Exploration of Internet Routing using GNS3

ELEN90061: Communication Networks

by

Christopher Loo 285106
Kelly Dionisio-See 1165953
Mukul Chodhary 1172562

in the
Department of Electrical and Electronics Engineering
Faculty of Engineering and IT
The University of Melbourne

August 5, 2025

Contents

1 Introduction (Mukul)

The Internet has become an integral part of our daily lives, facilitating global communication, commerce, and information exchange. At the core of this vast network lies the crucial routing process, which determines how data packets traverse the complex web of interconnected devices. Understanding and optimising routing protocols is essential for maintaining efficient, reliable, and secure network operations (Tanenbaum & Wetherall, 2011).

This project aims to comprehensively explore and analyse various Internet routing protocols using GNS3 (Graphical Network Simulator-3), a powerful network simulation tool (GNS3, 2024). We adopt a hands-on approach to explore routing protocols, leveraging GNS3 to create small-scale networks and configure FRR routers with well-known protocols. By observing their behaviour through Wireshark (Sanders, 2017), we aim to deepen our understanding of Internet routing mechanisms and develop skills crucial for designing and managing complex networks. Our methodology involves simulating various scenarios, including link failures and changes in link characteristics, to evaluate how routing protocols respond to network changes. Through these scenarios, we seek to gain valuable insights into the behaviour, performance, and scalability of key routing protocols.

The primary objectives of this study are:

1. To compare and contrast the functionality of RIP (Routing Information Protocol), OSPF (Open Shortest Path First), and EIGRP (Enhanced Interior Gateway Routing Protocol) in various network topologies.
2. To analyse the convergence times, scalability, and efficiency of each protocol under different conditions.
3. To identify the strengths and weaknesses of each protocol and determine their suitability for specific network environments.
4. To gain practical experience in configuring and troubleshooting routing protocols using GNS3 and FRRouting (FRR), a free and open-source Internet routing protocol suite for Linux and Unix platforms (FRRouting, 2024b).

The study begins with a comprehensive exploration of theoretical backgrounds in Section 2. This includes an examination of graph theory, classification of routing algorithms, and fundamental routing algorithms such as Dijkstra's algorithm, Bellman-Ford algorithm, and the Diffusing Update Algorithm. We also delve into the theoretical aspects of RIP, OSPF, and EIGRP, discussing their features, improvements, and potential issues. The subsequent Experiments section (Section 3) applies this theoretical foundation to introduce networks and experiments designed in GNS3, specifically crafted to highlight the strengths and weaknesses of each routing protocol. This section also presents simulation results and analyses our findings. In the Discussion section (Section 4), we synthesise these results to comprehensively compare and contrast each routing protocol. Finally, the Conclusion (Section 5) offers a reflection on the project's outcomes, provides a broad discussion of the implications, and outlines future research directions that naturally emerge from this study.

2 Theoretical Background (Chris, Kelly, Mukul)

Routing is a fundamental process in computer networks, responsible for determining the optimal path for data to travel from source to destination. The efficiency and reliability of network communication heavily depend on the effectiveness of routing algorithms. This section provides an overview of the theoretical foundations of routing algorithms and their practical implementations in widely-used protocols.

a Fundamentals of Routing

At its core, routing is the process of selecting paths in a network along which to send network traffic. Routing is performed for many types of networks, including the telephone network, electronic data networks (such as the Internet), and transportation networks. In packet-switching networks, routing directs packet forwarding, the transit of logically addressed packets from their source toward their ultimate

destination through intermediate nodes. The routing process usually directs forwarding on the basis of routing tables, which maintain a record of the routes to various network destinations. Thus, constructing routing tables, which are held in the router's memory, is very important for efficient routing. Most routing algorithms use only one network path at a time, but multipath routing techniques enable the use of multiple alternative paths (Tanenbaum & Wetherall, 2011).

i Graph Theory

Graph theory is integral to understanding routing algorithms. A network can be represented as a graph $G = (N, E)$, where N is the set of nodes (routers) and E is the set of edges (links) connecting these nodes. Routing algorithms use network construction as graphs to find the best route for the packets, whether by constructing a minimum-spanning tree, simply finding the shortest path to the nearest node, or constructing subgraphs to optimise for different criterion functions.

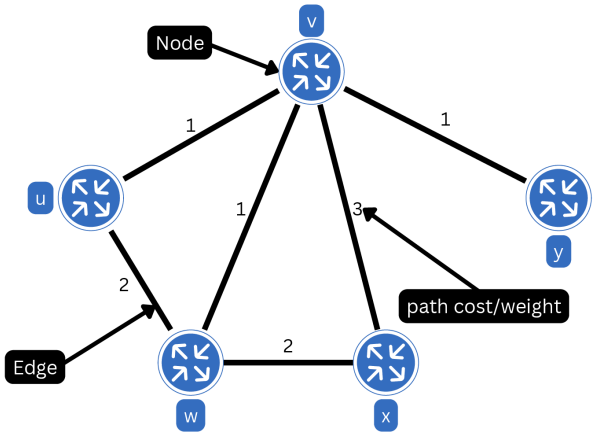


Figure 1: Abstract graph model of a computer network.

b Classification of Routing Algorithms

Routing algorithms can be classified along three main dimensions: global vs. decentralised, static vs. dynamic, and interior vs. exterior gateway protocols.

i Link State vs. Distance Vector Algorithms vs. Path Vector Protocols

Routing algorithms are classified into link-state, distance vector, and path vector protocols, each with unique characteristics suited to various network and routing scenarios.

Link-state (LS) algorithms, such as OSPF (Open Shortest Path First) and IS-IS (Intermediate System to Intermediate System), operate with full knowledge of the network topology. Each router builds a complete map of the network by exchanging link-state advertisements (LSAs) with all other routers, allowing it to independently compute the shortest paths to all destinations using algorithms like Dijkstra's. Although link-state protocols offer quick convergence and precise routing due to their global view, they can generate substantial message overhead, with complexity $O(nE)$, where n is the number of nodes and E the links, potentially impacting scalability in larger networks (Kurose & Ross, 2020).

In contrast, distance vector (DV) algorithms, such as RIP (Routing Information Protocol), operate with only local information. Routers share routing updates solely with immediate neighbours, using the Bellman-Ford algorithm to iteratively update distance vectors. While DV protocols reduce message overhead by relying on incremental updates, they generally exhibit slower convergence times and are vulnerable to issues like routing loops and the count-to-infinity problem. These challenges arise as DV protocols propagate updates slowly, delaying the correction of inaccurate routing information (Tanenbaum & Wetherall, 2011).

Path vector (PV) protocols, exemplified by Border Gateway Protocol (BGP), are primarily used for inter-domain routing between autonomous systems (ASes). Unlike LS and DV, path vector protocols maintain the entire path data must traverse across ASes, incorporating this information into BGP's AS-PATH attribute. This allows BGP to enforce policy-based routing based on attributes such as AS-PATH length or NEXT-HOP preference. Path vector protocols are particularly suited for large-scale networks, like the global Internet, as they provide control over routing policies between administrative domains. However, BGP's reliance on policy-based decisions rather than technical metrics like bandwidth can sometimes lead to performance trade-offs (Halabi & McPherson, 2000; Kurose & Ross, 2020).

Each type of algorithm has distinct advantages and limitations, depending on the scale and complexity of the network. Link-state protocols excel in environments where rapid convergence and precise routing decisions are essential but may face scalability challenges due to higher message overhead. Distance vector protocols are simpler and require fewer resources but are slower to converge and more prone to routing errors. Path vector protocols support scalable inter-domain routing while giving administrators control over traffic flow across networks.

ii Static vs. Dynamic Algorithms

Routing algorithms can be classified as either static or dynamic. Static algorithms are manually configured and remain unchanged unless updated by network administrators. In contrast, dynamic algorithms automatically adjust to changes in network topology by periodically updating routing tables. Static routing is predictable and ideal for smaller, stable networks where changes are infrequent. For example, a small office network with a single connection to an ISP may use static routing for simplicity. Static routes do not require bandwidth for updates, making them secure and resource-efficient (Cisco Press, 2016). However, static routing does not scale well in larger networks and cannot automatically reroute traffic in case of failures, leading to potential downtime (Kurose & Ross, 2020). Dynamic routing addresses these limitations by detecting network changes and adjusting routes accordingly. Protocols like OSPF and RIP use algorithms such as Dijkstra's and Bellman-Ford to compute optimal paths based on metrics like hop count or link cost (Tanenbaum & Wetherall, 2011). Dynamic routing is particularly useful in larger networks with frequent changes, as it can quickly adapt to link failures without manual intervention. However, this flexibility comes at the cost of higher resource consumption since dynamic protocols require more CPU power and bandwidth to maintain up-to-date routing tables (Medhi & Ramasamy, 2017).

iii Interior and Exterior Gateway Protocols

Additionally, routing algorithms can be categorized as interior gateway protocols (IGPs) for routing within autonomous systems, or exterior gateway protocols (EGPs) for routing between autonomous systems. Some common examples include Open Shortest Path First (OSPF) as a link-state IGP, Routing Information Protocol (RIP) as a distance vector IGP, and Border Gateway Protocol (BGP) as the primary EGP used on the internet.

Protocol	**LS vs DV vs PV**	**Static vs Dynamic**	**IGP vs EGP**
RIP	Distance Vector	Dynamic	IGP
OSPF	Link State	Dynamic	IGP
EIGRP	Advanced Distance Vector	Dynamic	IGP
MPLS	N/A	Dynamic	N/A
BGP	Path Vector	Dynamic	EGP

Table 1: Classification of Routing Protocols

c Fundamental Routing Algorithms

Fundamentally, routing algorithms intend to find the shortest path for the packet to travel on from its source to destination. Thus, the routing algorithms are closely based on shortest path algorithms, specifically Djikstra's, Bellman–Ford's and the Diffusing Update algorithms.

i Dijkstra's Algorithm

Dijkstra's algorithm is a fundamental component of link-state routing protocols, widely used to determine the shortest path from a source node to all other nodes in a network. The algorithm operates by maintaining a set of nodes whose shortest path from the source has been definitively determined. It iteratively expands this set by selecting the node with the minimum tentative distance, and updating the shortest paths to its neighboring nodes accordingly.

The algorithm begins by initializing the distance to the source node as zero and all other nodes as infinity. At each step, it selects the node with the smallest tentative distance, adds it to the set of nodes with known shortest paths, and updates the distances to its neighbouring nodes using the equation:

$$D(n) = \min(D(n), D(m) + c(m, n))$$

where $D(n)$ is the current minimum distance to node n, $D(m)$ is the known shortest distance to node m, and $c(m, n)$ is the cost of the edge between nodes m and n. This process continues until all nodes have been processed, ensuring that each node's shortest path from the source is accurately determined.

Dijkstra's algorithm is efficient for graphs with non-negative weights and is particularly suited for static networks where link costs do not change frequently. Its computational complexity is $O(N^2)$, though implementations using priority queues can achieve better performance, $O((N+E)log(N))$ with N nodes and E links. The algorithm's ability to provide precise routing information makes it a cornerstone in network routing, especially in protocols like OSPF (Open Shortest Path First).

ii Bellman–Ford Algorithm

The Bellman–Ford algorithm is the foundation for distance vector protocols and is instrumental in routing protocols like RIP (Routing Information Protocol). Unlike Dijkstra's algorithm, Bellman–Ford can handle graphs with negative edge weights, making it versatile for various network scenarios. The algorithm works by iteratively relaxing the estimated distance to each node, ensuring that the shortest path to each node is gradually improved upon with each iteration.

The core principle of the Bellman–Ford algorithm is based on dynamic programming. It starts by initializing the distance to the source node as zero and all other nodes as infinity. The algorithm then repeatedly updates the distance to each node by considering all edges in the graph. For each edge from node x to node n, it calculates whether a shorter path exists through n using Bellman's equation:

$$d_x(y) = \min_n\{c(x, n) + d_n(y)\}$$

where $d_x(y)$ represents the shortest known distance from node x to node y, and $c(x, n)$ is the cost of the edge from node x to node n. This process is repeated for $N - 1$ iterations, ensuring that all possible shortest paths are considered.

One of the notable features of the Bellman–Ford algorithm is its ability to detect negative weight cycles. If a shorter path is found after $N-1$ iterations (where N is the number of nodes), it indicates the presence of a negative cycle, which can lead to infinitely decreasing path costs. Despite its higher computational complexity than Dijkstra's algorithm—specifically, $O(N \cdot E)$, the Bellman–Ford algorithm's robustness in handling negative weights makes it a critical tool in specific network applications. It operates effectively in decentralized environments, where each router only needs information about its immediate neighbours rather than global network knowledge.

iii Diffusing Update Algorithm

The Diffusing Update Algorithm (DUAL) enables routers to determine the shortest path to each destination and also identify backup paths without causing routing loops. As a hybrid protocol, DUAL combines aspects of link-state and distance vector protocols but aligns more closely with distance vector behaviour. Similar to the Bellman-Ford algorithm, DUAL routers share routing information with neighbouring routers, advertising a distance metric to each destination. However, rather than a complete network map (as in link-state protocols using Dijkstra's algorithm), each DUAL router knows only the next hop and the distance to each destination. This approach allows DUAL to calculate the shortest path, known as the *successor path*.

DUAL differs from Bellman-Ford in two primary ways. First, each time it identifies successors, DUAL also finds backup paths called *feasible successors*, which are stored in a topology table. The routing table then uses these shortest paths to make forwarding decisions. When a route fails, a router can quickly replace the route in its routing table with a feasible successor, allowing rapid convergence since a backup path is already pre-calculated. Secondly, DUAL's topology table tracks two metrics: feasible distance (FD) and reported distance (RD), also known as advertised distance. RD is the distance to a destination that a router advertises to its neighbours, while FD is the router's total distance to that destination. Neighbouring routers calculate their own FD to the destination by adding the distance advertised (RD) by their neighbour and the distance between them.

DUAL guarantees that backup paths do not cause loops using a *feasibility condition*: a backup path can only be selected as a feasible successor if the neighbour's RD is less than the FD of the current successor. In this way, DUAL ensures that any neighbour offering a backup path is closer to the destination than the current best path. Although this rule may bypass a nearer path, it ensures that any feasible successor is loop-free, as it cannot direct traffic back to the router in the original path. Violating this condition could lead to packets being sent back to the original router, causing a routing loop. By guaranteeing loop-free backup paths and using feasible successors to enable rapid convergence, DUAL achieves greater efficiency than Bellman-Ford and solidifies its status as a hybrid algorithm.

d Practical Implementations: Routing Protocols

i RIP (Routing Information Protocol) (Mukul)

Routing Information Protocol (RIP) is a well-established distance-vector routing protocol that has been instrumental in the development of network routing since its inception in the 1970s at Xerox PARC. Designed primarily for small to medium-sized networks, RIP operates effectively in environments with a simple, flat network structure and a limited number of routers, typically not exceeding 15 hops between any two points (Tanenbaum & Wetherall, 2011).

RIP utilises the Bellman–Ford algorithm to calculate the shortest path based on hop count, with a maximum allowable hop count of 15 to prevent routing loops. This simplicity makes RIP easy to configure and understand, which is beneficial in networks where ease of implementation and low resource usage are prioritised (Kurose & Ross, 2020). However, this simplicity also limits its scalability and efficiency in larger, more complex networks.The protocol periodically broadcasts the entire routing table to the neighbouring routers every 30 seconds. This periodic update mechanism helps maintain synchronised routing information across the network but can lead to slow convergence times and increased network traffic(Medhi & Ramasamy, 2017). In addition to the update timer, RIP-2 utilises several other timers to manage routing stability. The invalid timer is set to 180 seconds; if no updates are received for a route within this period, it is marked as invalid. This marking prompts routers to prepare for potential route changes. The hold-down timer, also typically set to 180 seconds, prevents routers from accepting potentially incorrect information during network changes by maintaining the invalid state until accurate updates are confirmed. Finally, the flush timer is set to 120 seconds and dictates how long a router should retain information about an unreachable route before removing it entirely from its routing table (FRRouting, 2024b). These timers collectively ensure that routing information remains accurate and

stable across the network, although they can contribute to slower convergence times compared to more advanced protocols.

RIP has evolved through several versions, each addressing the specific limitations of its predecessors. RIP Version 1 (RIP-1) is a classful protocol that does not support subnetting or authentication, using broadcast addresses for routing updates. Classful routing protocols do not carry subnet masks. This approach can lead to inefficiencies and security vulnerabilities in modern network environments (Halabi & McPherson, 2000). RIP Version 2 (RIP-2) introduced several enhancements over RIP-1, making it more adaptable to contemporary networking needs. RIP-2 supports Classless Inter-Domain Routing (CIDR) and Variable Length Subnet Masks (VLSM), allowing for more efficient IP address allocation and flexible subnetting (Internet Engineering Task Force, 1998). Additionally, RIP-2 employs multicast addressing for routing updates using the address '224.0.0.9', which reduces unnecessary traffic by targeting only routers configured to receive updates (Internet Engineering Task Force, 1998). This method enhances network efficiency by minimising traffic sent to non-router devices. RIP-2 also significantly improves security. While RIP-1 lacks authentication mechanisms, making it susceptible to routing disruptions and attacks, RIP-2 supports authentication methods such as plain text and MD5, ensuring that routing updates are accepted only from trusted sources (Internet Engineering Task Force, 1998). This enhancement significantly bolsters network security by preventing unauthorised access and spoofing attacks. RIP-2 also uses split horizon with poison reverse to overcome the count-to-infinity problem, which occurs when routers continue to increment the hop count for a failed route indefinitely. Without split horizon and poison reverse, routers might keep advertising unreachable routes back to their neighbours, leading to routing loops and incorrect path calculations. Moreover, RIP-2 supports triggered updates, allowing routers to promptly inform neighbours about network changes rather than waiting for the next periodic update interval. This feature improves the protocol's responsiveness and adaptability in dynamic network environments (Halabi & McPherson, 2000).

Despite these improvements, RIP's inherent limitations in scalability and convergence speed compared to more advanced protocols like OSPF or EIGRP mean it is best suited for smaller networks where simplicity is prioritised over scalability. However, its compatibility across different router types allows it to be integrated into diverse network environments (Kurose & Ross, 2020).

ii OSPF (Open Shortest Path First) (Kelly)

The Open Shortest Path First (OSPF) is an open-standard protocol introduced in the mid-1980s to address the limitations of RIP (Verma & Bhardwaj, 2016). Given that it is an open-standard, this routing protocol is publicly available and user-friendly. It is an interior gateway protocol that utilises a link-state algorithm, specifically Dijkstra's algorithm (Verma & Bhardwaj, 2016), to determine the optimal path from source to destination by calculating the shortest path. In addition to that, OSPF uses the total link cost to identify the shortest path.

OSPF works well in managing large topologies or ASs by dividing them into smaller areas and introducing hierarchical structures (IBM, 2024). Each area runs its own link-state algorithm independently and regularly updates its state through link-state advertisements (LSAs), which contain information about the most updated shortest paths from OSPF neighbours and adjust their routing tables accordingly (IBM, 2024). With the recent adoption of IPv6, OSPF is also able to maintain its effectiveness in this shift.

OSPF has been introduced by the Internet Engineering Task Force to improve upon the shortcomings of RIP (Verma & Bhardwaj, 2016). Unlike RIP, OSPF offers better security as all messages are authenticated and exclusive to OSPF. It also increases the routing efficiency as it supports multiple paths with equal costs (IBM, 2024), allowing for alternative routing in response to network changes. Moreover, OSPF can integrate multicast capabilities easily with the address 224.0.0.5 using the same topology as traditional OSPF. Furthermore, its hierarchical design reduces overheads and enhances scalability (IBM, 2024), making it ideal for large corporate environments. Due to its scalability, OSPF is an ideal protocol for smaller networks that may anticipate future network growth.

Hierarchical OSPF breaks up and organises the AS into areas, each governed by Area Border Routers (ABRs) (Senanayake, 2024). These ABRs facilitate communication between their area and the backbone area, also known as Area 0, which allow for information exchange between areas (Verma & Bhardwaj, 2016). Each of these ABRs broadcasts *Hello* packets to maintain a connection with one another and identify its neighbours (Biradar, 2020). Using these *Hello* packets and LSAs, these routers store the topology of the routers in their vicinity and neighbour information within their memory. This structure optimises performance by reducing the routing complexity from $O(n^2)$ to $O(n)$ (Malik et al., 2012).

However, OSPF can be quite complex to configure, which makes setups more difficult as the network size increases and topology changes. This complexity leads to a higher memory use for storing neighbour information. Additionally, the algorithm used to calculate the shortest path would require more CPU processing as the size of the network increases (Verma & Bhardwaj, 2016).

iii EIGRP (Enhanced Interior Gateway Routing Protocol) (Chris)

Enhanced Interior Gateway Routing Protocol (EIGRP) is an advanced distance vector protocol used to exchange routing information between routers within a single autonomous system in a computer network. It employs several metrics, namely bandwidth, delay, reliability, load and maximum transmission unit (MTU) to calculate the best path for routing packets through a network. It is also a classless protocol that supports variable-length subnet masks and classless inter-domain routing (CIDR).

In EIGRP, routers maintain three lists: a) a list of neighbouring routers; b) a topology table; and c) a routing table. A router wanting to send a packet to a distant subnet will receive information from each of its neighbouring routers as to whether it can reach that subnet and the cost with which it can do so; by using the value of the metric to reach those neighbours it decides which neighbour is on the winning route. The router will then advertise this winning route to any neighbour routers downstream of it from the destination.

