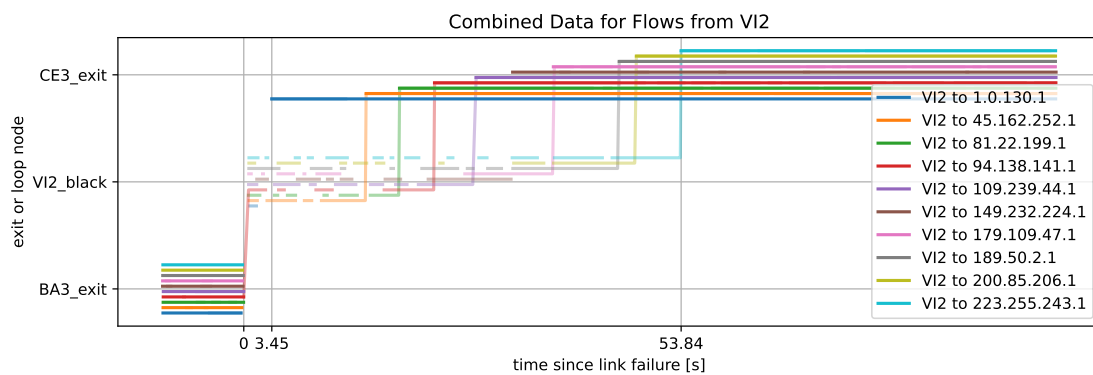


Figure C.45: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows from VI2

C.2.4 OSPF Distribution, per Prefix Graphs

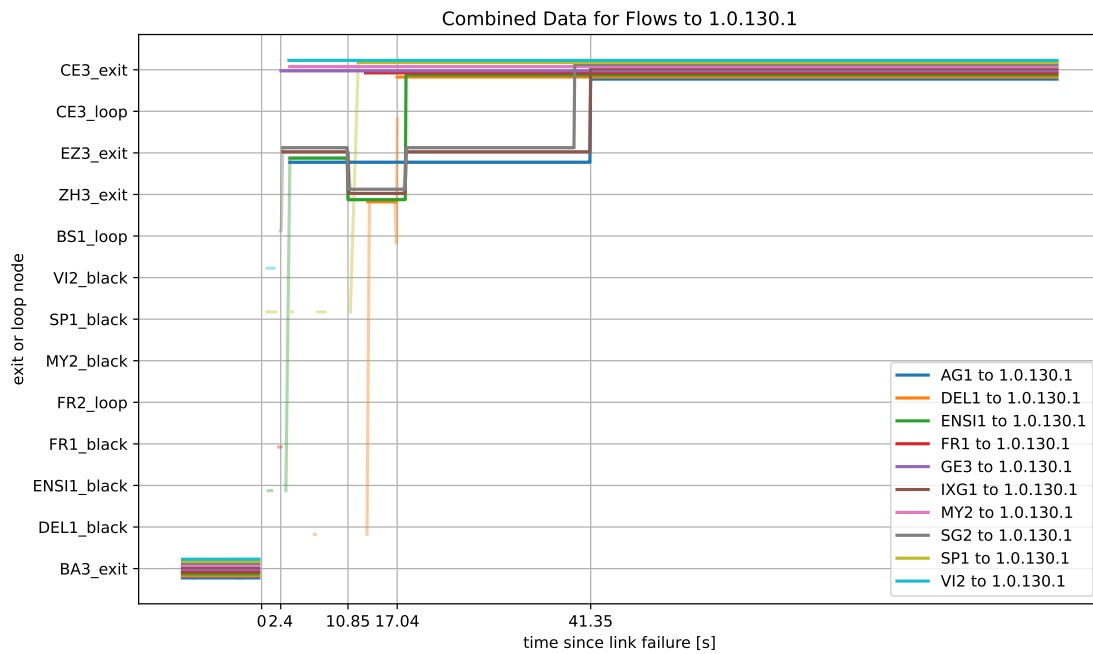


Figure C.46: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows towards 1.0.130.1