The way information being shared between neighbouring routers leads to finding the shortest path to a destination can be broken down into a three-step process:

1. Becoming neighbours (and maintaining the list of neighbours). As with the other routing protocols, EIGRP routers form neighbour relationships with their connected routers. Once EIGRP is enabled on a router, it will start looking for potential neighbours using a hello message. Hello messages are also used to maintain neighbour relationships, to let a router's neighbours know it is still alive. The time between Hello messages being sent is called the Hello interval; this is 5 seconds by default for high bandwidth links or 60 seconds for low bandwidth links. Hello messages are sent to the multicast address 224.0.0.10 (an instruction to send information to neighbours). If a Hello message is not received from a neighbour within three times the Hello interval (15 seconds by default, or 180 seconds), the link between them is assumed to be dead (this time is called the Hold timer, similar to OSPF's dead time).

When a router receives a Hello message from a neighbour, it runs some checks to decide whether or not to become neighbours with that routers: a) AS number: the autonomous system number of the router, which is set when configuring EIGRP on the router, must be the same on both routers; b) Subnet: routers must be on the same subnet; c) K-values must match (the values used to calculate the decision metric, in reality these are seldom changed); and d) Authentication (this also needs to match if authentication is being used). It's worth noting that EIGRP doesn't require the Hello and Hold timers to match; as long as the four sets of values just described match, the router receiving a Hello message will reply back with a Hello message of its own, thus establishing the neighbour relationship.

2. Exchanging routing information to populate the topology table. Neighbours will then exchange topology information containing all the known routes from each router to each possible destination and the distance to each destination. Rather than UDP or TCP, to send update messages EIGRP uses RTP (reliable transport protocol), which uses sequence numbers to identify whether messages have

been received by the neighbours. EIGRP then uses DUAL (described above) to handle all route computation to ensure no routing loops occur. Both routers will send full update messages, which contain all routing information known by that router; an ACK message is then sent to acknowledge the delivery of the update message; and once all routing information has been shared, from that point onwards, only partial updates will be sent if a change occurs in the network. If there are no changes, then only the friendly Hello messages will be sent back and forth.

If a link goes down, and there are no backup routes (no feasible successors), the router will start *route recomputation*, where the router will try and find a loop-free route to the subnet that just got cut off. The route first enters an 'Active' state — active here not meaning 'working' but rather 'active' in the sense that a route recalculation needs to be performed. Passive routes are unchanging and are working as they should; active routes are changing and the router is actively trying to find a path through to a destination. Active routes are listed in the EIGRP topology table with the *A* flag.

Then the connecting router (closest to the link that just went down) sends a query message to all its neighbouring routers asking if they have any routes to the lost subnet. The neighbours will either reply with a new route to the lost subnet, or tell the original router that there is no path. If there is no other route to that subnet, that destination is declared dead, and should be removed from everyone's routing tables.

3. Populating the routing table. Having exchanged routing information, routers need to calculate the best routes from the topology table and add these to their routing table. The distance/metric calculation is a complicated formula involving five possible inputs (weighted by K values):

$$\left[\left(\frac{K_1 \cdot 10^7}{\text{Bandwidth}} + \frac{K_2 \cdot 10^7}{\text{Bandwidth} \cdot (256 - \text{Load})} + K_3 \cdot \text{Delay}\right) \cdot \frac{K_5}{K_4 + \text{Reliability}}\right] \cdot 256$$

K	Name	Description	Default value
K_1	Bandwidth	Lowest bandwidth of all the hops in the route (in kbps)	1
K_2	Load	Worst load on the route based on packet rate	0
K_3	Delay	Cumulative interface delay through all hops in the route	1
K_4	Reliability	Packet error rate of the worst interface	0
K_5	MTU	Smallest maximum transmission unit in the path	0

Table 2: Values used in the EIGRP Metric Calculation. By default, only bandwidth and delay are used in the calculation. K_2 needs to be 1 in order to implement unequal-cost load balancing. If K_5 is set to zero, the final fraction in the expression is taken to have a value of 1.

By default, $K_1 = K_3 = 1$ and $K_2 = K_4 = K_5 = 0$, and only bandwidth and delay end up being used in the metric calculation (load and reliability are rarely used). Bandwidth here is the lowest bandwidth in kbps out of all the links in the path. Delay is the cumulative delay of all the hops in a path, measured in 10s of microseconds. By definition, bandwidth, delay and MTU are static measures, whereas load and reliability are dynamic. For example, a Fast Ethernet link is 100 Mbps and has a delay of 100 microseconds, whereas Ethernet is 10 Mbps and has a delay of 1 millisecond — these values don't change.

The DUAL algorithm (described above) is then applied — calculated metrics are used by routers to produce a reported distance (RD), which is then used by neighbouring routers to calculate the feasible distance (FD). The *successor* route (best route with the lowest metric value) is placed in the routing table. EIGRP allows for multiple equal cost successor routes, and can load balance the traffic through each of these equal cost paths. It also supports (with some extra configuration) unequal-cost load balancing, allowing routers to distribute traffic over multiple paths based on the cost of each path. As mentioned, additionally, *feasible successors* (backup routes meeting DUAL's *feasibility condition*) are recorded in

the topology tables of each of the routers.

In the event of a link failure (i.e. link being declared dead), where that link is part of a successor route, the router instantly updates the routing table using a feasible successor as a new successor route. The convergence time is then equal to the hold time (the time it takes to declare a link dead when hello messages stop being sent). As mentioned, the feasible successor may not have the second-lowest metric because of the feasibility condition, but this condition ensures that no routing loops are formed when a feasible successor becomes the successor.

With the use of the DUAL algorithm, one of the key benefits of EIGRP is its fast convergence time (meaning routers quickly adapt to network changes and update their routing tables quickly). Other features (that won't be explored in this project) include EIGRP's ability to use both IPv4 and IPv6, and EIGRP can be configured to support authentication, which enhances network security by preventing unauthorised access to routing information. Finally, EIGRP originated as a Cisco proprietary routing protocol (for use on Cisco routers), replacing in 1993 IGRP (which did not support classless IPv4 addresses). Although Cisco permitted other vendors in 2013 to implement a limited version of EIGRP with a subset of its features (RFC 7868), Cisco retains control over the EIGRP protocol and so EIGRP is nowhere near as established or well-supported as OSPF by third-party vendors.

e Administrative Distance and Route Redistribution (Chris, Mukul)

In complex network environments where multiple routing protocols coexist, the concepts of administrative distance (AD) and route redistribution play crucial roles in determining optimal routing paths and facilitating interoperability between different protocols. Administrative distance is a metric used by routers to assess the reliability of routing information received from various sources. It can be conceptualised as the "believability" of a routing source, with lower AD values indicating higher trustworthiness. When a router learns multiple routes to the same destination through different routing protocols, it selects the route with the lowest AD to populate its routing table. This hierarchical trust system ensures consistent and predictable routing decisions across the network.

The AD values assigned to common routing protocols reflect their relative reliability and efficiency:

- Directly connected routes: AD = 0
- Static routes: AD = 1
- External BGP: AD = 20
- Internal EIGRP: AD = 90
- OSPF: AD = 110
- RIP: AD = 120
- Internal BGP: AD = 200

These predefined AD values can be manually adjusted to influence routing behaviour. For instance, a network administrator might configure a static route with an AD of 111 to serve as a backup for an OSPF-learned route (AD 110). This configuration ensures that the static route is only used if the OSPF route becomes unavailable, providing a failover mechanism without disrupting normal OSPF operations. An illustrative example of configuring AD demonstrates its practical application in network management. Consider a scenario where a network is known through OSPF, but there's a need for an alternative route in case the OSPF-learned route fails. In this case, a static route can be configured to serve as a backup. By default, static routes have an AD of 1, which would typically take precedence over the OSPF route (AD 110). However, by setting the AD of the static route to 111, it becomes a backup route. This configuration ensures that the static route becomes the next best candidate to populate the routing table only when the OSPF route goes down, effectively creating a failover mechanism that maintains network stability while prioritising dynamic routing under normal conditions.

Route redistribution is another critical concept that facilitates communication between different routing protocols. This process involves exchanging routing information between distinct protocols, which is essential when combining networks that utilise different routing protocols or transitioning from one

protocol to another. Routers configured for redistribution can import routes learned from one protocol into another, enabling seamless communication across diverse network segments. However, this capability presents challenges due to the disparate metrics used by different routing protocols. For example, RIP uses hop count, OSPF uses cost based on bandwidth, and EIGRP employs a composite metric. When redistributing routes, network administrators must manually assign appropriate metrics to ensure accurate route selection across protocol boundaries. This process requires careful planning to prevent routing loops and suboptimal path selection. The interaction between administrative distance and route redistribution becomes particularly significant in scenarios involving multiple routing protocols. For instance, when EIGRP routes are redistributed into OSPF, they are treated as external OSPF routes with an AD of 110. However, if these same routes were learned natively through EIGRP, they would have an AD of 90. This difference in AD can lead to unexpected routing behaviour if not properly managed.

Understanding and effectively utilising administrative distance and route redistribution are essential skills for network engineers working with multi-protocol environments. These concepts not only facilitate the integration of diverse routing protocols but also provide powerful tools for implementing robust, flexible, and efficient routing strategies in complex network architectures. By manipulating AD values and redistributing routes appropriately, network engineers can ensure optimal performance and reliability in diverse networking scenarios.

f GNS3 (Mukul)

Graphical Network Simulator-3 (GNS3) is a sophisticated open-source network simulation tool that enables users to design, test, and troubleshoot virtual networks. Initially developed to simulate Cisco, networks, GNS3 has evolved into a versatile platform that supports a wide range of network devices from multiple vendors. This capability makes it an invaluable tool for both network professionals and students aiming to gain practical experience in network configuration and management (GNS3, 2024). One of the key strengths of GNS3 is its ability to emulate actual networking environments. Users can create detailed network topologies that include routers, switches, firewalls, and other network appliances. The simulator also supports packet capture through integration with Wireshark, enabling detailed analysis of data flows within the network. This feature is particularly useful for debugging and optimizing network performance (GNS3, 2024). For these reasons, GNS3 was chosen as the preferable tool for performing exploratory experiments and gaining a deeper understanding of the routing protocols.

3 Exploratory Experimentation (Chris, Kelly, Mukul)

a Evaluation Metrics

The key problem of the project is designing specific networks and scenarios showcasing the strengths and weaknesses of routing protocols outlined in Section 2. Each routing protocol has its own strengths and weaknesses, and comparing them presents its own challenges. To effectively compare these protocols against each other, we consider the convergence time of the forwarding table as the primary evaluation metric. Other metrics, such as the shortest path, may not be considered valid here because the protocols try to optimise over a different criterion function. For example, RIP considers equal link costs, OSPF has different link costs and EIGRP optimises over multiple criteria. However, we can still explore each protocol individually and understand the problems that they attempt to solve and where they might fail.

b GNS3 setup

Setting up routers using FRRouting (FRR) involves a series of straightforward steps to ensure proper configuration and operation. FRR is a versatile open-source routing software suite that supports various protocols, making it ideal for network simulations and real-world applications. The scripts for setting up the router can be found in the documentation provided with the FRR package (FRRouting, 2024b). We provide a sample step-by-step guide in the Appendix a.

c Experiments

i RIP

Scenario 1: New routers join the network

In this scenario, we examine how a new router, FRR-3, integrates into an existing RIP-based network. Initially, the local area network (LAN) connected to FRR-3 is isolated from the rest of the network, as depicted in Figure 2. The network consists of multiple routers running RIP (Routing Information Protocol), a distance-vector protocol that relies on periodic updates to share routing information between neighbouring routers.

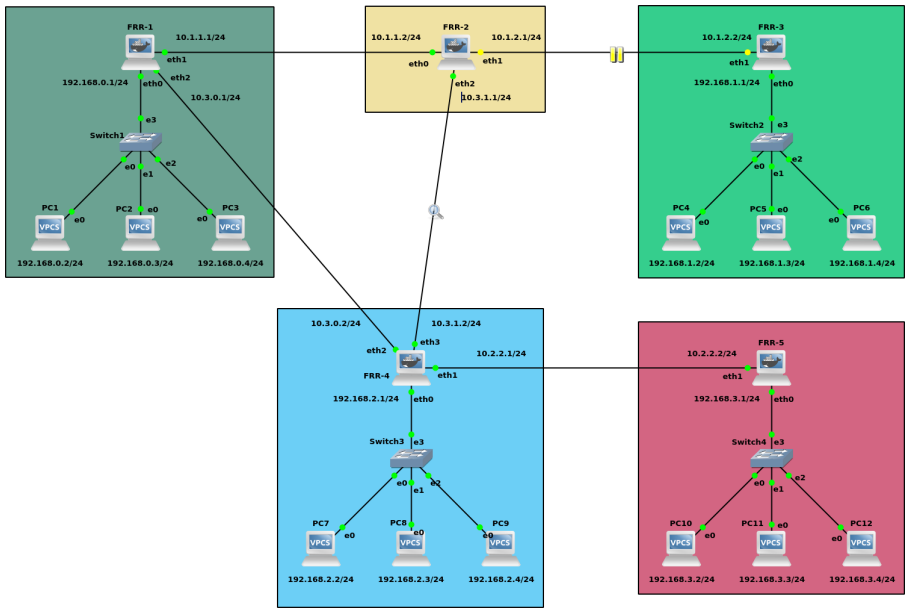


Figure 2: RIP Scenario 1 topology. FRR-3 joins the existing network.

At time $t = 10$, FRR-3 establishes its connection to the rest of the network and immediately sends a multicast broadcast to the address '224.0.0.9', which is reserved for RIP updates. This multicast occurs earlier than the standard 30-second update interval because RIP allows routers to send triggered updates when significant changes occur in the network topology, such as when a new router joins. This early multicast ensures that all neighbouring routers are promptly informed about the new routes available through FRR-3. The Wireshark capture in Figure 3 illustrates how FRR-3 advertises its directly connected networks (192.168.1.0/24 and 10.1.2.0/24) to its neighbours using RIP update packets. These packets contain distance-vector information, including hop counts to each destination network. At time $t = 30$, we observe the usual periodic RIP updates from other routers in the network, such as FRR-2 and FRR-4. These updates occur every 30 seconds as part of RIP's regular operation. By this time, FRR-2 has received and processed the routing information from FRR-3 and has updated its routing table to include the LAN connected to FRR-3 (192.168.1.0/24). The routing table now reflects that traffic destined for this LAN can be forwarded through FRR-3 with a hop count of 2, indicating that FRR-2 has learned about this route indirectly through another router.

A key observation from Figure 3 is that RIP-V2 still uses classful addressing in its distance-vector packets. For instance, even though modern networks often use classless inter-domain routing (CIDR), RIP continues to advertise routes based on classful IP addresses without explicitly including subnet mask information in its updates. As seen in Figure 3, networks such as 192.168.1.0/24 are advertised without any indication of their subnet mask, leading to inefficiencies in more complex networks where subnetting is used extensively.

The behaviour observed in this scenario highlights several key aspects of RIP's operation. First, it demonstrates how new routers can seamlessly integrate into an existing network by broadcasting their routing information using triggered updates. This mechanism ensures that changes in the network

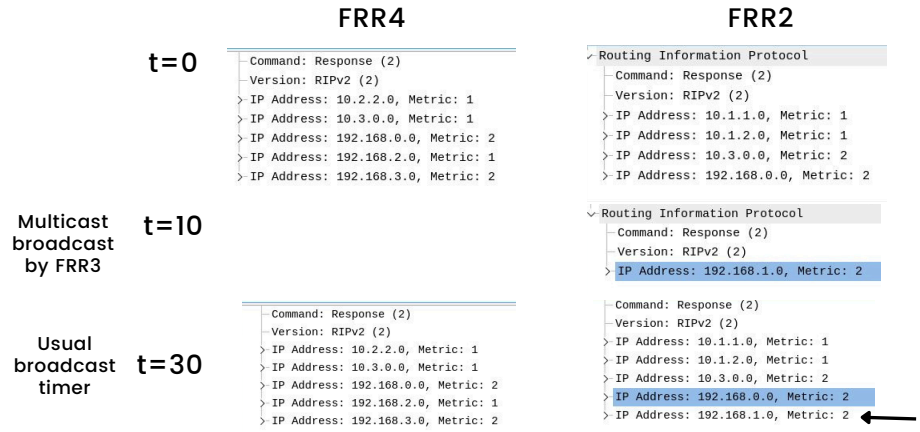


Figure 3: RIP distance-vector packets observed through Wireshark at different time intervals (t=0, t=10, t=30).

topology are propagated quickly without waiting for the next scheduled update interval. Second, it shows how RIP's distance-vector algorithm allows routers to learn about routes indirectly through neighbouring routers, with each router maintaining only local knowledge of its immediate neighbours' routes and hop counts. Lastly, this scenario illustrates some limitations inherent in RIP's design. While triggered updates help speed up convergence when changes occur, reliance on periodic updates means that convergence can still be slow in larger networks with frequent changes. Additionally, because RIP uses hop count as its sole metric for route selection, it may not always choose the most efficient path in terms of bandwidth or latency. Therefore, from this scenario, we can see how router FRR-3 joins the network and how routing information is exchanged using RIP's distance-vector mechanism.

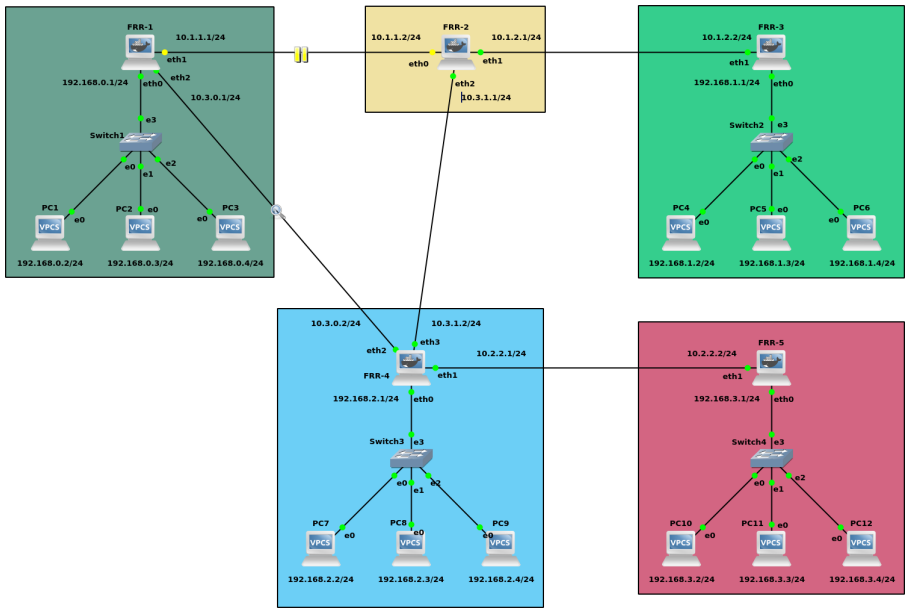


Figure 4: RIP Scenario 2 topology. A link failure between FRR-1 and FRR-2.

Scenario 2: Changes to topology

In this scenario, the network experiences a failure of the link between router FRR-1 and FRR-2, as shown in Figure 4. This event triggers a series of updates across the RIP-based network as routers adjust their routing tables to reflect the change in topology. RIP, being a distance-vector protocol, relies on periodic updates to propagate changes in the network. However, when a significant event such as a link failure occurs, RIP uses timers and specific metrics to handle unreachable routes.

When the link between FRR-1 and FRR-2 goes down, FRR-1 begins the process of marking the route through FRR-2 as unreachable. According to RIP's standard behaviour, an invalid timer is started for each route that becomes unreachable. In this case, we observe that it takes approximately 180 seconds for FRR-1 to declare the route through FRR-2 as invalid, as shown in Figure 5. During this time, FRR-1 continues to send distance-vector updates to its neighbours, but with an increased metric of 16 for the affected routes. A metric of 16 in RIP signifies that the route is unreachable. The Wireshark capture in Figure 5 illustrates how FRR-1 broadcasts this information using RIP version 2 (RIPv2) packets. The affected routes (such as 10.1.1.0/24 and 192.168.1.0/24) are marked with a metric of 16, indicating that these networks are no longer reachable via FRR-2. This broadcast occurs after the invalid timer expires, ensuring that all neighbouring routers are informed about the change in topology.

Once the invalid timer expires and the unreachable routes are broadcasted, RIP's flush timer begins counting down. The flush timer determines how long a router should retain information about an unreachable route before removing it from its routing table entirely. Typically, this process takes an additional 30 seconds after the invalid timer expires. However, in this scenario, we observe that it takes slightly less than 60 seconds for the flush process to complete, as indicated by the routing table entries shown in Figure 6. This discrepancy may be due to minor variations in timer synchronisation across different routers.

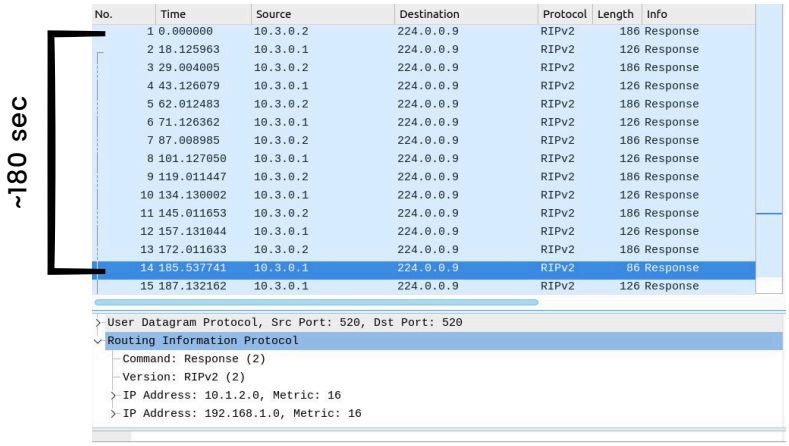


Figure 5: FRR-1 waits approximately 180 seconds before broadcasting that the link between FRR-1 and FRR-2 is unreachable with a metric of 16 (unreachable).

Figure 6: FRR-1 routing tables before and after the flush process, showing how unreachable routes are removed from the table after timers expire.

As seen in Figure 6, once the flush timer expires, FRR-1 removes all references to routes through FRR-2

from its routing table. The network now adjusts by rerouting traffic through alternative paths where possible. For example, traffic that was previously routed through FRR-2 may now be redirected via FRR-4 or other available paths depending on each router's updated distance-vector calculations.

This scenario highlights several important aspects of RIP's operation during topology changes. First, it demonstrates how RIP handles link failures through its use of timers (invalid and flush) and special metrics (such as a metric of 16 for unreachable routes). Second, it shows how RIP's convergence time can be relatively slow compared to more advanced protocols like OSPF or EIGRP, especially when relying on periodic updates and timers to propagate changes throughout the network.

In summary, this scenario illustrates how RIP responds to changes in network topology by marking routes as unreachable and eventually removing them from routing tables once timers expire. Although effective in small networks with infrequent changes, RIP's reliance on periodic updates and relatively long timers can result in slower convergence times when compared to modern routing protocols.

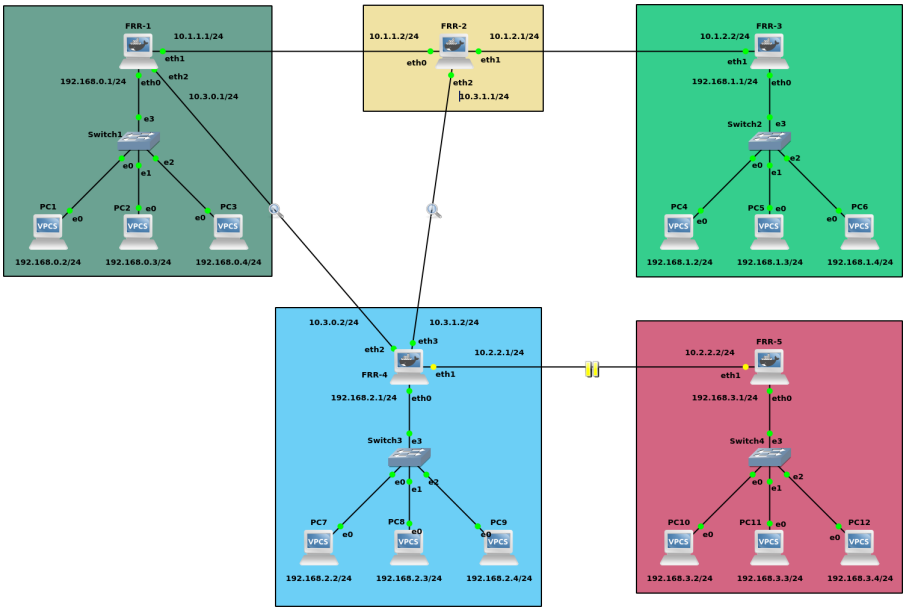


Figure 7: RIP Scenario 3 topology. Demonstrating the count-to-infinity problem.

Scenario 3: A local area network becomes unreachable

In this scenario, the link between routers FRR-4 and FRR-5 fails, causing the local area network (LAN) connected to FRR-5 to become unreachable from the rest of the topology, as shown in Figure 7. The objective of this scenario is to demonstrate how RIP handles such failures and to observe the split horizon with poison reverse mechanism in action. This mechanism is designed to prevent routing loops, a common issue in distance-vector protocols like RIP, particularly when dealing with the count-to-infinity problem.

When the link between FRR-4 and FRR-5 goes down, FRR-4 begins the process of marking the route through FRR-5 as unreachable. In RIP, routes are marked as unreachable by assigning them a metric of 16, which signifies that the destination is no longer reachable. As seen in Figure 8, it takes approximately 150 seconds from the time the link failure is detected for FRR-4 to flush the invalid route from its routing table. This delay is due to RIP's hold-down timer, which prevents routers from accepting potentially incorrect routing information during a network change. The hold-down timer ensures that routers have sufficient time to propagate accurate updates before allowing further changes to their routing tables.

To observe split horizon with poison reverse in action, we can examine how FRR-4 advertises its routes after detecting the link failure. Split horizon prevents FRR-4 from advertising back onto the interface from which it learned about the now-failed route, while poison reverse extends this by advertising the

route back with a metric of 16 (unreachable). This ensures that other routers do not attempt to use FRR-4 as a path to reach the disconnected LAN behind FRR-5. As shown in the Wireshark capture Figure 8, by packet 16, FRR-4 has already marked the route as invalid with a metric of 16. However, it is not until packet 26 that the route is fully flushed from FRR-4's routing table. This behaviour highlights how RIP's timers work together to ensure that routes are not prematurely removed or updated with incorrect information.

The impact of this link failure on traffic can be observed in Figure 9, which traces packets from FRR-3 to PC-12 before and after the link failure. Initially, before the failure (1), packets are successfully routed through FRR-5 to reach PC-12. However, immediately after the link failure (2), packets are dropped as FRR-3 attempts to route traffic through what it still believes is a valid path via FRR-5. Finally, after RIP convergence (3), once all routers have updated their routing tables and removed references to the failed link, no path to PC-12 exists and the network is seen as unreachable.

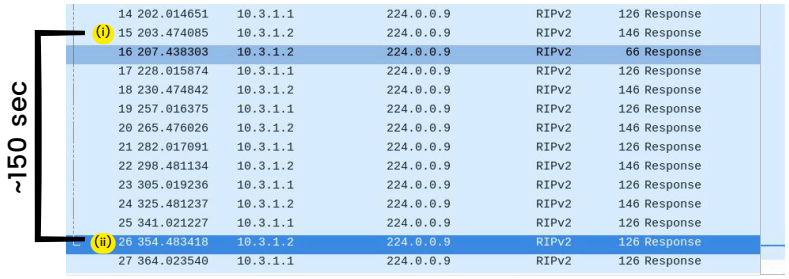


Figure 8: FRR-4: Flushed invalid link after 150 seconds when the hold-down timer expires.

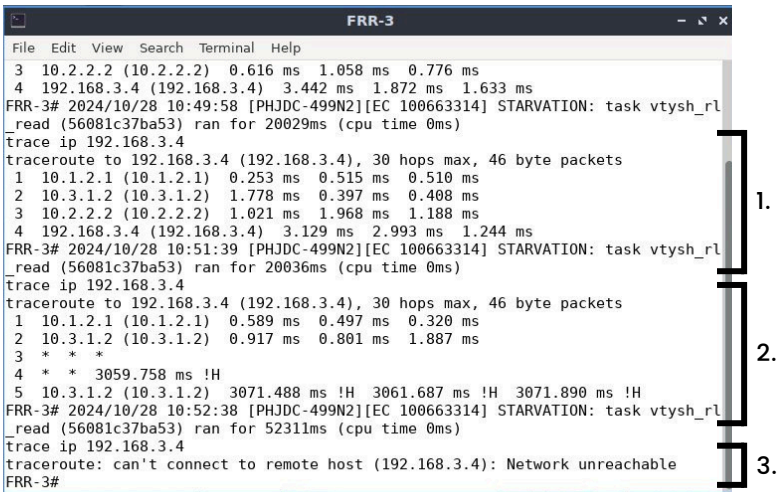


Figure 9: Tracing packets from FRR-3 to PC-12: 1) before link failure; 2) immediately after link failure; 3) after RIP table convergence.

This scenario also exposes one of RIP's fundamental limitations: its vulnerability to slow convergence times during network changes. As seen here, it takes several minutes for all routers in the network to update their routing tables and remove references to the failed link between FRR-4 and FRR-5. During this period, traffic may be misrouted or dropped entirely, leading to potential disruptions in communication. The count-to-infinity problem is another issue that can arise in such scenarios if split horizon with poison reverse is not implemented correctly. Without these mechanisms, routers could continue advertising incorrect routes indefinitely, leading to routing loops and further delays in convergence. In this case, however, split horizon with poison reverse successfully prevents such loops by ensuring that routers do not advertise routes back onto interfaces from which they were learned.

In conclusion, this scenario demonstrates how RIP handles link failures and highlights both its strengths and weaknesses. While mechanisms like split horizon with poison reverse help mitigate some of RIP's

inherent limitations, such as routing loops and count-to-infinity problems, its reliance on periodic updates and long timers results in slower convergence times compared to more advanced protocols.

ii OSPF

The topology in Figure 10 was used for all scenarios. This topology is a single autonomous system (AS) containing four Area Border Routers (ABRs) for four different areas that are connected to the backbone area, denoted by Area 0. Each area has a variety of devices attached to their respective ABRs. Additionally, a redundant connection, via a router, is located between Areas 2 and 4, where each of its interfaces is assigned to both areas. Moreover, each interface was set up to have a link cost of 5. For the OSPF protocol, we investigate its performance and its capabilities in three different scenarios.

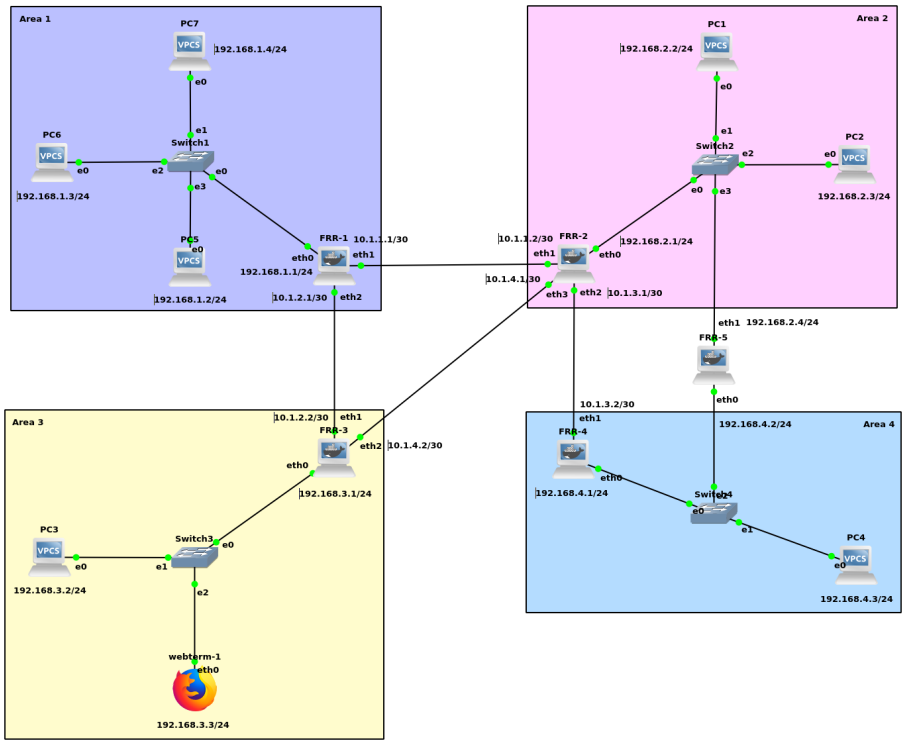


Figure 10: Multi-Area OSPF Network

Scenario 1: Flat run

This scenario analyses the route that messages take from source to destination. In this scenario, we obtain the baseline time it takes to complete the transmission and the initial calculated shortest path prior to modifying the network. This network is hierarchical topology to highlight the strengths of the OSPF routing protocol. Each router was setup to use the OSPF protocol, which calculates the shortest route to each area. Upon initialisation, the routers utilise the link-state algorithm, specifically Dijkstra's algorithm, to determine these pathways between ABRs.

Typically, OSPF routers exchange *Hello* packets to maintain connection and neighbour information, whether or not the router is directly linked to the backbone area. We observe this behaviour in the Wireshark, which captures the traffic through the link between FRR-2 and FRR-3. From Figure 11, we find that the ABRs, FRR-2 and FRR-3, send these *Hello* packets every 10 seconds to a broadcast address '224.0.0.5'. This address collects neighbour information, which is then broadcasted to all the OSPF routers. The process allows individual routers to identify which routers are reachable and available. However, these *Hello* packets must be sent before a threshold time called the dead interval. If an ABR is unable to send a *Hello* packet before the dead interval, the backbone area would declare this neighbour as unavailable. All other OSPF routers are transmitted a link-state update (LSU) and acknowledgement

(LSA) to inform of this loss and provided an updated shortest path calculation. As in Figure 12, these LSUs and LSAs are separate from the *Hello* packets and only occur when there are new changes in the link-state database (LSDB), allowing for a more efficient algorithm.

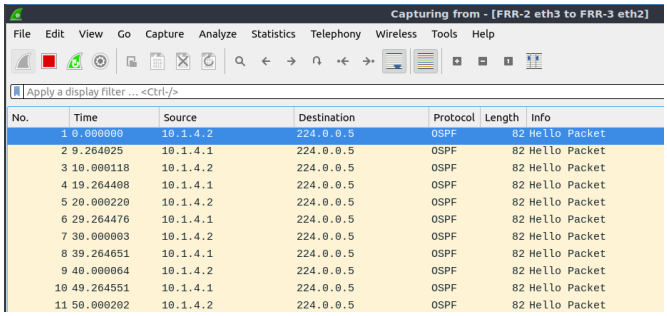


Figure 11: Hello packets seen in link between FRR-2 and FRR-3

To understand how the algorithm operates, we use the *trace* command to send a message from PC3 in Area 3 to PC4 in Area 4 and track the route this message follows. Provided the message is assumed to take the shortest path, we should observe the packet passing through the link between FRR-2 and FRR-4. However, there is a redundant link through FRR-5 that serves as an alternative path, albeit more costly. The use of the *trace* command transmits ICMP packets to ensure the reliability of a link and the reachability of an address. With this flat run, we can also determine the baseline transmission time to send a message from PC3 to PC4. Figure 13 and Figure 14 show that the it only takes less than 1 millisecond to send a packet using the OSPF routing protocol and was achieved in only 4 hops. This finding shows that the OSPF protocol achieves rapid convergence, leading to better network stability and performance. Thus, it supports network scalability while maintaining its stability and performance.

In Figure 14, we observe *Time-to-Live Exceeded* messages, followed by the successful transmission of the messages. This behaviour indicates temporary network instabilities during the initial transmission, which is a typical occurrence caused by the use of the *trace* command. However, this issue was resolved quickly, allowing the packets to be sent continuously and easily.

Scenario 2: Link between FRR-2 and FRR-4 goes down.

Scenario 2 examines the network's ability to reach an area disconnected from the backbone area. Normally, the ABRs of each area must be directly connected to the backbone area for the OSPF protocol to work successfully. For this scenario, the link between FRR-2 and FRR-4 is severed to simulate an area, specifically Area 4, getting disconnected from the rest of the network. However, there is an alternative route through FRR-5, but it is not directly linked to the backbone area. Thus, this approach allows us to investigate how the OSPF protocol handles the area disconnection and see if it will reroute through FRR-5.

The disconnection of a link or an area causes a change in the network, causing the LSDB to update using Dijkstra's algorithm. As mentioned in Scenario 1, a change in the network causes the recalculation of the

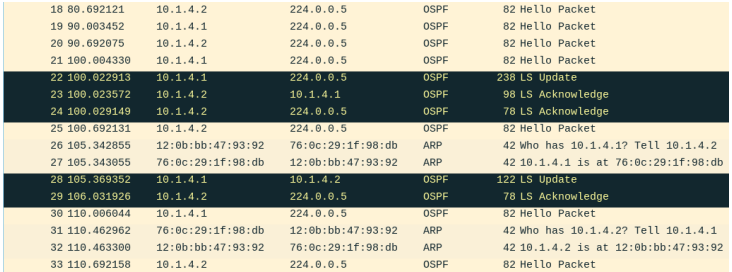


Figure 12: Wireshark updates upon network change

```
PC3> trace 192.168.4.3 -P 1
trace to 192.168.4.3, 8 hops max (ICMP), press Ctrl+C to stop
 1   192.168.3.1   0.327 ms  0.666 ms  0.124 ms
 2   10.1.4.1   0.623 ms  0.647 ms  0.729 ms
 3   10.1.3.2   1.210 ms  0.885 ms  0.532 ms
 4   192.168.4.3   2.394 ms  1.019 ms  0.813 ms
```

Figure 13: Packet route from PC3 to PC4

6	24.340924	192.168.3.2	192.168.4.3	ICMP	106 Echo (ping) request id=0xbdcc, seq=0/0, ttl=1 (no response found!)
7	24.341152	10.1.4.1	192.168.3.2	ICMP	134 Time-to-live exceeded (Time to live exceeded in transit)
8	24.342052	192.168.3.2	192.168.4.3	ICMP	106 Echo (ping) request id=0xbdcc, seq=0/0, ttl=1 (no response found!)
9	24.342306	10.1.4.1	192.168.3.2	ICMP	134 Time-to-live exceeded (Time to live exceeded in transit)
10	24.343080	192.168.3.2	192.168.4.3	ICMP	106 Echo (ping) request id=0xbdcc, seq=0/0, ttl=1 (no response found!)
11	24.343297	10.1.4.1	192.168.3.2	ICMP	134 Time-to-live exceeded (Time to live exceeded in transit)
12	24.344075	192.168.3.2	192.168.4.3	ICMP	106 Echo (ping) request id=0xbdcc, seq=0/0, ttl=2 (no response found!)
13	24.344338	10.1.3.2	192.168.3.2	ICMP	134 Time-to-live exceeded (Time to live exceeded in transit)
14	24.345448	192.168.3.2	192.168.4.3	ICMP	106 Echo (ping) request id=0xbdcc, seq=0/0, ttl=2 (no response found!)
15	24.345679	10.1.3.2	192.168.3.2	ICMP	134 Time-to-live exceeded (Time to live exceeded in transit)
16	24.346435	192.168.3.2	192.168.4.3	ICMP	106 Echo (ping) request id=0xbdcc, seq=0/0, ttl=2 (no response found!)
17	24.346596	10.1.3.2	192.168.3.2	ICMP	134 Time-to-live exceeded (Time to live exceeded in transit)
18	24.347968	192.168.3.2	192.168.4.3	ICMP	106 Echo (ping) request id=0xbdcc, seq=0/0, ttl=3 (reply in 19)
19	24.349868	192.168.4.3	192.168.3.2	ICMP	106 Echo (ping) reply id=0xbdcc, seq=0/0, ttl=62 (request in 18)
20	24.351102	192.168.3.2	192.168.4.3	ICMP	106 Echo (ping) request id=0xbdcc, seq=0/0, ttl=3 (reply in 21)
21	24.351463	192.168.4.3	192.168.3.2	ICMP	106 Echo (ping) reply id=0xbdcc, seq=0/0, ttl=62 (request in 20)
22	24.352129	192.168.3.2	192.168.4.3	ICMP	106 Echo (ping) request id=0xbdcc, seq=0/0, ttl=3 (reply in 23)
23	24.352492	192.168.4.3	192.168.3.2	ICMP	106 Echo (ping) reply id=0xbdcc, seq=0/0, ttl=62 (request in 22)
24	29.379300	10.1.4.1	224.0.0.5	OSPF	82 Hello Packet

Figure 14: Packet route from PC3 to PC4 (Wireshark)

shortest path, which stores these updates to the network's LSDB. When this occurs, all the OSPF routers are given an update on the database changes and the loss of an area using LSUs and LSAs as in Figure 12.

Similar to Scenario 1, we want to obtain the route of the transmitted message to determine if the area disconnection causes a recalculation of the shortest path. We aim to see if the message goes through FRR-5 to reach the destination address in Area 4. To achieve this, the *trace* command was utilised in the same way as in Scenario 1. Figure 16 shows that the packets are lost at FRR-2 due to its inability to find a pathway to PC4.