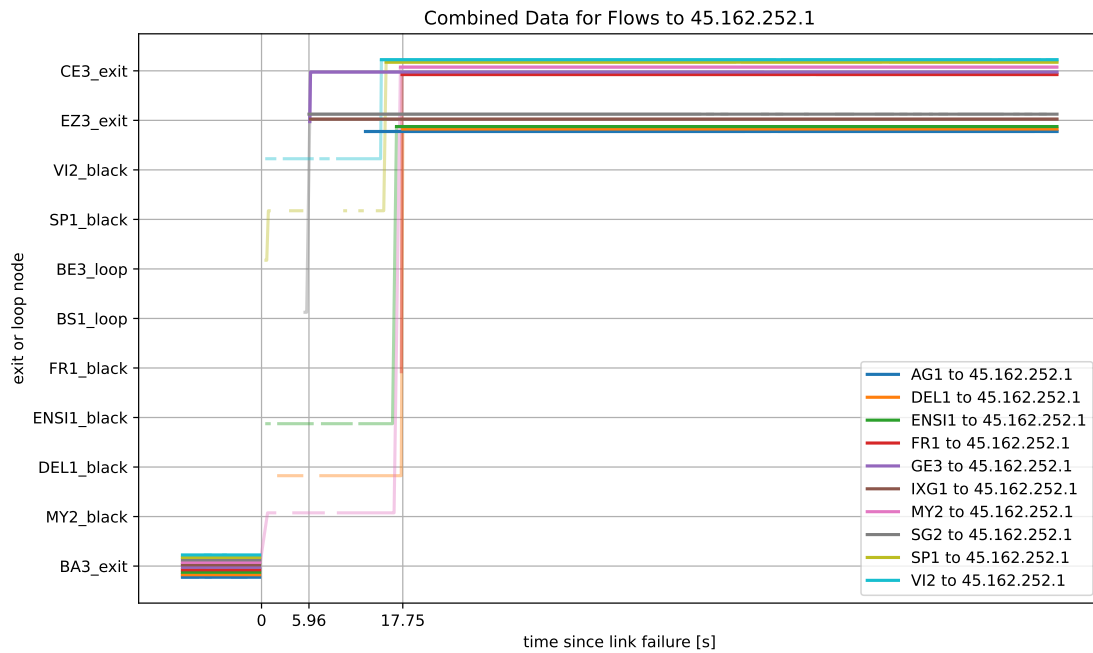


Figure C.47: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows towards 45.162.252.1

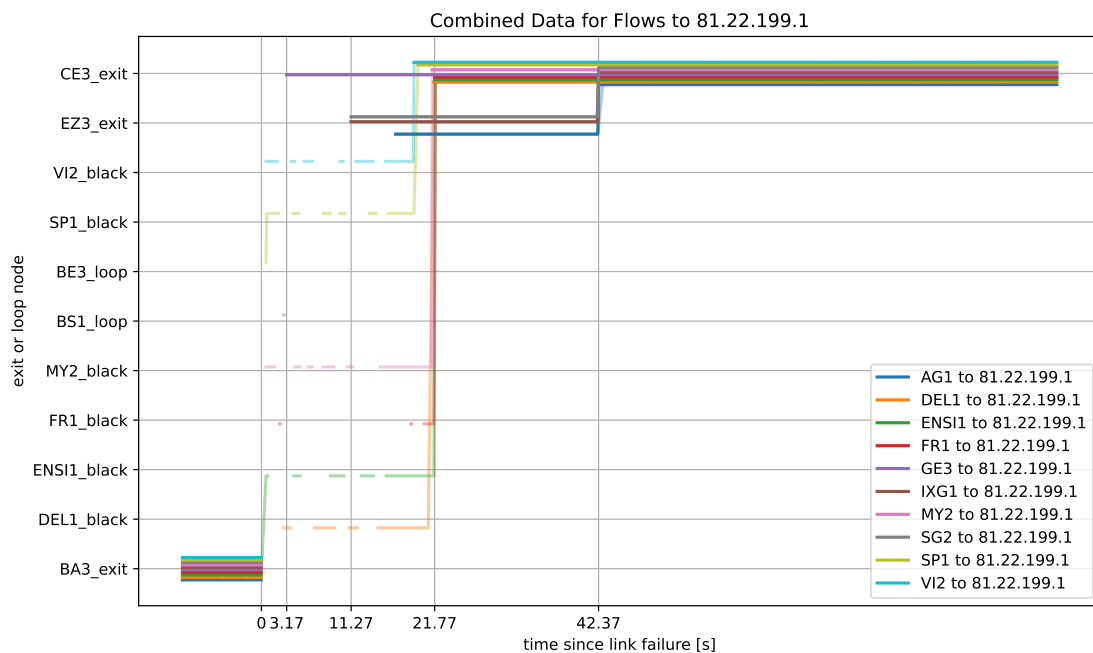


Figure C.48: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows towards 81.22.199.1