Prior to suspending the link, we see in Figure 15a that all areas are reachable from FRR-3, where PC3 links to. As long as the area is within reach of the ABR of Area 3, FRR-3, PC3 can send messages to any device in the AS. By severing the connection, we then find that the gateway address '192.168.4.0', which is used for Area 4, is no longer reachable. This means that the disconnection of the direct link causes a loss of a pathway from FRR-3 to PC4, as long as the area has no immediate connection to the backbone area. This behaviour highlights the weakness of the routing protocol in reaching a 'disconnected' area.

```
FRR-3# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, F - PBR,
       f - OpenFabric,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

O>* 10.1.1.0/30 [110/10] via 10.1.2.1, eth1, weight 1, 00:53:40
  *                      via 10.1.4.1, eth2, weight 1, 00:53:40
O   10.1.2.0/30 [110/5] is directly connected, eth1, weight 1, 00:53:45
C>* 10.1.2.0/30 is directly connected, eth1, 00:53:51
O>* 10.1.3.0/30 [110/10] via 10.1.4.1, eth2, weight 1, 00:53:40
O   10.1.4.0/30 [110/5] is directly connected, eth2, weight 1, 00:53:46
C>* 10.1.4.0/30 is directly connected, eth2, 00:53:51
O>* 192.168.1.0/24 [110/10] via 10.1.2.1, eth1, weight 1, 00:53:40
O>* 192.168.2.0/24 [110/10] via 10.1.4.1, eth2, weight 1, 00:53:40
O   192.168.3.0/24 [110/5] is directly connected, eth0, weight 1, 00:53:46
C>* 192.168.3.0/24 is directly connected, eth0, 00:53:51
O>* 192.168.4.0/24 [110/15] via 10.1.4.1, eth2, weight 1, 00:10:50
```

(a) Flat Run

```
FRR-3# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, F - PBR,
       f - OpenFabric,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

O>* 10.1.1.0/30 [110/10] via 10.1.2.1, eth1, weight 1, 01:05:07
  *                      via 10.1.4.1, eth2, weight 1, 01:05:07
O   10.1.2.0/30 [110/5] is directly connected, eth1, weight 1, 01:05:12
C>* 10.1.2.0/30 is directly connected, eth1, 01:05:18
O>* 10.1.3.0/30 [110/10] via 10.1.4.1, eth2, weight 1, 01:05:07
O   10.1.4.0/30 [110/5] is directly connected, eth2, weight 1, 01:05:13
C>* 10.1.4.0/30 is directly connected, eth2, 01:05:18
O>* 192.168.1.0/24 [110/10] via 10.1.2.1, eth1, weight 1, 01:05:07
O>* 192.168.2.0/24 [110/10] via 10.1.4.1, eth2, weight 1, 01:05:07
O   192.168.3.0/24 [110/5] is directly connected, eth0, weight 1, 01:05:13
C>* 192.168.3.0/24 is directly connected, eth0, 01:05:18
```

(b) Scenario 2

Figure 15: Reachable addresses from FRR-3 before and after network change

This finding raises the question if the OSPF routing protocol is able to reroute packets successfully after a network disturbance. To investigate this ability, we use the *trace* command to send a packet from PC6 in Area 1 to PC4 in Area 4. As observed, there are two possible paths to achieve this, with the route through FRR-1, FRR-2, then FRR-4 as the preferred path. However, the severance of the link between FRR-1 and FRR-2 should trigger a LSDB update, causing the packets to reroute through the other path. We observe this exact behaviour in Figure 17. Given that the alternative path has a higher cost, we see that the time it take for the packets to be sent is longer than the original shortest path.

Scenario 3: Latency between FRR-2 and FRR-3

```
PC3> trace 192.168.4.3 -P 1
trace to 192.168.4.3, 8 hops max (ICMP), press Ctrl+C to stop
 1   192.168.3.1   0.163 ms  0.209 ms  0.448 ms
 2   10.1.4.1   0.717 ms  0.582 ms  0.721 ms
 3     *  *  *
 4     *  *  *
 5     *  *  *
 6     *  **10.1.4.1   63.642 ms (ICMP type:3, code:1, Destination host unreachable)
```

Figure 16: Packet route after area disconnection

```
PC6> trace 192.168.4.3 -P 1
trace to 192.168.4.3, 8 hops max (ICMP), press Ctrl+C to stop
 1   192.168.1.1   0.160 ms  0.155 ms  0.227 ms
 2   10.1.1.2   0.223 ms  0.306 ms  0.429 ms
 3   10.1.3.2   0.793 ms  0.531 ms  0.573 ms
 4   192.168.4.3   2.001 ms  1.103 ms  0.914 ms

PC6> trace 192.168.4.3 -P 1
trace to 192.168.4.3, 8 hops max (ICMP), press Ctrl+C to stop
 1   192.168.1.1   0.215 ms  0.331 ms  0.156 ms
 2   10.1.2.2   0.683 ms  0.358 ms  0.539 ms
 3   10.1.4.1   0.932 ms  0.960 ms  0.913 ms
 4   10.1.3.2   1.118 ms  1.159 ms  0.963 ms
 5   192.168.4.3   2.780 ms  0.973 ms  1.094 ms
```

Figure 17: Packet route from PC6 to PC4 before and after network disturbance

This scenario explores the maximum tolerable latency in the calculated shortest path from the source address to the destination address. We aim to see if these latencies or delays can cause a link-state update, which forces the packets to reroute through a different path. Typically, the OSPF routing protocol only uses the cost of the path as its main metric, neglecting the additional latencies. This approach tests how much delay the network can handle before encountering issues. To test this, The latency in the link between FRR-2 and FRR-3 is incremented by 100 milliseconds (ms), starting from 0 ms.

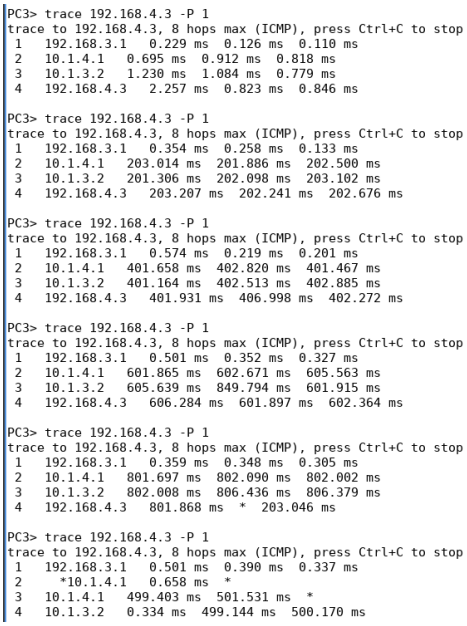
In Figure 15, we observe that a single destination address may have multiple potential pathways, especially if these paths have the same costs. This concept allows for packets to be divided between these equal cost pathways, ensuring its efficient and reliable transmission and improved traffic management. However, from Figure 10, we find that these multiple pathways have different cost. Thus, we focus on the effect of latency on the delivery of the packets from source to destination.

To obtain these results, we utilise the *trace* and *ping* commands to see where the route breaks down due to the latency. Figure 18 shows the transmission time and the number of successful transmissions for delays of 0 to 500 ms, giving six outcomes. The *ping* command sends a series of packets from the source to the destination address. We observe that the transmission time increases by 200 ms as we increment the delay by 100 ms, indicating that the transmission time takes twice the time of the delay. Once the delay goes over 300 ms, we find instabilities during the transmission of the packet, as seen in Figure 18a. However, in Figure 18b, we are given a *timeout* outcome when the delay is at 300 ms, signifying its inability to send a packet successfully. This finding tells us that the transmission of packets are less reliable once the latency in the path reaches 300 ms. Thus, the maximum tolerable delay that ensures reliable and successful packet transmission is 200 ms.

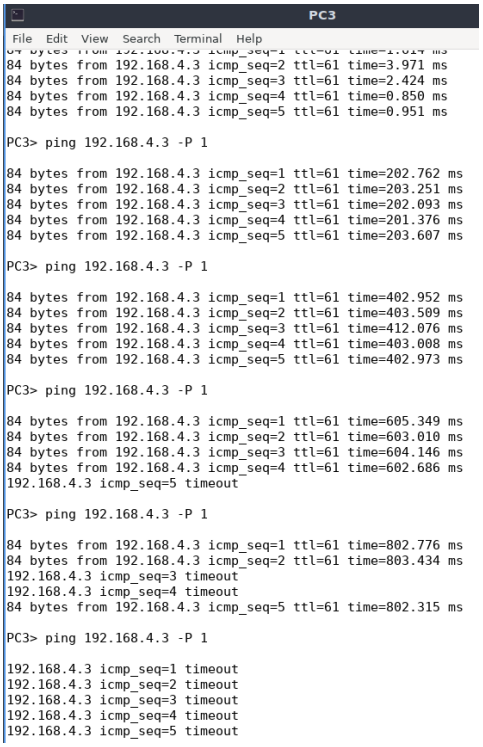
Additionally, we observe that the route the packet takes, per the OSPF protocol, does not change despite the instabilities that the delay causes. Figure 18b proves that the OSPF protocol does not consider the latencies and delays in the network when calculating for the shortest path. This finding reveals the concept of equal-cost multi-path (ECMP) as a potential weakness of the OSPF routing protocol. ECMP can be a powerful feature for networks with multiple paths of the same cost. However, if it is presented with a network without multiple routes with the same cost, it is vulnerable to disturbances in the network, causing instabilities or unreliable transmissions.

iii EIGRP

Three simple topologies were constructed to demonstrate the basic operation of EIGRP, how routers maintain neighbour, topology and routing tables, how feasible distances and reported distances are used



(a) Packet path



(b) Ping success

Figure 18: Sending packets with latencies from 0 ms to 500 ms

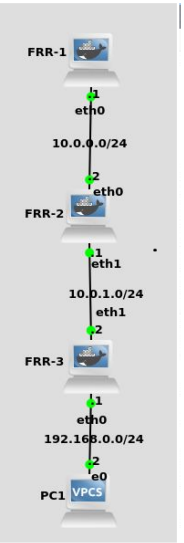
to decide on successors and how the feasibility condition ensures that only loop-free routes can become feasible successors.

Topology A. The first topology is a simple linear arrangement of three routers configured with EIGRP and a virtual PC connected in series (Figure 19). Subnetwork addresses and the IP addresses of each interface are shown. On the right of Figure 19 are (from top to bottom) the routing table, neighbour table, list of interfaces and topology table for FRR-2.

The *interface table* lists the interfaces associated with FRR-2 and each interface's associated characteristics. By default, the bandwidth of the interface is set to 1000000 kbps, delay set to 100 μs (it is listed in tens of microseconds), Hello interval set to 5 seconds and Hold time set to 15 seconds.

The *neighbour table* lists all of FRR-2's neighbour's: each neighbour has an index number (H number or 'Handle'), the next-hop router and interface connecting it, the remaining hold time (if another EIGRP packet is not received from this neighbour within this time, the adjacency will be dropped), the uptime (how long a neighbour has been up), smooth round trip time (time in milliseconds from sending EIGRP packets to a neighbour then receiving an ACK in return), retransmission timeout (time in milliseconds router will wait before retransmitting a packet if an ACK is not received), a Q count (the number of update/query/reply packets in the queue awaiting transmission) and sequence number of the last update, query or reply received from the neighbour. Q count is ideally 0, otherwise it is an indication of the congestion on the network.

The *topology table* lists all the subnets that are potentially reachable from FRR-2, what state each route is in (P stands for passive, meaning that the route is available and unchanging, not needing recalculation), how many routes have the shortest path (successors), the feasible distance to each subnet and the next-hop router through which the path must travel to reach it. Using the metric calculation formula and default values of 1 for K_1 and K_3 and 0 for the rest, the default bandwidth of 100000 and delay of 10 is consistent with a metric of $(\frac{10^7}{100000} + 10) \cdot 256 = 28160$ to reach the subnets immediately either side of FRR-2. To reach the PC, FRR-3 advertises to FRR-2 a reported distance of 28160. The feasible




```
FRR-2
File  Edit  View  Search  Terminal  Help
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, F - PBR,
       f - OpenFabric,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

E   10.0.0.0/24 [90/28160] is directly connected, eth0, weight 1, 00:00:12
C>* 10.0.0.0/24 is directly connected, eth0, 00:00:36
E   10.0.1.0/24 [90/28160] is directly connected, eth1, weight 1, 00:00:04
C>* 10.0.1.0/24 is directly connected, eth1, 00:00:28
E>* 192.168.0.0/24 [90/30720] via 10.0.1.2, eth1, weight 1, 00:00:02
FRR-2# write
Note: this version of vtysh never writes vtysh.conf
Building Configuration...
Integrated configuration saved to /etc/frr/frr.conf
[OK]
2024/10/27 13:04:50 WATCHFRR: [WFP93-1D146] configuration write completed with exit code 0
FRR-2# show ip eigrp neighbor

EIGRP neighbors for AS(1)

H   Address           Interface           Hold   Uptime   SRTT   RTO   Q     Seq
                                          (sec)           (ms)        Cnt   Num
0   10.0.0.1          eth0                11     0        0      2     0     2
0   10.0.1.2          eth1                10     0        0      2     0     2
FRR-2# show ip eigrp interface

EIGRP interfaces for AS(1)

Interface        Bandwidth  Delay      Peers  Xmit Queue   Mean    Pacing Time    Multicast    Pending  Hello    Holdtime
                                              Un/Reliable  SRTT    Un/Reliable    Flow Timer   Routes
eth0             100000     10         1      0 / 0         0       0              0            0        5        15
eth1             100000     10         1      0 / 0         0       0              0            0        5        15
FRR-2# show ip eigrp topology

EIGRP Topology Table for AS(1)/ID(10.0.1.1)

Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply
       r - reply Status, s - sia Status

P  10.0.0.0/24, 1 successors, FD is 28160, serno: 0
       via Connected, eth0
P  10.0.1.0/24, 1 successors, FD is 28160, serno: 0
       via Connected, eth1
P  192.168.0.0/24, 1 successors, FD is 30720, serno: 0
       via 10.0.1.2 (30720/28160), eth1
FRR-2#
```

Figure 19: On the left, topology A: three routers and a virtual PC arranged in series. On the right, routing, neighbour, interface and topology tables shown in the console for FRR-2.

distance to reach the PC has an additional delay of 10 to pass through FRR-3 with effectively unlimited bandwidth for an additional distance of 2560 and a total feasible distance of $28160 + 2560 = 30720$.

Finally, using the topology table the *routing table* lists the next-hop router of the best route (*successor*) to each subnet and the feasible distance to reach it. 90 is the administrative distance for routes learned from EIGRP. Administrative distance is relevant in a situation where routers are running multiple routing protocols and different routes are being learned from different protocols — more on this in the Discussion.

Wireshark was used to capture EIGRP packets being sent along the link between FRR-1 and FRR-2. Hello packets are being sent from each router to the multicast destination of 244.0.0.10 every 5 seconds.

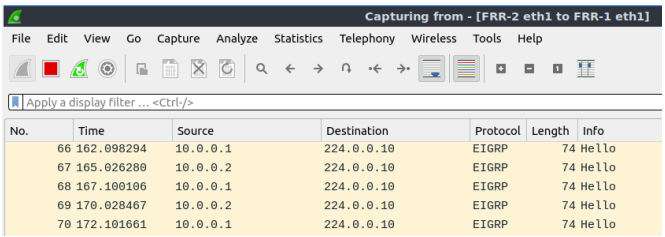


Figure 20: Hello packets from FRR-1 and from FRR-2 sent every 5 seconds.

Generally speaking, when a successor route fails, the entry for that route in the topology table goes from passive to active. If no feasible successor has been nominated, Query packets are sent on all interfaces other than that of the successor (as a multicast). All Query packets should in turn generate corresponding reply packets (unicast), which themselves contain either the metric for a new route to the lost subnet or a metric with a value of infinity for the lost subnet. The receipt of both Query and Reply packets also generate Acknowledgement packets (unicasts).

When the link between FRR-2 and FRR-3 was suspended, the Wireshark window monitoring that link stopped displaying any new transmitted packets (as expected). However, we would expect FRR-2 then to send a Query packet to FRR-1 to see whether FRR-1 has a route to the lost subnet 192.168.0.0. No such query packets were seen in the Wireshark capture of the link between FRR-2 and FRR-1 (which we will refer to later in the Discussion as Bug #1, since this is potentially an issue with implementing a proprietary protocol EIGRP on FRR, an open source routing package). The same situation was observed when the router FRR-3 itself was taken offline.

When the link between FRR-2 and FRR-3 was restored, we observed appropriate behaviour where Update packets were being sent on that link (by unicast), along with associated Acknowledgements in reply (Appendix Figure 30). The same behaviour is seen when changes to link metrics (ie the delay) were configured.

A similar topology (topology A*), this time with two connected routers and two PCs on either side, all linked in series, was used to investigate how pings and traces might be used to investigate time taken to reach a destination (Figure 21).

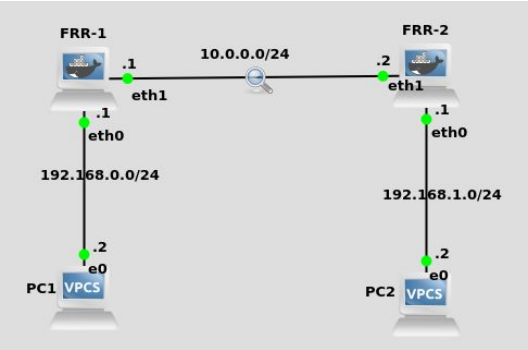


Figure 21: A version of the same topology with a PC, two routers and another PC connected in series.

Pings involve the source computer sending a series of four request packets to the target destination. Upon receipt, the destination PC will send back four reply packets (Appendix Figure 31). The trace command allows us to track the passage of one packet through the network to a destination subnet (up to a maximum of eight hops). By default, the transport layer protocol is UDP but the destination subnet in this case is unreachable. By specifying an alternative transport protocol ('-P 1' for ICMP and '-P 6' for TCP), the destination becomes reachable (Appendix Figure 32).

We then attempted to change the metric by adjusting the bandwidth, but in almost all but a couple of instances, this produced an I/O error in the vtysh shell, severing the connection to the EIGRP daemon (Bug #2, Figure 22).

The I/O error was able to be avoided by simply configuring changes to only the delay in order to change the EIGRP metric values (Appendix Figure 29). Corresponding changes then appear to the reported and feasible distances in the topology table (Figure 23). Another issue with implementing EIGRP using FRR can be seen in the Figure 23. Duplicate routes (extra routes with metrics associated with delay values totally unclear in origin!) appear in the topology table, despite each route only having one definite FD. The extra routes also appear in the routing table (Bug #3).

Ping and trace packets were no longer able to find the destination once link metrics had been changed (Appendix Figure 33). This is potentially another manifestation of Bug #3, where packets might not be able to reach the destination if the routing table for the destination contains multiple entries. When the delay was changed back to the default value, ping and trace were still unable to find the destination, which is consistent with the hypothesis that this is happening because old routes cannot be removed from the topology table (Appendix Figure 34).

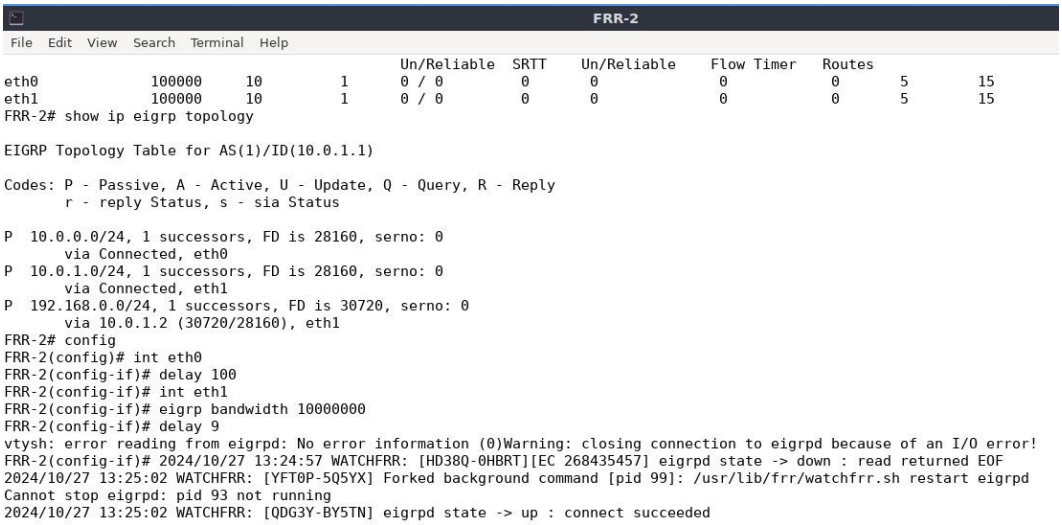


Figure 22: An I/O error produced by a change configured in the bandwidth of an interface.

```
EIGRP Topology Table for AS(1)/ID(10.0.0.1)

Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply
       r - reply Status, s - sia Status

P  10.0.0.0/24, 2 successors, FD is 28160, serno: 0
       via Connected, eth0
       via Connected, eth0
P  10.0.1.0/24, 2 successors, FD is 30720, serno: 0
       via 10.0.0.2 (30720/28160), eth0
       via 10.0.0.2 (30720/5120), eth0
P  192.168.0.0/24, 2 successors, FD is 33280, serno: 0
       via 10.0.0.2 (33280/30720), eth0
       via 10.0.0.2 (33280/7680), eth0
FRR-1# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, F - PBR,
       f - OpenFabric,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

E   10.0.0.0/24 [90/28160] is directly connected, eth0, weight 1, 00:27:35
                          is directly connected, eth0, weight 1, 00:27:35
C>* 10.0.0.0/24 is directly connected, eth0, 00:45:30
E>* 10.0.1.0/24 [90/30720] via 10.0.0.2, eth0, weight 1, 00:02:04
                           via 10.0.0.2, eth0, weight 1, 00:02:04
E>* 192.168.0.0/24 [90/33280] via 10.0.0.2, eth0, weight 1, 00:02:03
                              via 10.0.0.2, eth0, weight 1, 00:02:03
FRR-1#
```

Figure 23: Changes to the topology table as a result of introducing a delay to the link between FRR-2 and FRR-3. Two routes are listed in the topology table, and also appear in the routing table.

Topology B. The second topology features two PCs joined by a network for five routers, offering three distinct paths between the PCs (Figure 24). This is useful for then demonstrating how the *feasibility condition* applies to the nomination of backup paths (*feasible successors*).

Just as before, using default bandwidth and delay for all links, ping and trace can be used to show how packets can go from one PC to the other (Appendix Figure 35).

While not demonstrated here, we also note that EIGRP can have multiple equal cost successor routes (which would be the case in this topology, prior to metrics being changed from default values) and EIGRP will load balance the traffic through each of those equal cost paths.

We then introduced changes to the distance metric for each link in order to show how the feasibility condition works. The network topology annotated with updated metrics is shown in Figure 24 (with some of the corresponding new delay values, at least for FRR-1, in Appendix Figure 36). Interestingly, trace did work to show the passage of a packet travelling through the network along the shortest path to reach 192.168.1.2 (Appendix Figure 37), unlike in topology A where ping and trace stopped working after the metrics were changed.

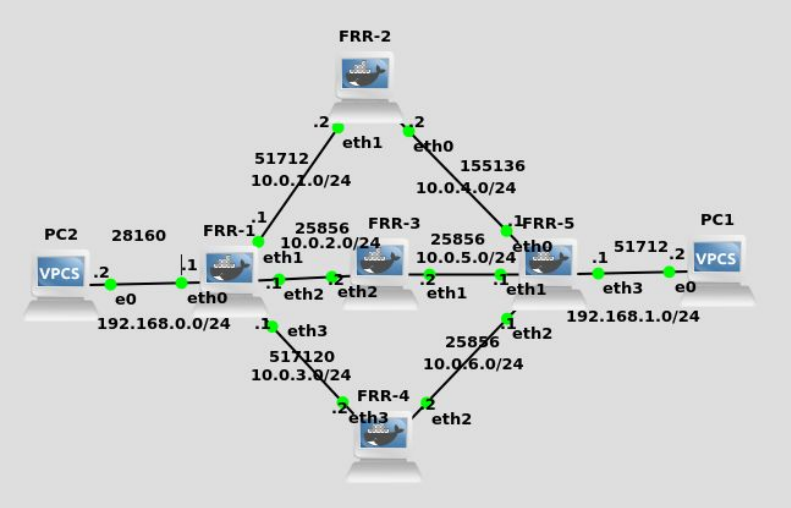


Figure 24: Topology B with link annotations with weights. Initially all the links have equal weights. Only later are updated metrics configured to produce the weights displayed.

In the topology table of FRR-1, which lists all the routes learned from its neighbours using EIGRP, let us focus on how FRR-1 would reach subnet 192.168.1.0 (Figure 25, full set of neighbour and topology tables in Appendix Figure 38).

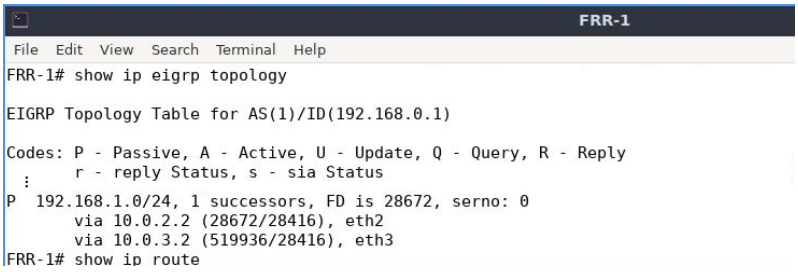


Figure 25: FRR-1's topology table entry for 192.168.1.0.

FRR-5 can reach 192.168.1.0 with a feasible distance (FD) of **2560** (simply a delay of 10 to go through FRR-5 itself, and essentially unlimited bandwidth). FRR-5 will advertise this distance to each of FRR-2, FRR-3 and FRR-4, and each of those routers then enters into their respective topology tables that the best path to reach 192.168.1.0 is through FRR-5 with a reported distance (RD) of 2560.

Intermediate routers calculate the cost of the route through FRR-5 and advertise this to FRR-1:

- FRR-2 can reach 192.168.1.0 by going through FRR-5 with a FD of 2560 (the RD from FRR-5) + 155136 (the cost of the link between FRR-2 and FRR-5) = **157696**. This then becomes the RD advertised from FRR-2 to FRR-1.
- FRR-3 can reach 192.168.1.0 by going through FRR-5 with a FD of 2560 (the RD from FRR-5) + 25856 (the cost of the link between FRR-2 and FRR-5) = **28416**. This then becomes the RD advertised from FRR-3 to FRR-1.
- FRR-4 can reach 192.168.1.0 by going through FRR-5 with a FD of 2560 (the RD from FRR-5) + 25856 (the cost of the link between FRR-4 and FRR-5) = **28416**. This then becomes the RD advertised from FRR-4 to FRR-1.

Finally we come to consider how FRR-1 populates its topology table.

- FRR-1 can reach 192.168.1.0 by going through FRR-2 with a FD of 157696 (the RD from FRR-2) + 26112 (delay of 102 in the link between FRR-1 and FRR-2) = **183808**.
- FRR-1 can also reach 192.168.1.0 by going through FRR-3 with a FD of 28416 (the RD from FRR-3) + 256 (delay of 1 in the link between FRR-1 and FRR-3) = **28672**.
- FRR-1 can also reach 192.168.1.0 by going through FRR-4 with a FD of 28416 (the RD from

FRR-4) + 491520 (delay of 1920 in the link between FRR-1 and FRR-3) = **519936**.

The minimum FD out of all of these **28672**, the cost of the route through FRR-3. This becomes the successor route, and is installed in FRR-1's routing table, so that packets destined for 192.168.1.0 are forwarded to FRR-3 as the next-hop router (Figure 26).

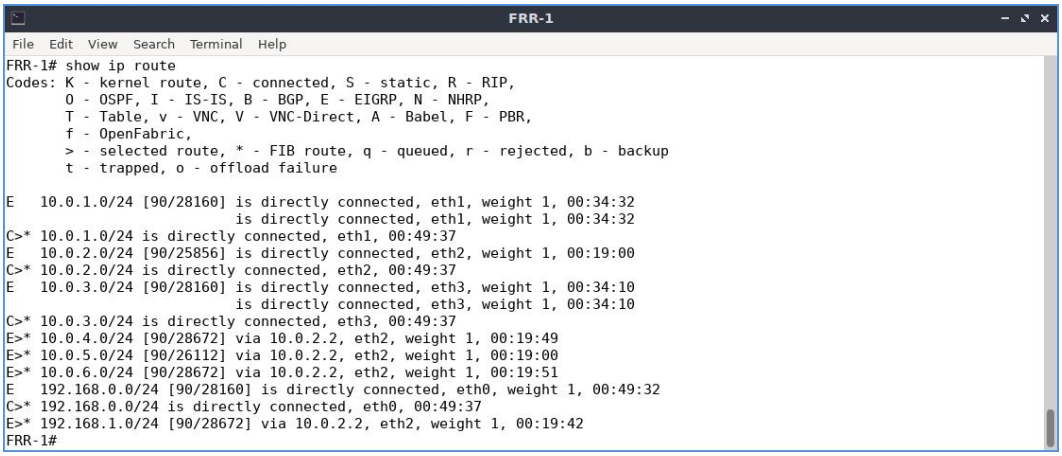


Figure 26: FRR-1's routing table.

Recalling that a feasible successor is a backup route precalculated alongside the best route, to be used in case the successor fails, we can now consider whether the other two (higher-cost) paths might be used as backup routes. The feasibility condition states that a route may only become a feasible successor if its RD is *less than* the FD of the successor route. Even though the next shortest FD (183808) to 192.168.1.0 is the route through FRR-2, this route is not eligible to become a feasible successor since the RD from FRR-2 (157696) is *more than* the successor FD (28672). The route through FRR-4 (with the longer FD of 519936) becomes the feasible successor, eligible because the RD from FRR-4 (28416) is *less than* the successor FD (28672). Only successors and feasible successors appear in topology tables with other routes not appearing, and this is what we see in Figure 25.

The feasibility condition prevents the second best path from being used as a backup path here, but this does not mean that a route that violates the condition won't eventually be used as a feasible successor or even a successor, after recalculation. A DUAL route recalculation is triggered whenever the current router successor fails or link to that router goes down, and any feasible successor is used as the successor path only for as long as it takes for that recalculation process to be completed. The backup is used only temporarily, until it is either established to be the definitive shortest route, or replaced by the shortest route found by the recalculation process.

We attempted to see what happens when the middle path (successor) fails, both by suspending FRR-3 and by suspending the link between FRR-1 and FRR-3. We would expect that the feasible successor would be used while a query process is initiated — the initial backup path would be through FRR-4 but then settle on the path through FRR-2. However, in both cases, we observed errors: while the neighbour table for FRR-1 correctly omitted the interface for FRR-3 (Appendix Figures 39 and 41), routes going through FRR-3 still appeared in both the topology and routing tables for FRR-1, and not being removed when they should be (Bug #3 again, Appendix Figures 40 and 42). No query packets were sent from FRR-1 to FRR-2 or FRR-4. What happened with the trace command varied either running in an infinite loop or finding that the destination host was unreachable (Appendix Figures 43 and 44).

The final topology C will show how the feasibility condition prevents loop-free backup routes.

Topology C. The final topology features three routers connected in a loop, linked to another router linked to a PC (Figure 27). We now examine the topology table of FRR-3 to analyse how it is that the feasibility condition guarantees loop-free paths. To avoid Bug #3 (shown in Appendix Figure 45),

instead of configuring the metric in FRR, let us simply imagine that each link has an arbitrary cost of 10.

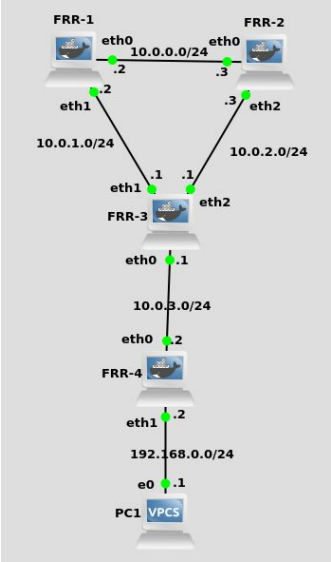


Figure 27: Topology C: loop arrangement of three routers connected to another router, which is connected to a PC, each link having an equal metric.

To illustrate loop-free paths, we must consider what FRR-1 and FRR-2 are advertising to FRR-3.

- First, FRR-4 advertises to FRR-3 its feasible distance of 10 to reach the target subnet 192.168.0.0. At FRR-3, this becomes the RD from FRR-4, so that FRR-3's FD to reach the target subnet is 10 (the FD from FRR-4) + 10 (cost of link between FRR-3 and FRR-4) = 20.
- FRR-3 then advertises its FD of 20 to FRR-1 and FRR-2 (but not to FRR-4 because of the operation of the split horizon principle). FRR-1 receives from FRR-3 a RD of 20, and so its FD to reach the target subnet is 30, going through FRR-3. FRR-2 receives from FRR-3 a RD of 20, so its FD to reach the target subnet is also 30, going through FRR-3.
- FRR-1 and FRR-2 then advertise to each other their routes to the target subnet, with RDs of 30.
- For FRR-1, apart from the route through FRR-3, an alternative route is through FRR-2, which would have a FD of 40 (RD from FRR-2 30 + 10 the cost of the link between FRR-1 and FRR-2).
- For FRR-2, apart from the route through FRR-3, an alternative route is through FRR-1, which would have a FD of 40 (RD from FRR-2 30 + 10 the cost of the link between FRR-1 and FRR-2).
- In the topology tables of both FRR-1 and FRR-2, the feasibility condition operates to prevent these alternate paths from becoming feasible successors (since the RD 30 is *not less than* the FD 30 of the successor

Without the feasibility condition, for FRR-1, the route through FRR-2 would become the feasible successor, and FRR-1 would use the route through FRR-2 as a backup route to get to the target subnet in case the route through FRR-3 fails. Similarly, for FRR-2, the route through FRR-1 would become the feasible successor, and FRR-2 would use the route through FRR-1 as a backup route to get to the target subnet in case the route through FRR-3 fails.

Imagine now that FRR-3 goes down, and the successor route fails. DUAL would initiate a query process between FRR-1 and FRR-2, each router asking the other if they have a different route to reach the target subnet. Both would reply 'no route exists' (by providing an infinite metric) and eventually the target subnet would be declared lost. However, in the meantime prior to this route recalculation process concluding, each router would use the other as the next-hop router (since these routes were feasible successors) and packets destined for the target subnet would be sent back and forth between the two routers in an endless loop.

This example illustrates how the feasibility condition prevents loop-free routes from being adopted by stopping routes that potentially contain a loop from becoming feasible successors in the first place. It

also highlights why it is important that the condition insists that the RD of the feasible successor is *strictly less than* the FD of the successor, rather than less than or equal to.

4 Discussion (Chris, Kelly, Mukul)

In this section we compare the three routing protocols and discuss their relative strengths and weaknesses, under different headings of convergence and scalability, metrics, timers, load balancing, ease of implementation and best use cases. Our experimental results using GNS3 simulations provide valuable insights into the performance characteristics of RIP, OSPF, and EIGRP routing protocols. Each protocol demonstrated distinct strengths and weaknesses in various network scenarios, aligning with their theoretical foundations discussed in Section 2. We will also discuss some limitations, particularly when using the FRR package to implement EIGRP. Finally, avenues for ongoing exploration are also proposed for taking this project further.

a Convergence and Scalability

Our experimental results, as detailed in Section 4, provide valuable insights into the convergence and scalability characteristics of EIGRP, OSPF, and RIP, aligning with the theoretical foundations discussed in Section 2. These findings not only corroborate our initial expectations but also offer practical validation of the protocol behaviours outlined in the theoretical background.

Had FRR been able to implement EIGRP correctly, we would expect to have been able to demonstrate the fastest convergence times in our experiments, particularly in scenarios involving link failures and network changes. This is attributable to EIGRP's proactive approach to route calculation, which underlies the Diffusing Update Algorithm (DUAL), explained in Section 2.c.iii. As a distance vector protocol, EIGRP benefits from sharing information only between neighbouring routers, reducing the overall communication overhead. This aligns with the discussion in Section 2.d.iii, which highlights EIGRP's efficient use of network resources. The scalability of EIGRP was regretfully also unable to be demonstrated in simulation, but is said to be excellent, especially if features such as route summarisation and stub routing are used for more extensive networks (CiscoPress, 2011). Unfortunately stub routing was not included in the RFC 7868 open standard (Ergun, 2019). Unlike OSPF, which requires careful area partitioning for optimal performance in large topologies, EIGRP demonstrates excellent scalability without such partitioning requirements. This advantage stems from EIGRP's distributed computation model, where the convergence process is spread across multiple routers, each handling a manageable subset of the network topology.

OSPF, while not as fast as EIGRP, exhibited moderate convergence times and proved to be more scalable than initially anticipated, especially in larger network topologies. This performance, observed in our experiments detailed in Section 4.b.ii, aligns with OSPF's nature as a link-state protocol, which provides each router with a comprehensive view of the network topology, as described in Section 2.d.ii. The use of Dijkstra's algorithm for path calculation, with its computational complexity of $O(n^2)$, did introduce some overhead in larger networks, consistent with the theoretical discussion in Section 2.b.i. However, our experiments confirmed that OSPF's hierarchical design, particularly its ability to segment the network into areas, effectively mitigated this scalability challenge. Interestingly, we observed that OSPF's global knowledge of the network topology, while computationally intensive, enabled more informed routing decisions in complex scenarios. This advantage was particularly evident in situations where optimal path selection required a holistic view of the network, supporting the theoretical strengths of link-state protocols outlined in Section 2.c.ii.

RIP exhibited the slowest convergence times among the three protocols, consistent with the limitations described in Section 2.d.i and demonstrated in our experimental results in Section 4.b.i. Our simulations confirmed RIP's vulnerability to the count-to-infinity problem, despite its implementation of split horizon and poisoned reverse techniques. The reliance on periodic updates and the limitation of a maximum

hop count of 16 significantly impacted RIP's ability to adapt quickly to network changes, especially in larger topologies. These observations align with the theoretical limitations of RIP as a basic distance vector protocol discussed in Section 2.c.i.

The performance disparity between RIP and the other protocols became more pronounced as network complexity increased, validating the scalability concerns raised in Section 2.b.ii and clearly demonstrated in our experimental results across different network sizes in Section 4. In smaller, more stable network environments, RIP's simplicity offered some advantages in terms of ease of configuration and low resource utilisation, as predicted in the theoretical background. However, its slower convergence and limited scalability make it less suitable for large or rapidly changing networks, where would EIGRP and OSPF clearly outperform.

Our results provide empirical evidence supporting the theoretical strengths and weaknesses of each protocol as outlined in Section 2 (at least with respect to OSPF and RIP). EIGRP's combination of rapid convergence and excellent scalability positions it as a top performer, especially in large and dynamic networks. OSPF, while slightly slower in convergence, offers robust performance and scalability when properly configured with area partitioning, aligning with both theoretical expectations and our practical findings. RIP, despite its limitations, remains a viable option for smaller, more stable network environments where simplicity is prioritised over rapid adaptation to changes, as evidenced by our experimental results in simpler network topologies. These findings not only validate the theoretical framework presented earlier but also offer practical insights into the real-world behaviour of these routing protocols under various network conditions and sizes.

b Metric Calculation and Path Selection

Our experiments revealed significant differences in how each protocol calculates metrics and selects optimal paths, reflecting the theoretical foundations discussed in Section 2.b.i and 2.d.

EIGRP's composite metric, as described in Section 2.d.iii, allowed for more nuanced path selection compared to RIP and OSPF. This sophisticated approach was evident in scenarios where link characteristics varied, demonstrating EIGRP's ability to make more intelligent routing decisions. Our simulations showed that EIGRP's consideration of bandwidth, delay, reliability, and load provides a more comprehensive approach to path selection, particularly in networks with diverse link characteristics. OSPF's cost-based metric, derived from link bandwidth as explained in Section 2.d.ii, provided a balance between simplicity and effectiveness. Our experiments confirmed the protocol's ability to prioritise higher bandwidth paths, aligning with its theoretical design. We observed that OSPF's cost-based metric offers network administrators fine-grained control over traffic flow, a feature particularly useful in complex enterprise networks. RIP's hop count metric, discussed in Section 2.d.i, lead to suboptimal path selections, especially in networks with varying link speeds. This observation reinforces the theoretical limitations of RIP in complex network environments. Our simulations demonstrated that RIP consistently chose paths based solely on the number of hops, potentially overlooking faster routes with more intermediate nodes.

c Protocol Overhead and Resource Utilisation

The experimental results regarding protocol overhead and resource utilisation closely matched the theoretical expectations outlined in Section 2. RIP's periodic full table updates, as described in Section 2.d.i, generate significant overhead in larger networks. This behaviour aligns with the theoretical discussion on RIP's scalability limitations. We observed that RIP's simplicity, while beneficial for easy configuration, can be problematic in larger networks or those with frequent changes due to the increased network traffic from periodic updates. OSPF's link-state advertisements (LSAs), discussed in Section 2.d.ii, generally produced less overall traffic than RIP in stable networks. However, we observed temporary spikes in network traffic during topology changes, consistent with OSPF's flooding mechanism. Our experiments confirmed that OSPF's hierarchical area structure effectively contained this overhead in larger networks. EIGRP would likely demonstrate the most efficient use of network resources among the three protocols,

due to its partial update mechanism described in Section 2.d.iii. EIGRP's query-based approach to gathering routing information would mean minimal overhead, even in dynamic network environments. This efficiency makes EIGRP particularly suitable for networks where bandwidth conservation is a priority.

d Timers and Load Balancing

Our experiments provided insights into the impact of timers and load balancing capabilities of each protocol, as discussed in Section 2.d. We observed that OSPF's dead interval timer effectively managed routing stability, while RIP's multiple timers (update, invalid, hold-down, and flush) provided robustness at the cost of slower convergence. EIGRP's flexible hold time can provide adaptability in heterogeneous environments. In terms of load balancing, while all three protocols support equal-cost load balancing, EIGRP's unique capability for unequal-cost load balancing through its variance feature would provide superior adaptability in diverse network environments. This feature allows EIGRP to optimise resource utilisation across all available routes, a significant advantage over RIP and OSPF.

e Suitability for Different Network Types

Based on our experimental findings and the theoretical framework presented in Section 2, we can draw conclusions about the suitability of each protocol for different network environments:

- RIP is best suited for small, simple networks with stable topologies, as predicted by its theoretical design discussed in Section 2.d.i. Our experiments confirmed its ease of configuration and low computational requirements, making it viable for environments where simplicity is prioritised over optimal performance. However its not scalable to a level often needed in practice and is now defunct.
- OSPF excels in medium to large enterprise networks, especially those with a hierarchical structure, aligning with its theoretical strengths outlined in Section 2.d.ii. Our simulations demonstrated OSPF's scalability, support for VLSM, and ability to handle complex topologies, making it a robust choice for diverse network designs. OSPF is the most common IGP, and is often preferred for traffic engineering. DV algorithms, RIP and EIGRP, are not good for traffic engineering.
- EIGRP offers the best overall performance in terms of convergence speed, efficient resource utilisation, and adaptability to various network conditions, consistent with the theoretical advantages discussed in Section 2.d.iii. EIGRP is particularly well-suited for large, complex networks that require rapid adaptation to changes and optimal path selection. EIGRP is ideal for scenarios where you want to implement unequal-cost load balancing — using all available routes despite differences in bandwidth.

f Limitations of FRR in implementing EIGRP

The implementation of EIGRP in FRRouting (FRR) presents several significant limitations that impact its functionality and reliability. As an open-source project developed primarily through volunteer efforts, FRR's EIGRP implementation lacks the comprehensive support and regular updates seen in its more widely used protocols like OSPF and BGP ("FRRouting", 2024a). This disparity is evident in the project's release notes, where EIGRP-related improvements are notably absent from recent versions. Compounding this issue is the documentation for EIGRP within FRR, which is particularly problematic; the last substantial update occurred in 2017, leaving users with outdated and potentially inaccurate information ("EIGRP", 2024). This lack of current documentation hinders effective deployment and troubleshooting of EIGRP in FRR environments, especially for network administrators accustomed to well-documented proprietary implementations.