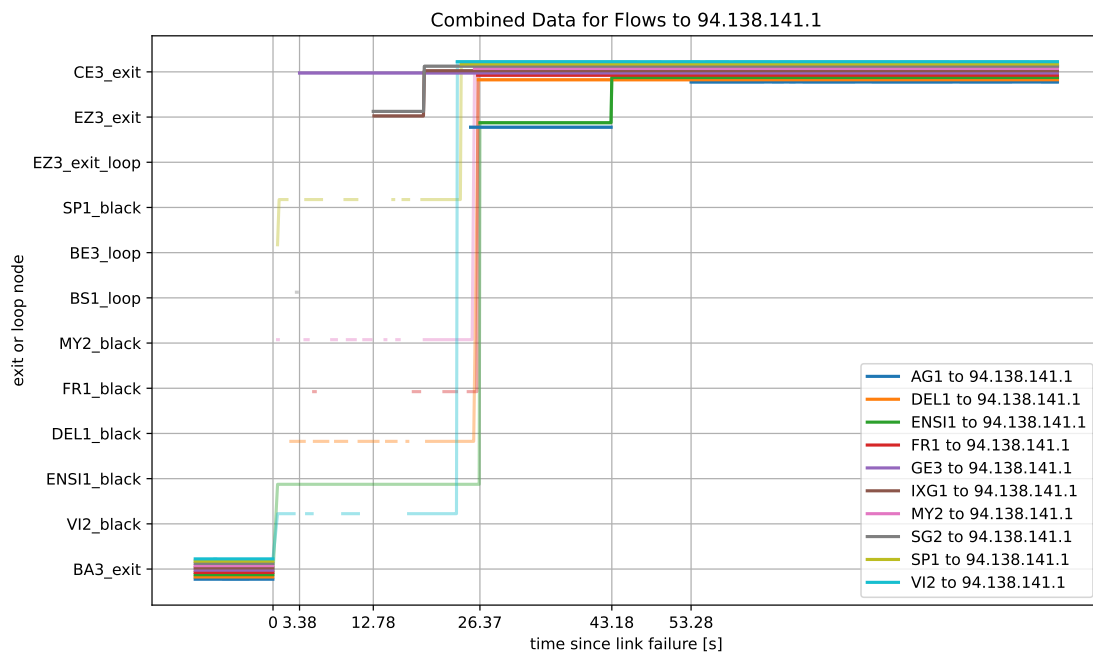


Figure C.49: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows towards 94.138.141.1

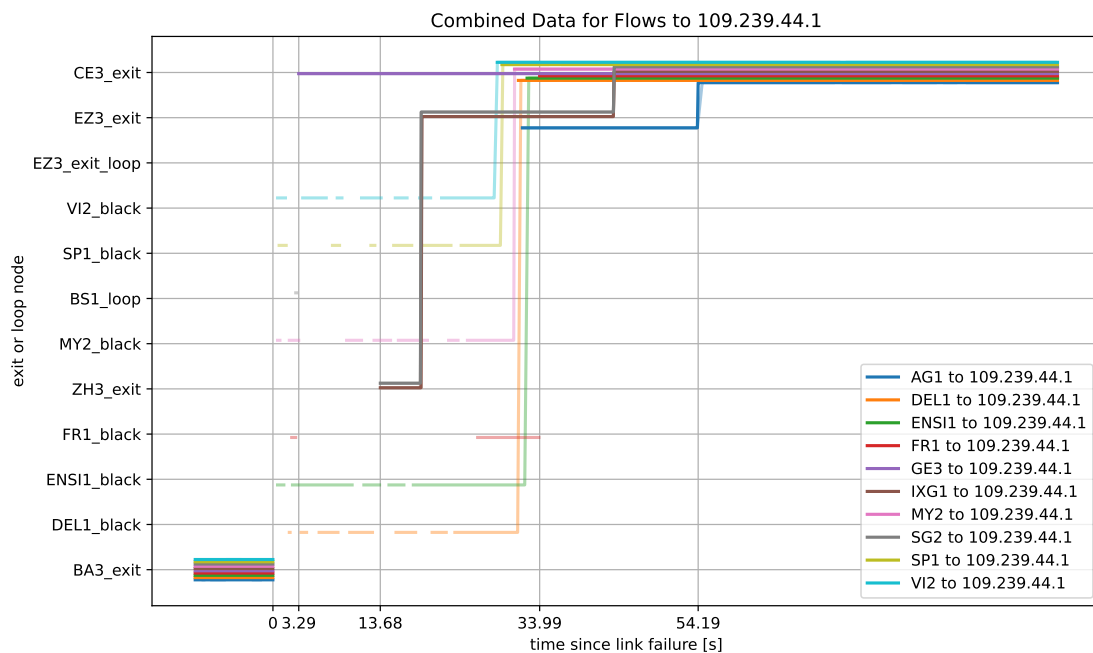


Figure C.50: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows towards 109.239.44.1

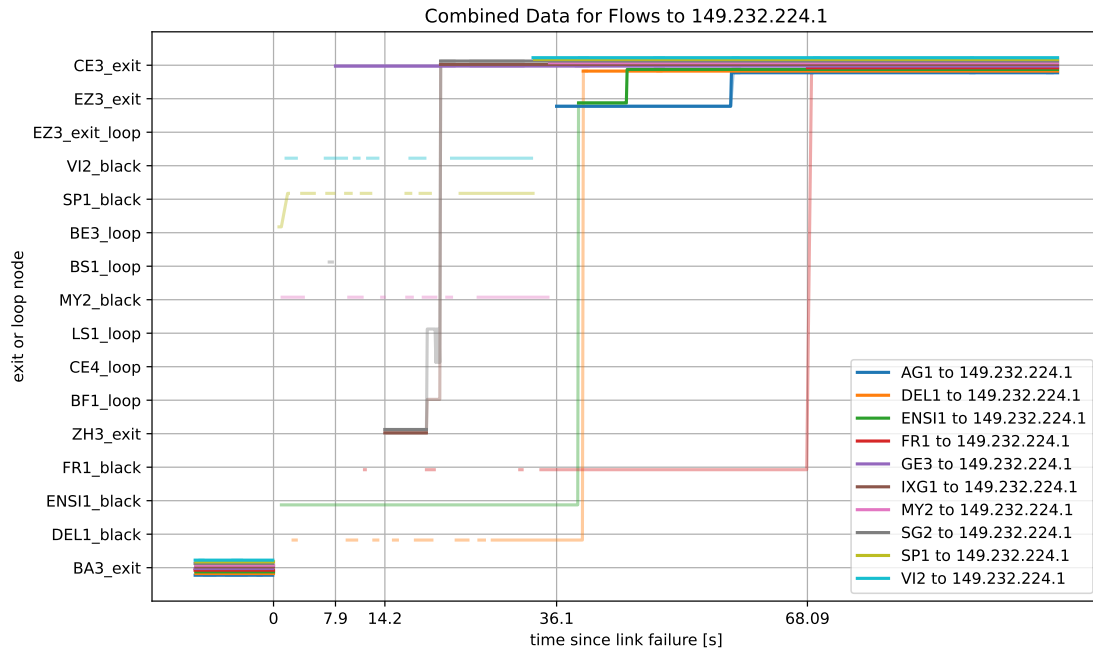


Figure C.51: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows towards 149.232.224.1