The limited development of EIGRP in FRR can be attributed, in part, to its historical status as a proprietary protocol. Although Cisco has released some aspects of EIGRP as open-source, its RFC 7868 maintains only informational status rather than Internet Standards Track status (Savage et al., 2016). This ambiguous position in the networking standards landscape may contribute to reduced motivation for comprehensive implementation within open-source projects like FRR. Practical implementations of EIGRP in FRR have revealed several critical bugs that significantly impair its functionality (Brezular,

2017). A notable issue is the absence of query and reply packets (Bug #1), which are crucial for EIGRP's operation when no feasible successor is available. This limitation prevents the exploration of convergence scenarios essential for understanding EIGRP's behaviour in complex network topologies. Additionally, I/O errors have been reported when attempting to change bandwidth settings (Bug #2), further limiting the ability to test and optimise network configurations.

Perhaps the most significant issue is the persistence of outdated routing information (Bug #3). Extra paths and extraneous next-hop routers often appear in topology and routing tables, with old paths failing to be removed properly (Brezular, 2017). This bug severely undermines EIGRP's reputation for fast convergence, as it prevents the demonstration of rapid adaptation to link failures — a key advantage of the protocol in its proprietary implementations. Furthermore, the FRR implementation lacks certain commands that are standard in proprietary EIGRP versions. For instance, the absence of the 'no auto-summary' command limits flexibility in network design and addressing schemes ("EIGRP", 2024). While this particular limitation may not have affected our classful address-based experiments, it illustrates the gaps between FRR's EIGRP and fully-featured implementations.

These limitations collectively suggest that FRR's EIGRP implementation, as of version 9.0.3 used in this study, is not yet mature enough for production environments or comprehensive academic study ("FRRouting", 2024a). The lack of recent updates, persistent bugs, and missing features indicate that significant development work is still required to bring FRR's EIGRP implementation to parity with its proprietary counterparts or even with other routing protocols within FRR itself.

Our experimental results largely validate the theoretical characteristics of RIP, OSPF, and EIGRP discussed in Section 2. The choice between these protocols should be based on careful consideration of network size, complexity, stability, and specific performance requirements, as demonstrated by our GNS3 simulations. Our findings underscore the importance of selecting the appropriate routing protocol based on the unique needs of each network environment.

5 Conclusion (Chris, Kelly, Mukul)

This study has provided an in-depth analysis and exploration of three significant Interior Gateway Protocols (IGPs): RIP, OSPF, and EIGRP. Our investigation has illuminated the evolution of routing protocols and their critical role in modern network infrastructures, offering valuable insights into their performance characteristics, strengths, and limitations. The progression from RIP to OSPF to EIGRP reflects the increasing complexity and demands of modern networks. RIP, with its simplicity and ease of implementation, served as a foundational protocol but showed significant limitations in scalability and efficient path selection. OSPF addressed many of RIP's shortcomings by introducing a link-state approach, offering improved scalability through area segmentation and more nuanced path selection based on link costs. EIGRP, representing the latest evolution among these protocols, demonstrated superior performance in convergence speed, scalability, and adaptability to diverse network conditions.

Our analysis highlighted EIGRP's advantages in rapid convergence and efficient resource utilisation, particularly in complex network topologies. Its unique features, such as the Diffusing Update Algorithm (DUAL) and support for unequal-cost load balancing, position it as a highly effective solution for large enterprise networks. However, we also observed that OSPF remains a robust and widely-adopted protocol, especially in scenarios where link-state topology awareness is crucial, such as in traffic engineering applications. The limitations encountered in implementing EIGRP using the FRR package underscored the challenges in working with protocols that have proprietary roots. These constraints, while limiting our ability to fully explore some of EIGRP's advanced features, also highlight the ongoing development in open-source networking tools and the importance of standardisation in protocol implementation. Despite these limitations, the field of routing protocols offers abundant opportunities for further learning and exploration, with numerous resources available online, including materials for CCNA and other certifications.

Our study also emphasised the importance of considering administrative distance and route redistribution when designing networks that implement multiple routing protocols. The interplay between different protocols and their respective administrative distances can significantly impact route selection and overall network performance, particularly in complex, multi-protocol environments.

Looking forward, several avenues for further research and exploration present themselves. These include a deeper investigation into EIGRP's advanced features such as load balancing and unequal-cost load balancing using the variance command, exploring the integration of multiple protocols in a single network environment, and examining how the DUAL algorithm is modified and utilised in newer protocols like Babel. Additionally, extending the study to include Exterior Gateway Protocols, particularly BGP, would provide a more comprehensive view of internet routing dynamics.

It is crucial to emphasise the fundamental role that routing protocols play in shaping modern communications and digital infrastructure. As our reliance on interconnected systems continues to grow, the efficiency and reliability of these protocols become increasingly critical. When designing a network and selecting a routing protocol, it is imperative to consider not only how the protocol will function under ideal conditions but also how it might behave in failure scenarios. This approach ensures the development of robust, resilient networks capable of withstanding the challenges of our increasingly connected world.

In conclusion, this study reinforces the critical importance of understanding and implementing appropriate routing protocols in network design. The choice between RIP, OSPF, and EIGRP —— or indeed, any routing protocol —— should be based on careful consideration of network size, complexity, stability requirements, and specific performance needs. As we continue to rely more heavily on digital communications in all aspects of life, the ongoing evolution and refinement of these protocols will remain vital to meeting the ever-growing demands of our interconnected world. The field of network routing protocols continues to be a dynamic and crucial area of study, promising further innovations to address the challenges of tomorrow's networks.

References

Biradar, A. G. (2020). A comparative study on routing protocols: Rip, ospf and eigrp and their analysis using gns-3. *2020 5th IEEE International Conference on Recent Advances and Innovations in Engineering (ICRAIE)*, 1–5. https://doi.org/10.1109/ICRAIE51050.2020.9358327

Brezular. (2017). Frrouting software with eigrp support [Accessed: 2024-11-06]. https://brezular.com/2017/11/02/frrouting-software-with-eigrp-support/

Cisco Press. (2016). Static vs Dynamic Routing.

CiscoPress. (2011, December). *Designing cisco network service architectures (arch): Developing an optimum design for layer 3 (ccdp)* [Accessed: 2024-11-06]. https://www.ciscopress.com/articles/article.asp?p=1763921

Eigrp [Accessed: 2024-11-06]. (2024). https://docs.frrouting.org/en/latest/eigrpd.html

Ergun, O. (2019, November). *Eigrp rfc 7868* [Accessed: 2024-11-06]. Retrieved November 26, 2019, from https://orhanergun.net/eigrp-rfc-7868

Frrouting [Accessed: 2024-11-06]. (2024a). https://frrouting.org/

FRRouting. (2024b). Frr documentation [Accessed: October 16, 2024].

GNS3. (2024). Gns3 documentation [Accessed: October 16, 2024].

Halabi, S., & McPherson, D. (2000). *Internet routing architectures, second edition* (2nd). Cisco Systems.

IBM. (2024, October). *Open shortest path first*. Retrieved October 7, 2024, from https://www.ibm.com/docs/en/i/7.5?topic=routing-open-shortest-path-first

Internet Engineering Task Force. (1998). RFC2453 - RIP version 2.

Kurose, J. F., & Ross, K. W. (2020). *Computer networking: A top-down approach* (8th). Pearson.

Malik, S. U. R., Srinivasan, S. K., Khan, S. U., & Wang, L. (2012). A methodology for ospf routing protocol verification.

Medhi, D., & Ramasamy, K. (2017). *Network routing, second edition: Algorithms, protocols, and architectures* (2nd). Morgan Kaufmann Publishers Inc.

Sanders, C. (2017). *Practical packet analysis: Using wireshark to solve real-world network problems* (3rd ed.). No Starch Press.

Savage, P., Ng, J., Moore, S., Slice, D., White, R., & Paluch, S. (2016, May). *Cisco's enhanced interior gateway routing protocol (eigrp)* (RFC No. 7868). RFC Editor. RFC Editor. http://www.rfc-editor.org/rfc/rfc7868.txt

Senanayake, R. (2024). Module 4 - network layer [ELEN90061 Communication Networks, The University of Melbourne].

Tanenbaum, A. S., & Wetherall, D. J. (2011). *Computer networks* (5th ed.). Pearson.

Verma, A., & Bhardwaj, N. (2016). A review on routing information protocol (rip) and open shortest path first (ospf) routing protocol. *International Journal of Future Generation Communication and Networking*, *9*, 161–170. https://doi.org/10.14257/ijfgcn.2016.9.4.13

6 Appendix

a GNS3 FRR setup

1. **Create Configuration Files:**

```
cd /etc/frr
touch vtysh.conf
touch frr.conf
```

2. **Start FRR Daemons:**

```
cd /usr/lib/frr
# RIP: ripd,
# OSPF: ospfd,
# EIGRP: eigrpd
./watchfrr zebra ripd &
```

3. **Enter vtysh Shell:**

```
vtysh
```

4. **Configure Router Interfaces:**

```
# add all the ethernet
# terminals' ip
configure terminal
interface eth0
ip address 192.168.0.1/24
exit
interface eth1
ip address 10.1.1.2/24
exit
exit
```

5. **Set Up Routing Protocol:**

```
# add all the rip networks
# connected to the router
# RIP: rip
# OSPF: ospf
# EIGRP: eigrp
configure terminal
router rip
network 192.168.0.0/24
network 10.1.1.0/24
exit
exit
```

6. **Save Configuration:**

```
write
```

Some other useful commands are:

1. **Show IP Routing Table:**

```
show ip route
```

2. **Show Router Interface:**

```
show int brief
```

3. **Show EIGRP Tables:**

```
show ip eigrp neighbor
show ip eigrp topology
show ip eigrp interface
```

4. **Configure EIGRP links:**

```
config
int eth0
eigrp bandwidth 1000000
delay 20
```

b Supplementary screenshots

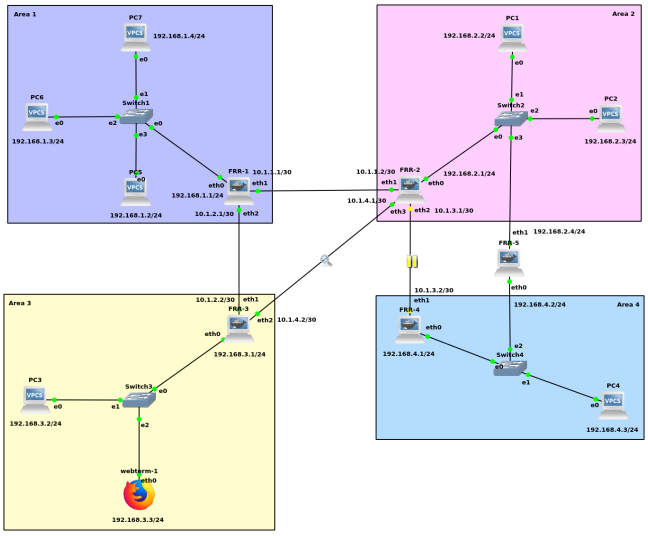


Figure 28: OSPF post-disturbance topology

```
FRR-2
File  Edit  View  Search  Terminal  Help
2024/10/27 13:28:47 WATCHFRR: [WFP93-1D146] configuration write completed with exit code 0
FRR-2# show ip eigrp interface

EIGRP interfaces for AS(1)

Interface        Bandwidth  Delay      Peers  Xmit Queue   Mean    Pacing Time    Multicast    Pending  Hello    Holdtime
                                              Un/Reliable  SRTT    Un/Reliable    Flow Timer   Routes
eth0             100000     10         1      0 / 1         0       0              0            0        5        15
eth1             100000     10         1      0 / 0         0       0              0            0        5        15
FRR-2# config
FRR-2(config)# int eth0
FRR-2(config-if)# delay 100
FRR-2(config-if)# q
FRR-2(config)# q
FRR-2# write
Note: this version of vtysh never writes vtysh.conf
Building Configuration...
Integrated configuration saved to /etc/frr/frr.conf
[OK]
2024/10/27 13:29:05 WATCHFRR: [WFP93-1D146] configuration write completed with exit code 0
FRR-2# show ip eigrp interface

EIGRP interfaces for AS(1)

Interface        Bandwidth  Delay      Peers  Xmit Queue   Mean    Pacing Time    Multicast    Pending  Hello    Holdtime
                                              Un/Reliable  SRTT    Un/Reliable    Flow Timer   Routes
eth0             100000     100        1      0 / 1         0       0              0            0        5        15
eth1             100000     10         1      0 / 0         0       0              0            0        5        15
FRR-2#
```

Figure 29: Delay of interface eth0 changed from default value of 10 to 100.

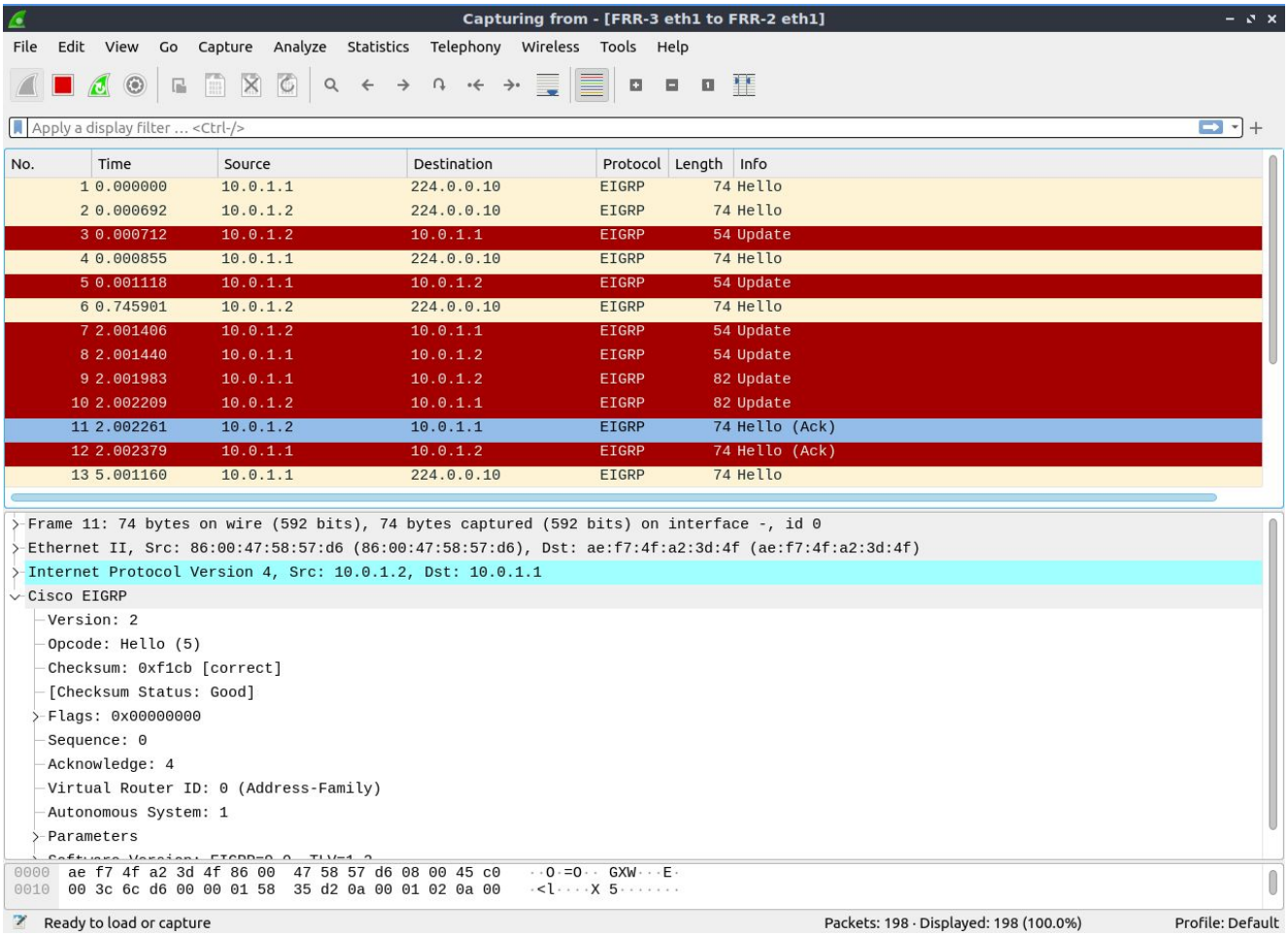


Figure 30: Wireshark capture of Update packets, with associated Acknowledgements, sent on a previously suspended link, immediately after the link had been restored.

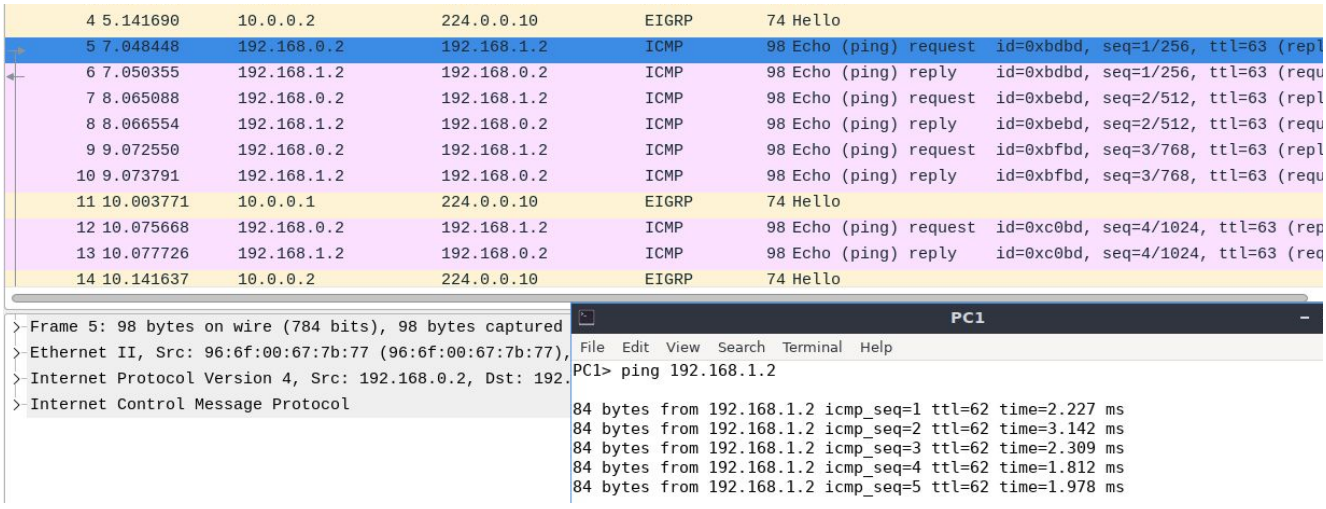


Figure 31: Ping from PC1 to PC2 in topology A*.

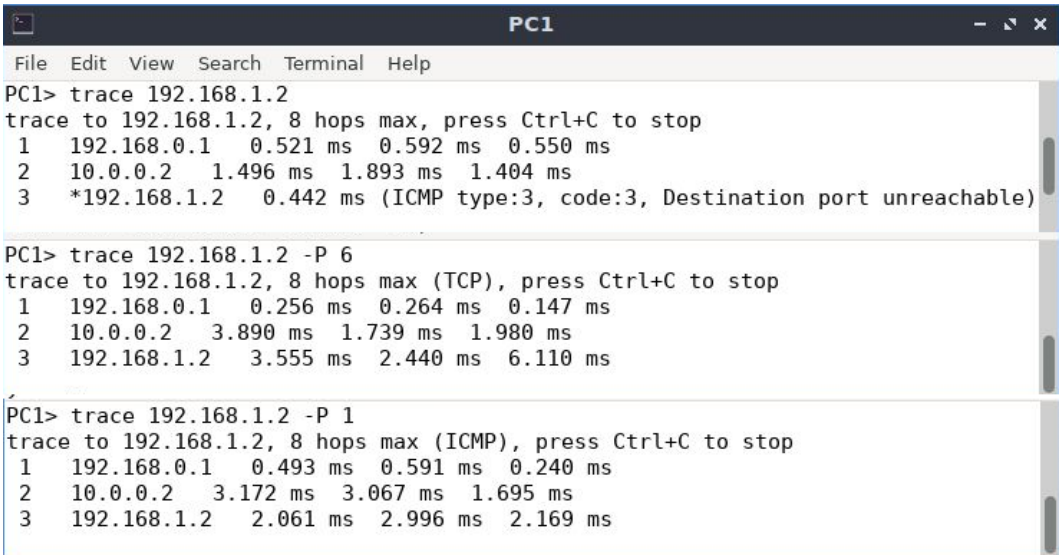


Figure 32: Trace tracks the movement of a packet through the network to a destination interface. The destination address 192.168.1.2 was reachable using ICMP or TCP, but not using UDP.