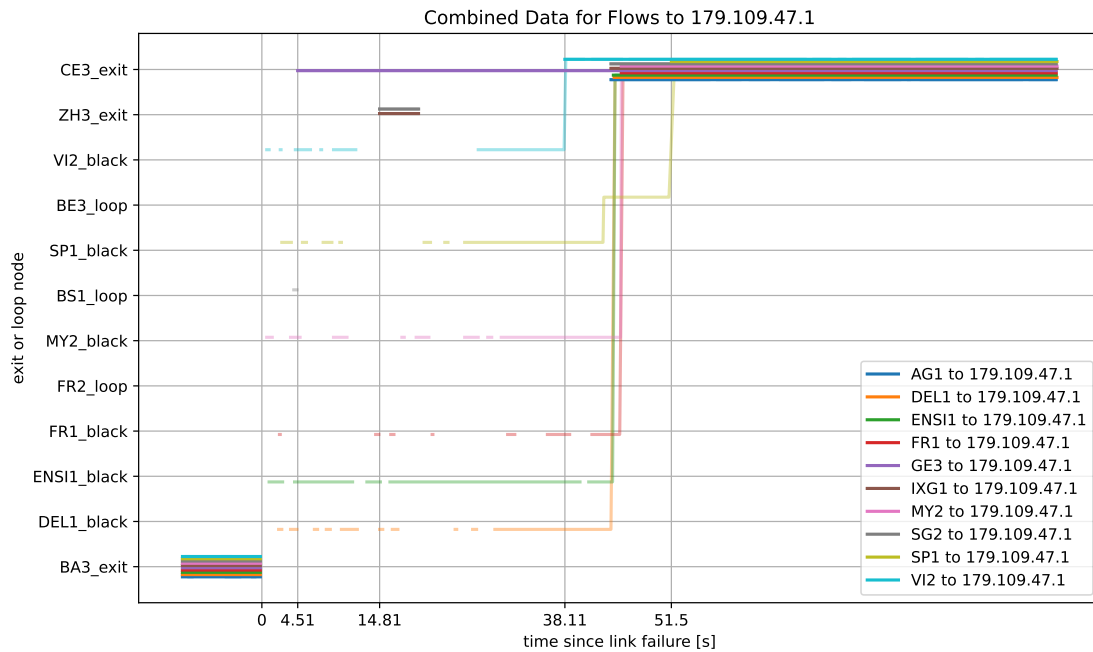


Figure C.52: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows towards 179.109.47.1

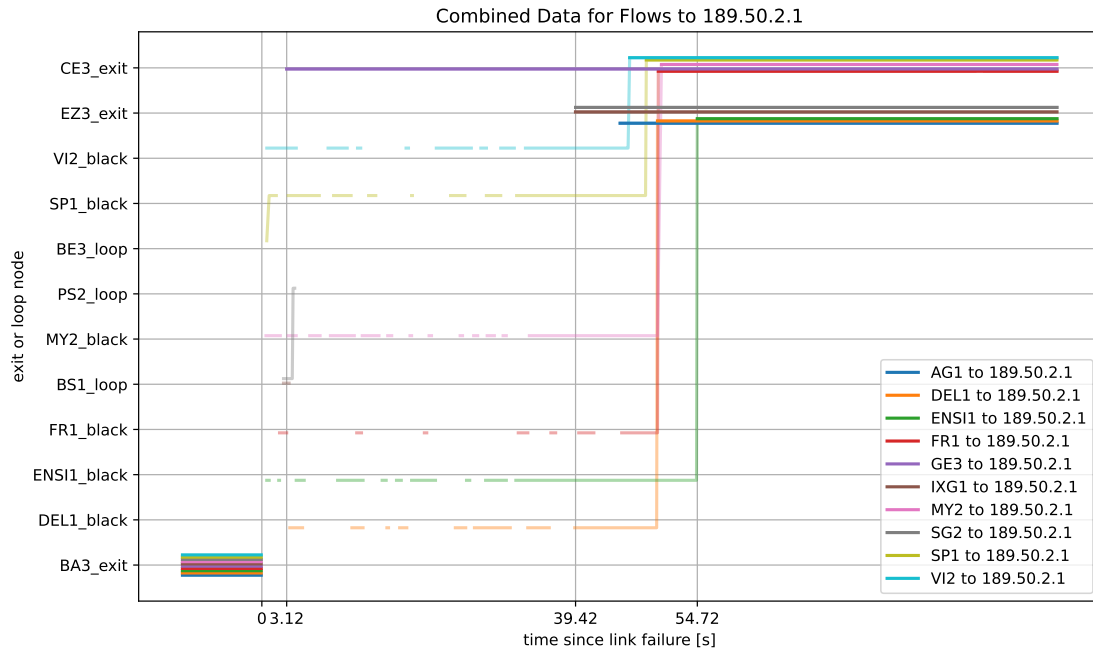


Figure C.53: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows towards 189.50.2.1