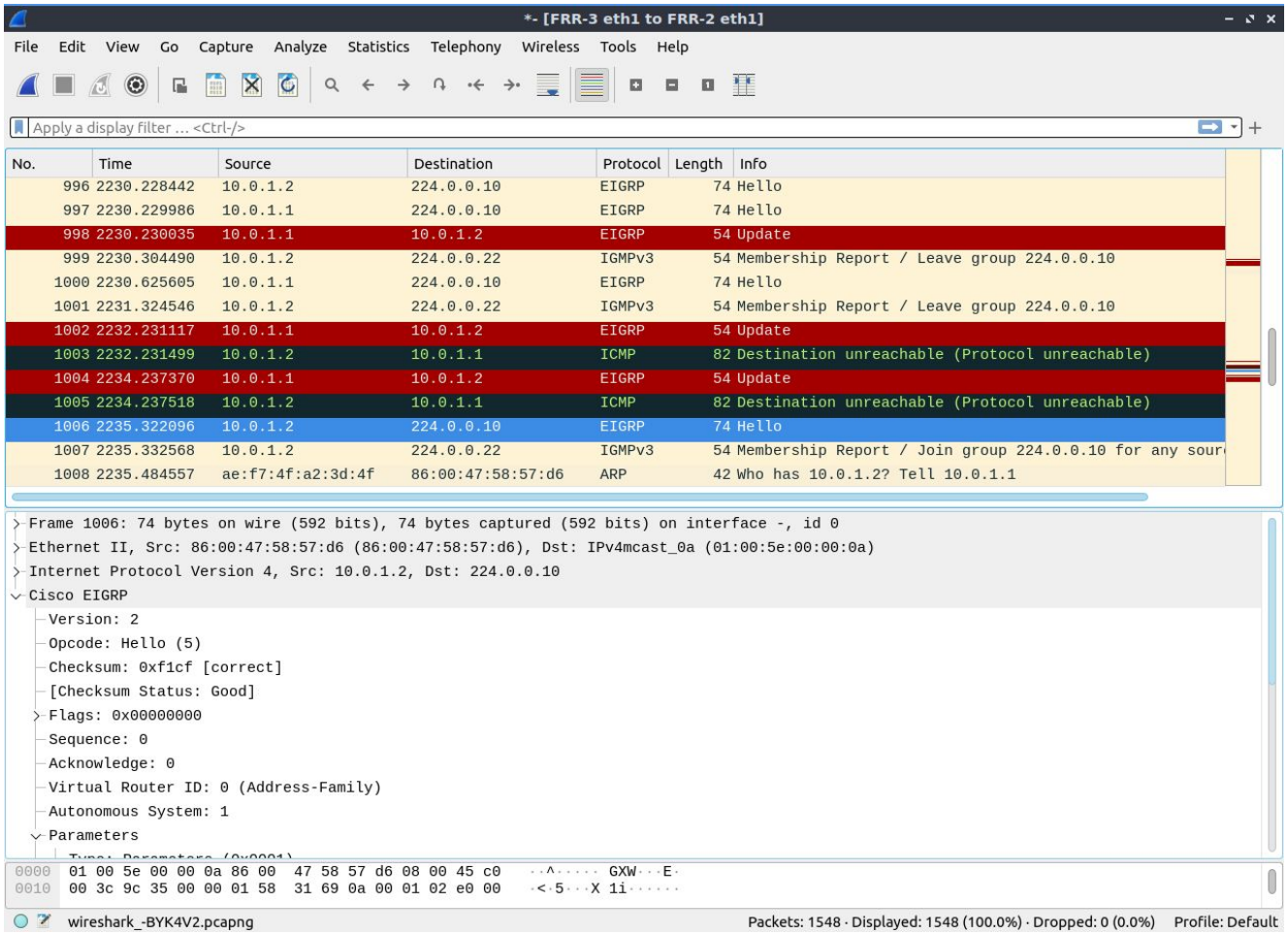


Figure 33: Ping and trace commands fail to reach the destination address after link metrics have been updated.

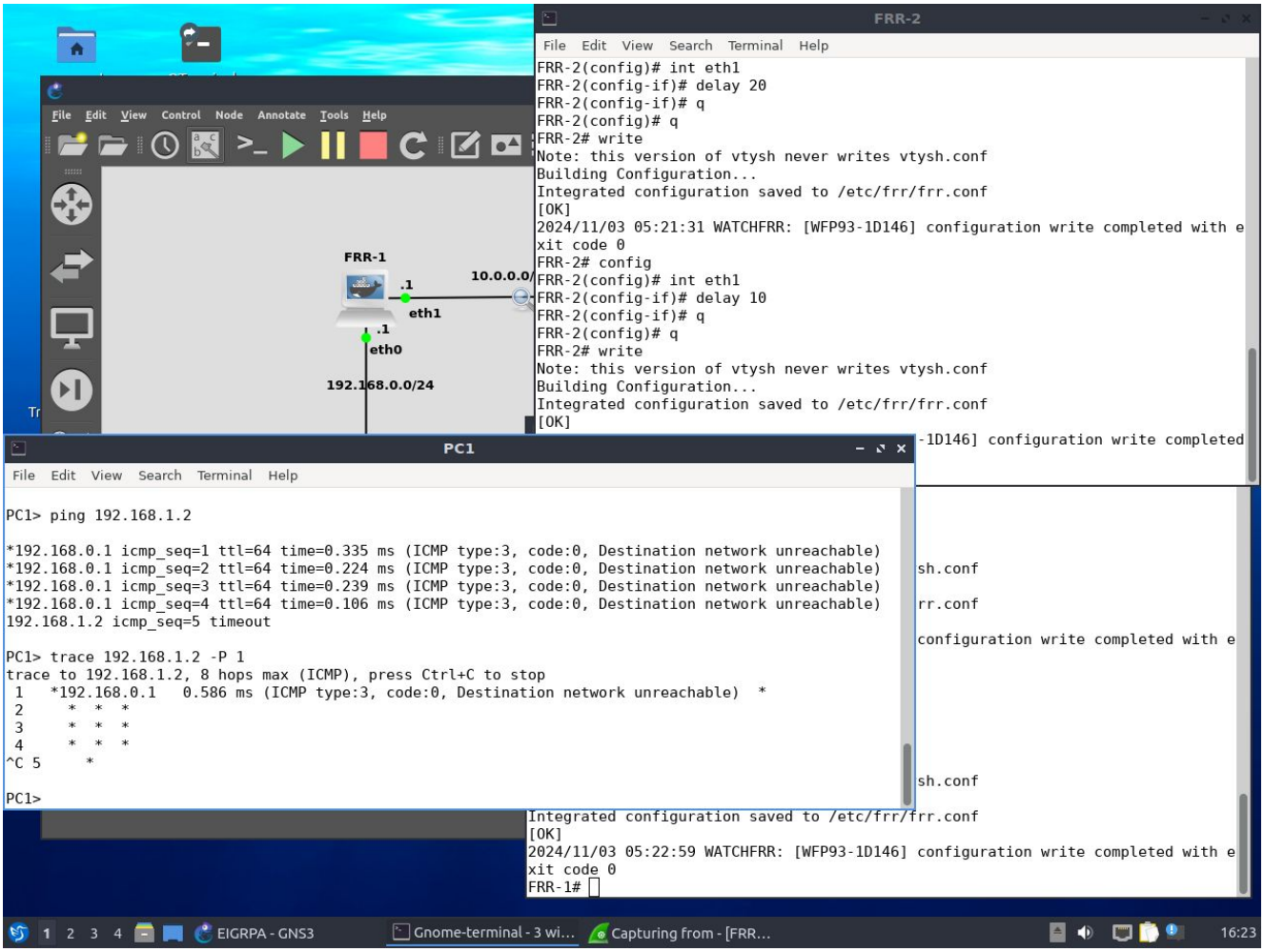


Figure 34: Ping and trace commands fail to reach the destination address even after link metrics reverted back to default values.

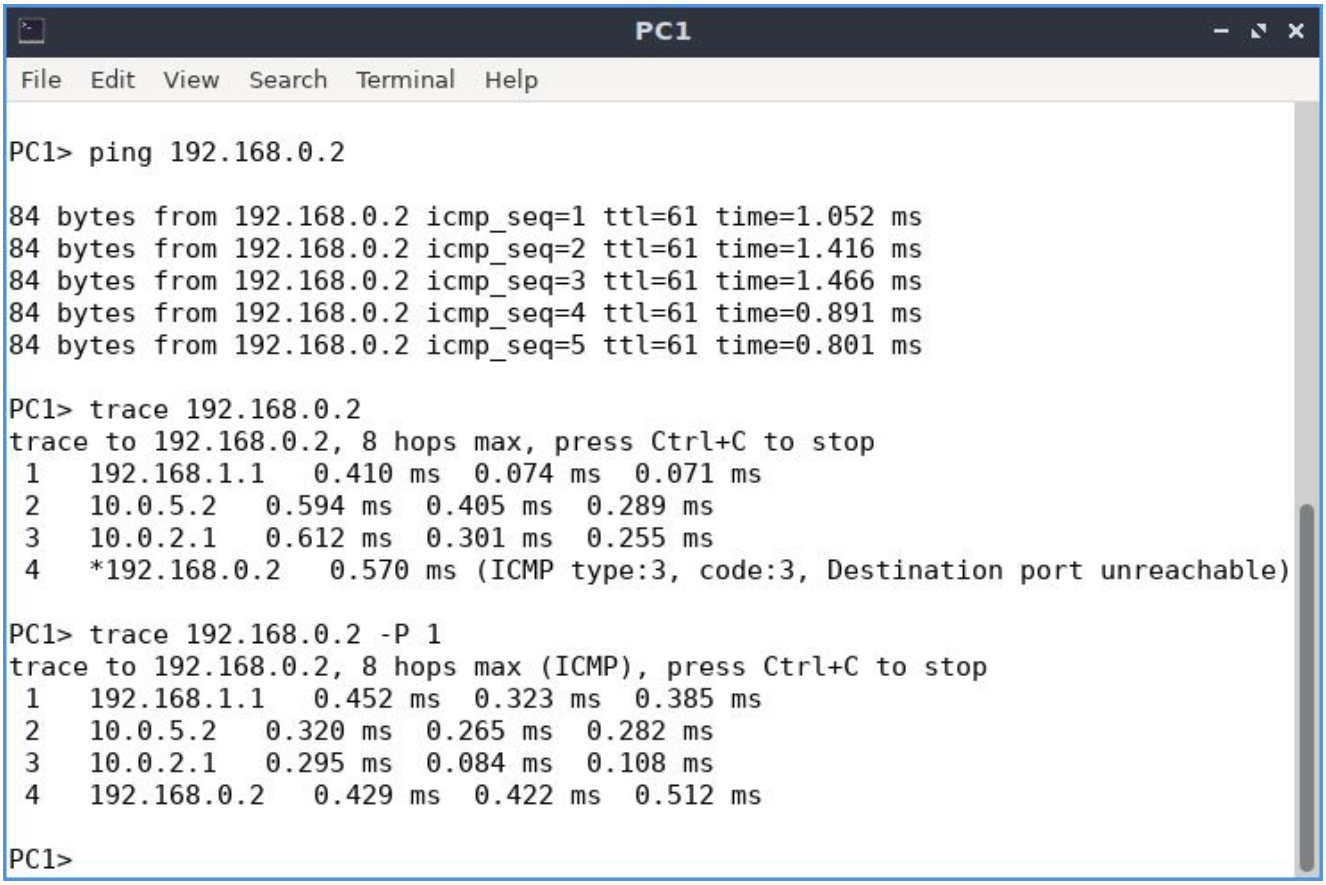

PC1
File Edit View Search Terminal Help
PC1> ping 192.168.0.2
84 bytes from 192.168.0.2 icmp_seq=1 ttl=61 time=1.052 ms
84 bytes from 192.168.0.2 icmp_seq=2 ttl=61 time=1.416 ms
84 bytes from 192.168.0.2 icmp_seq=3 ttl=61 time=1.466 ms
84 bytes from 192.168.0.2 icmp_seq=4 ttl=61 time=0.891 ms
84 bytes from 192.168.0.2 icmp_seq=5 ttl=61 time=0.801 ms
PC1> trace 192.168.0.2
trace to 192.168.0.2, 8 hops max, press Ctrl+C to stop
1 192.168.1.1 0.410 ms 0.074 ms 0.071 ms
2 10.0.5.2 0.594 ms 0.405 ms 0.289 ms
3 10.0.2.1 0.612 ms 0.301 ms 0.255 ms
4 *192.168.0.2 0.570 ms (ICMP type:3, code:3, Destination port unreachable)
PC1> trace 192.168.0.2 -P 1
trace to 192.168.0.2, 8 hops max (ICMP), press Ctrl+C to stop
1 192.168.1.1 0.452 ms 0.323 ms 0.385 ms
2 10.0.5.2 0.320 ms 0.265 ms 0.282 ms
3 10.0.2.1 0.295 ms 0.084 ms 0.108 ms
4 192.168.0.2 0.429 ms 0.422 ms 0.512 ms
PC1>

Figure 35

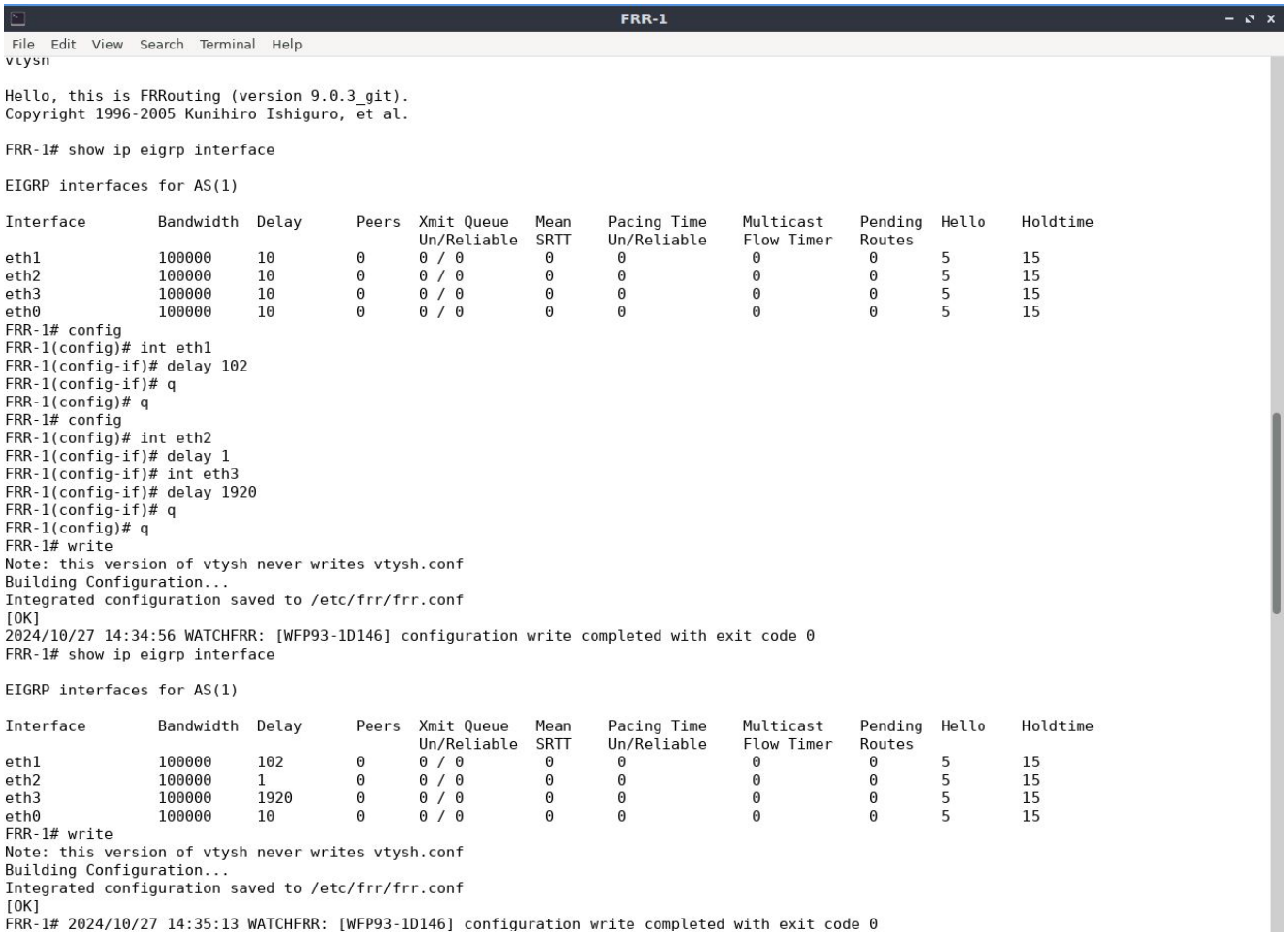


Figure 36: Updated delay values for FRR-1 when link distance metrics are changed.

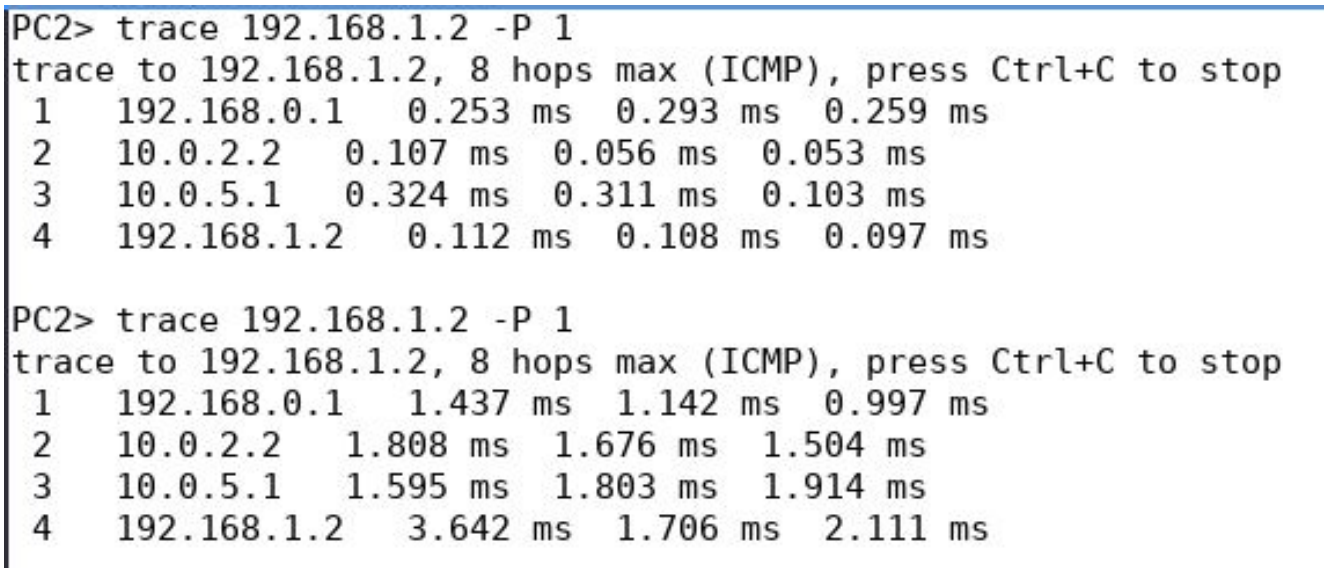


Figure 37: Trace command showing the passage of a packet through the middle path of the network to reach 192.168.1.0, before and after metrics changed. The packet takes longer in the second execution after longer delays were introduced.