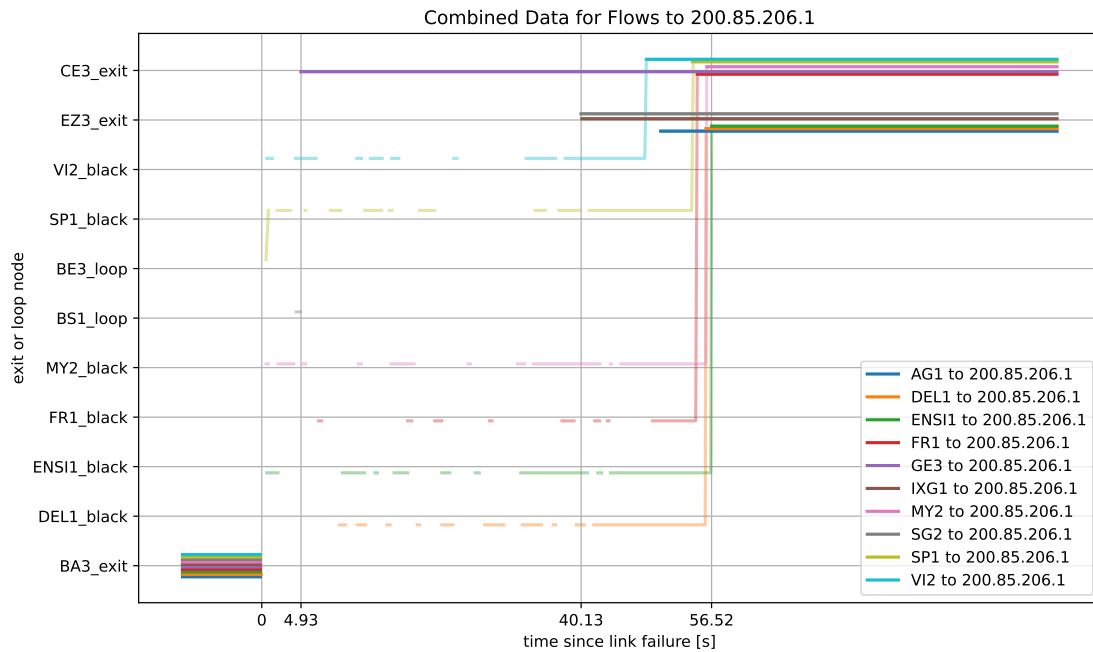


Figure C.54: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows towards 200.85.206.1

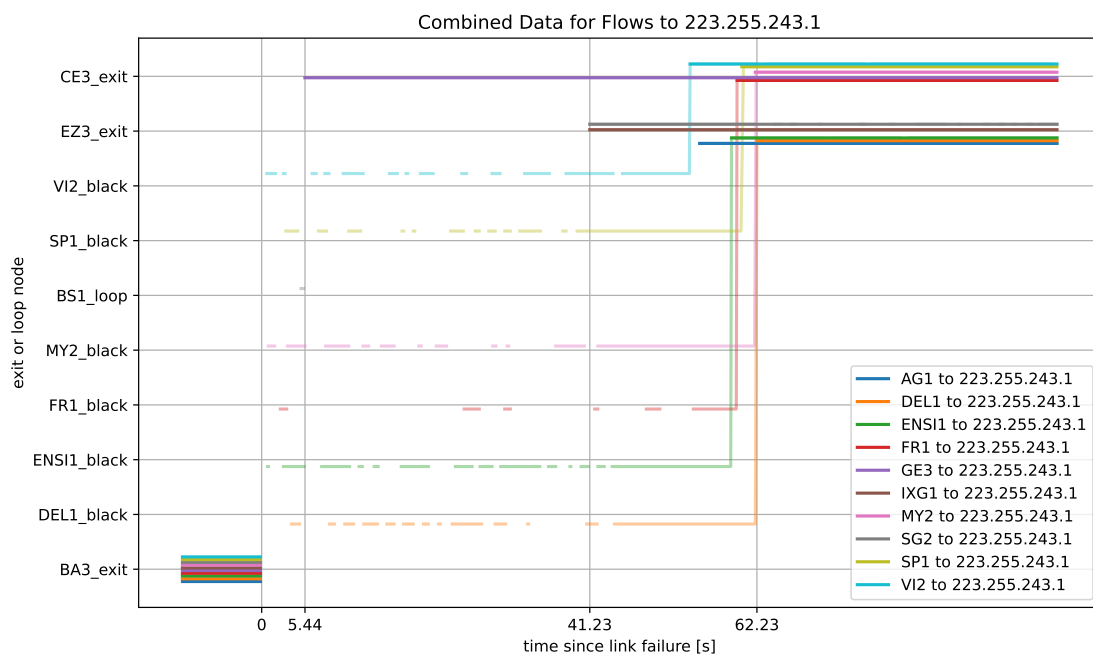


Figure C.55: Combined trace (semi-transparent) and capture (solid) data showing the convergence process for flows towards 223.255.243.1