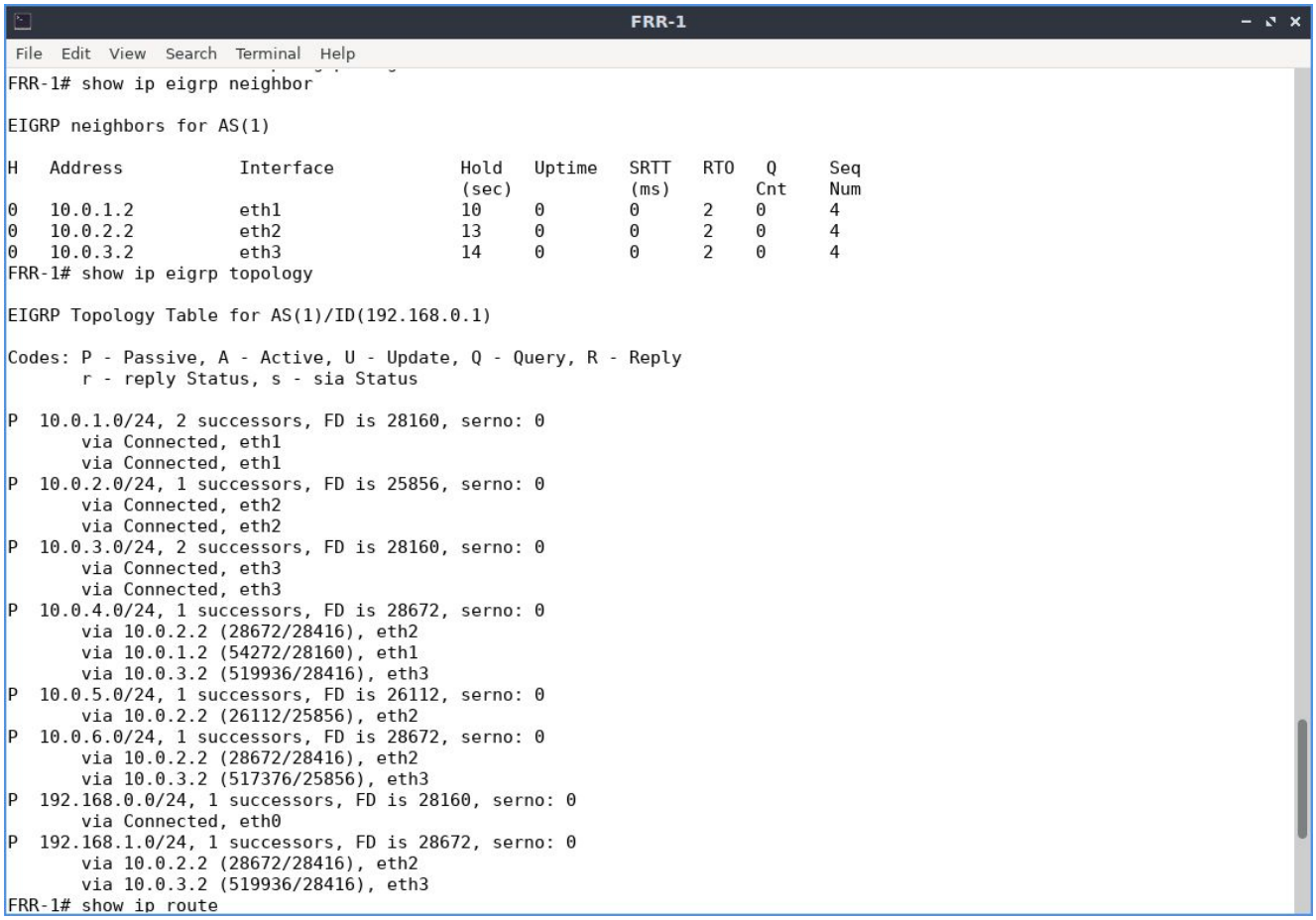


Figure 38: Full neighbour and topology tables for FRR-1.

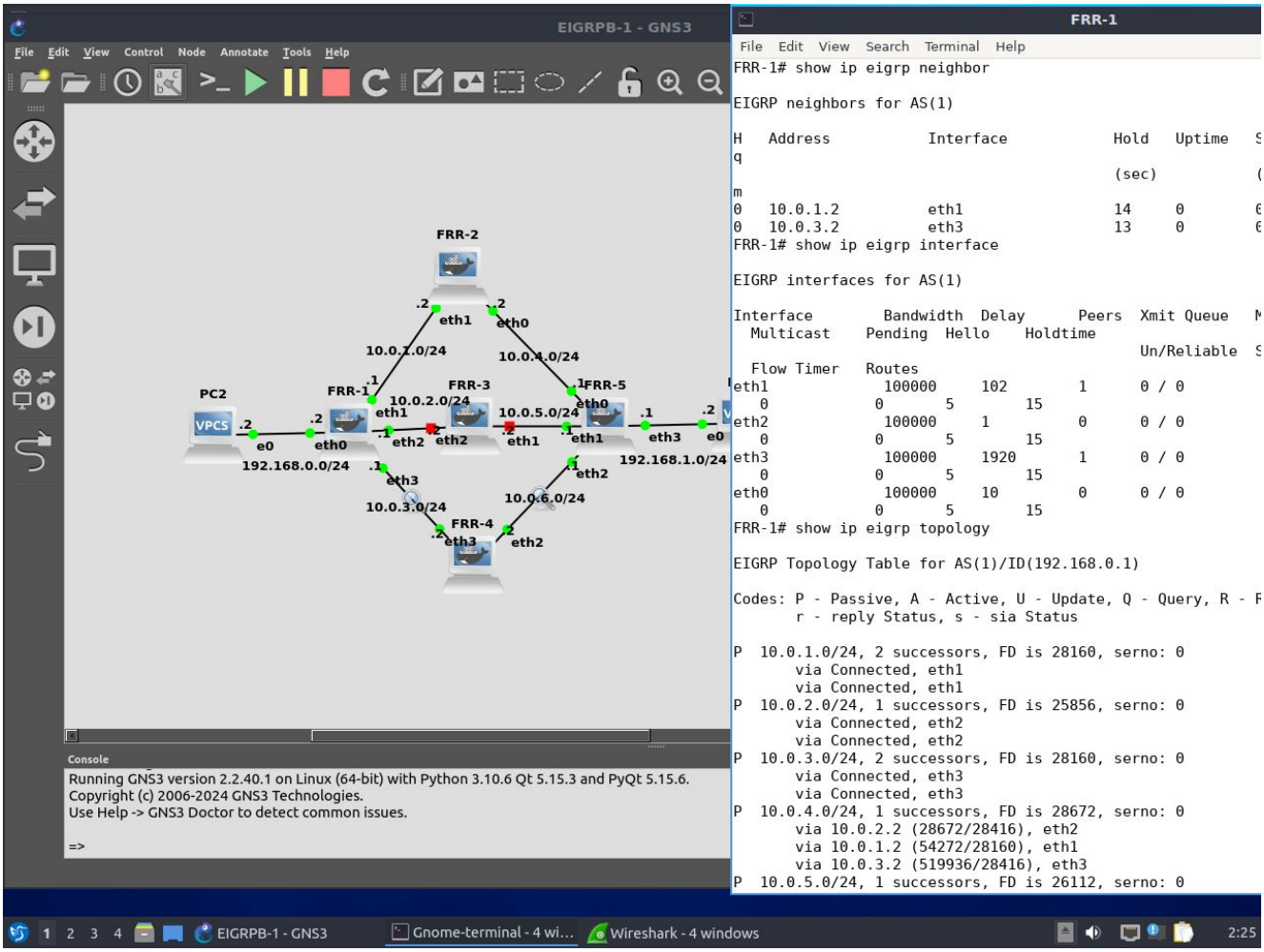


Figure 39: When FRR-3 in Topology B fails, the neighbour table for FRR-1 is correct.

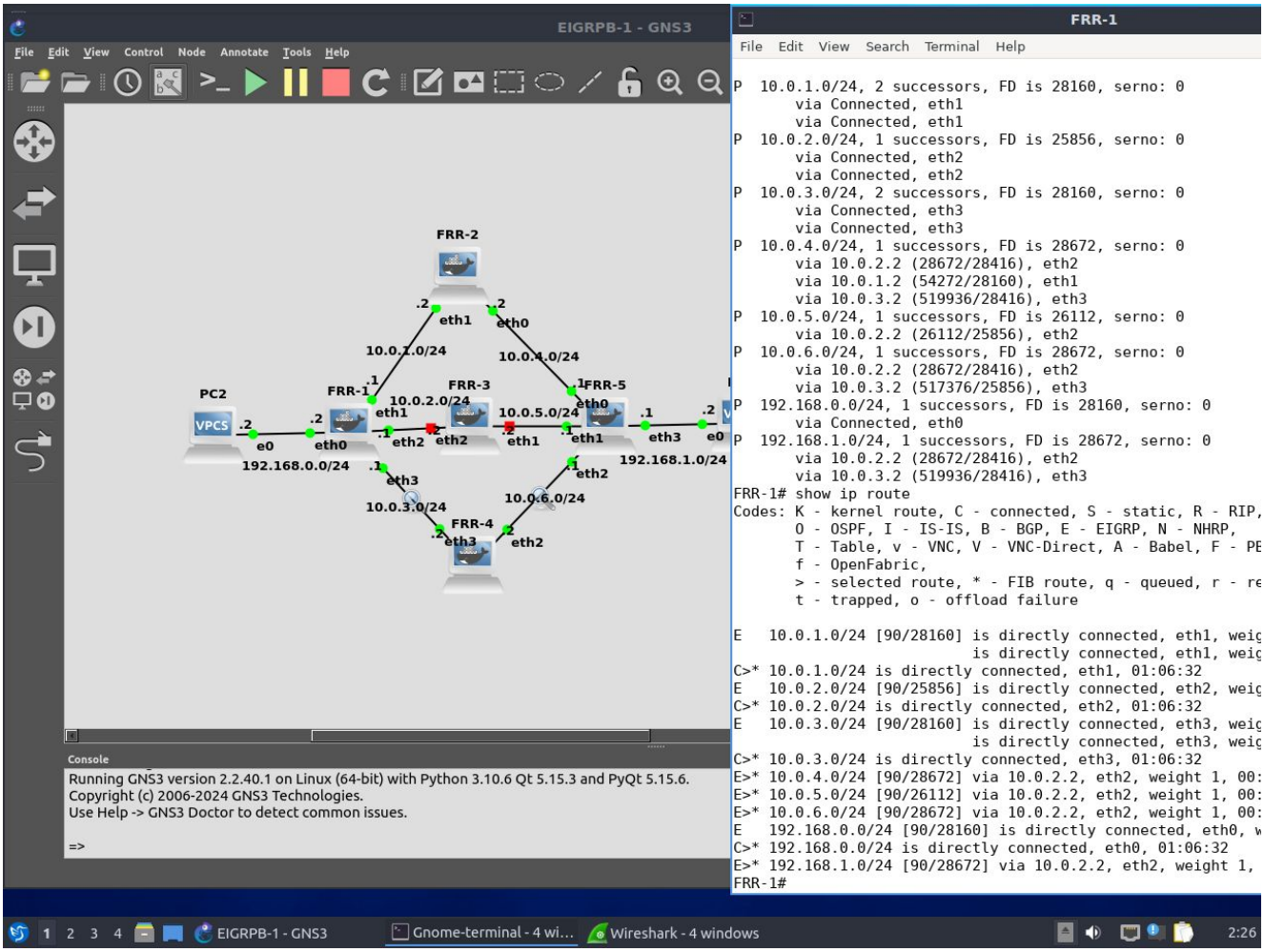


Figure 40: When FRR-3 in Topology B fails, the topology and routing tables still contain routes through FRR-3.

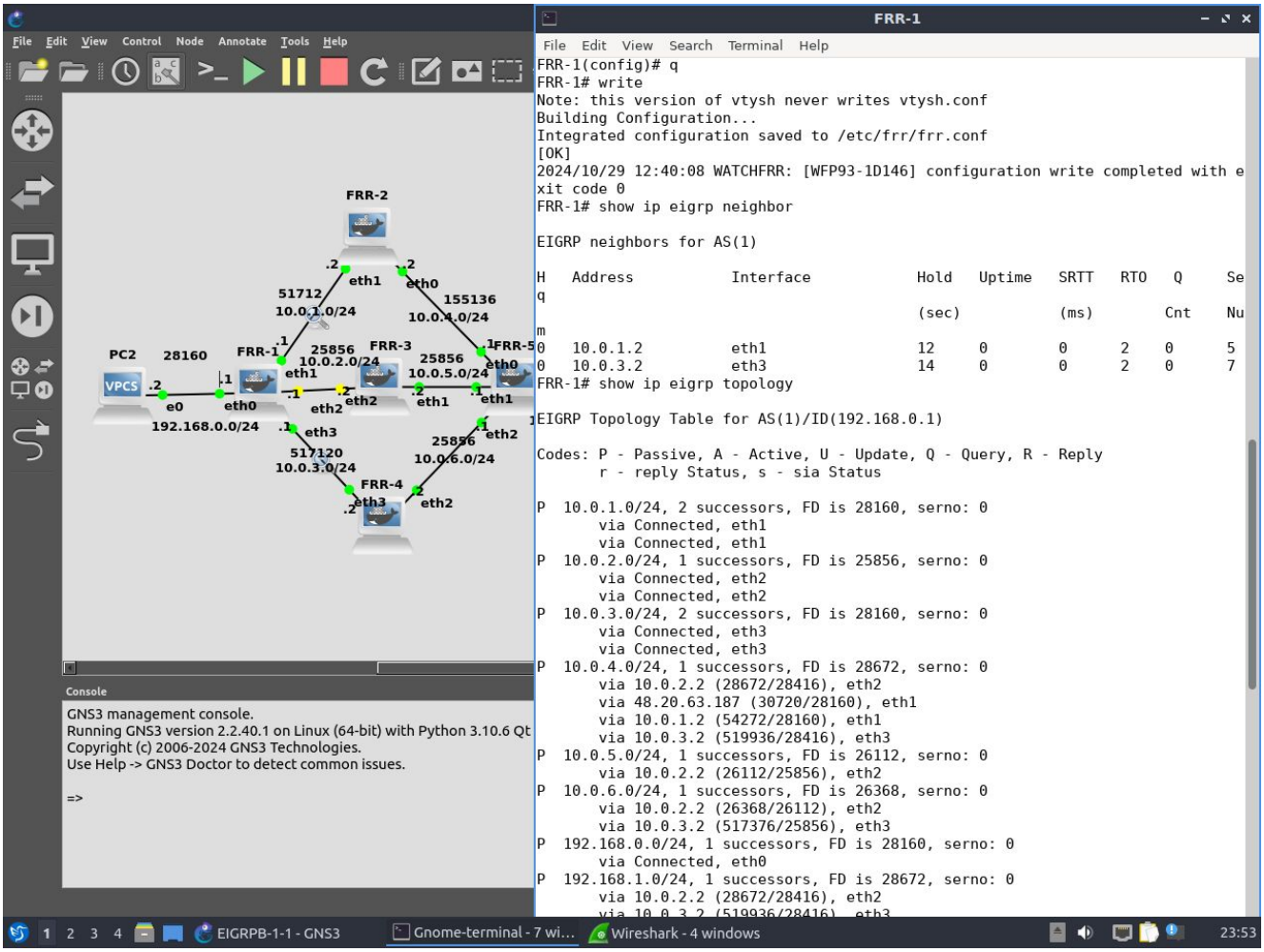


Figure 41: When the link between FRR-1 and FRR-3 in Topology B fails, the neighbour table for FRR-1 is correct.

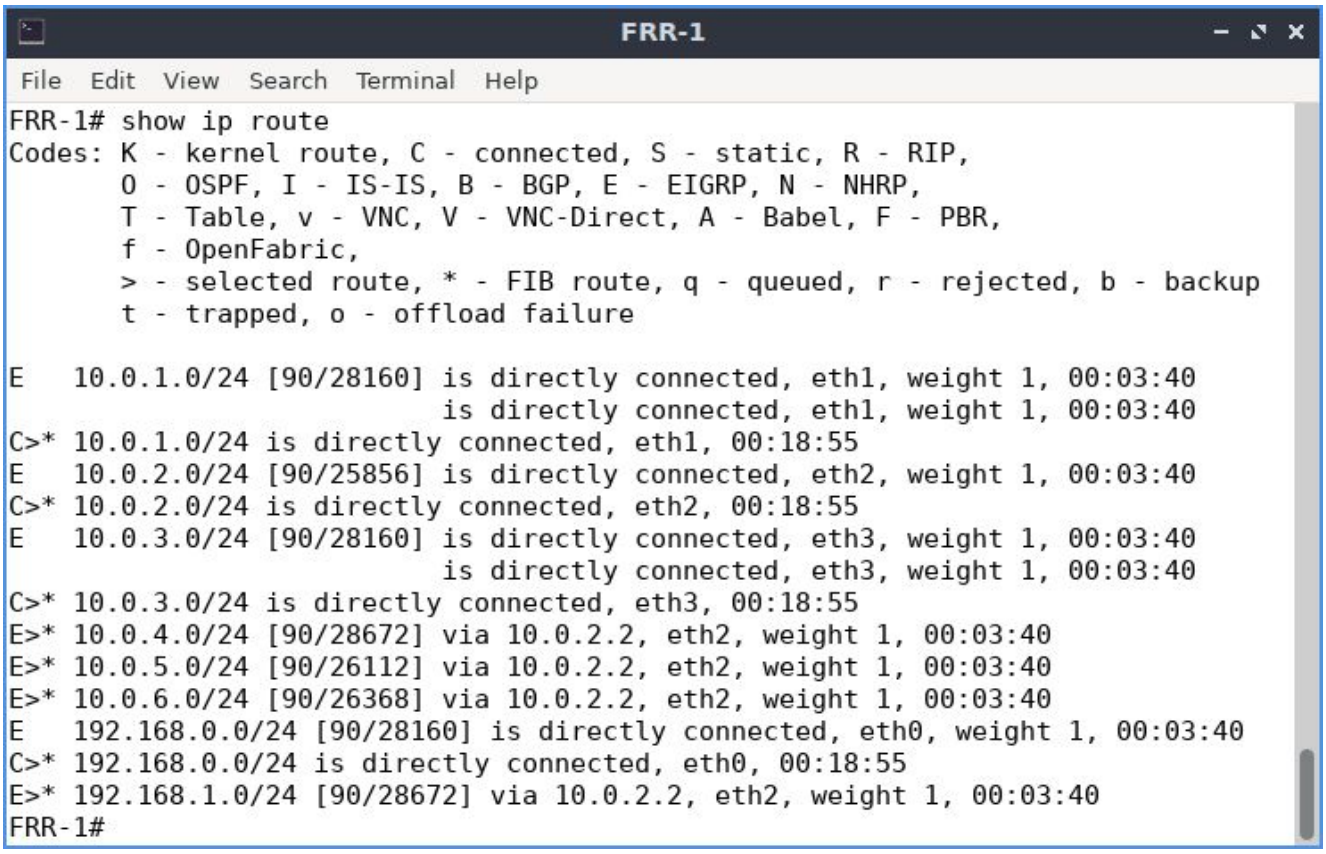


Figure 42: When the link between FRR-1 and FRR-3 in Topology B fails, the topology and routing tables still contain routes through FRR-3.

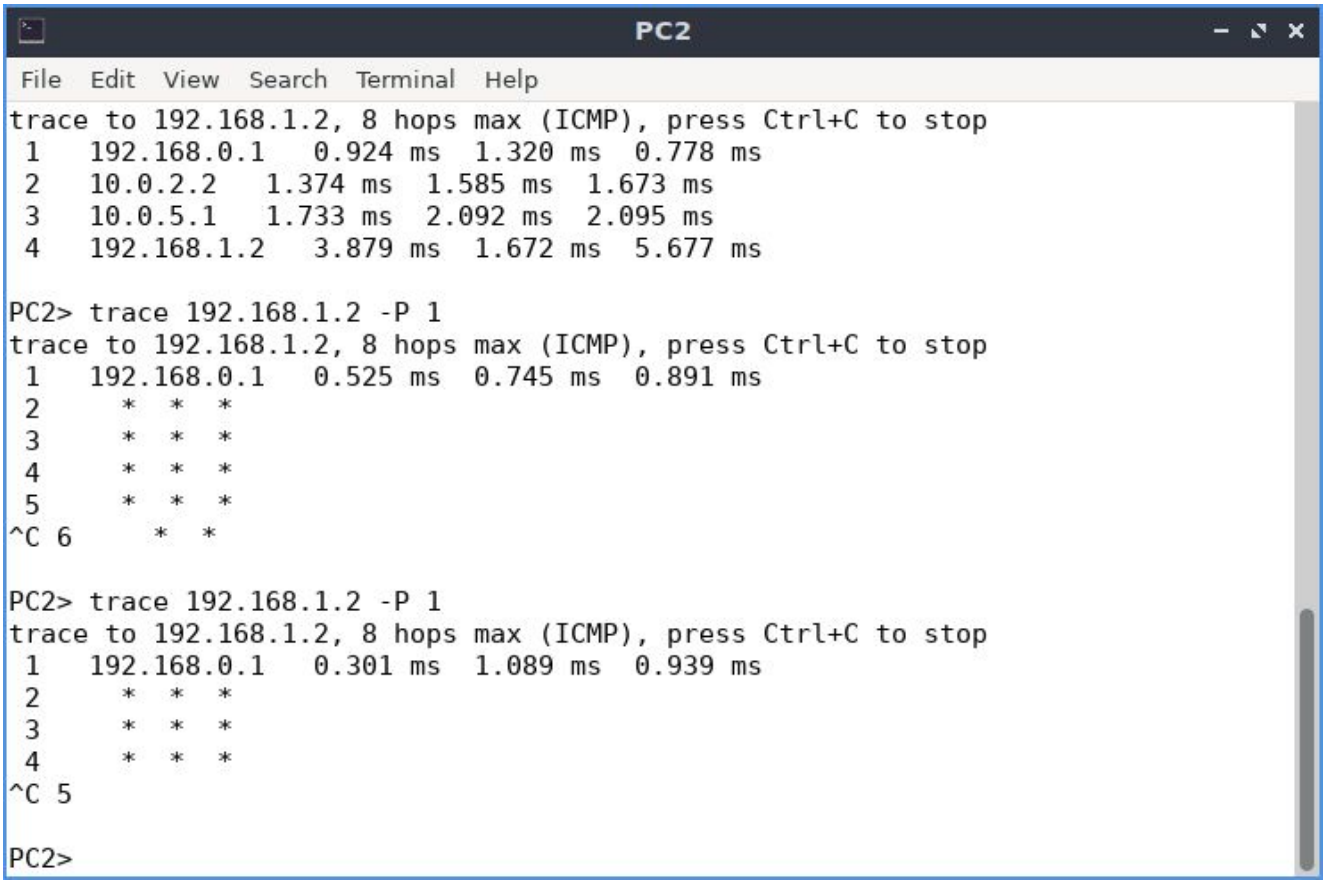


Figure 43: When FRR-3 in Topology B fails, using the trace command the system enters an infinite loop, unable to find the next-hop router since it is down.

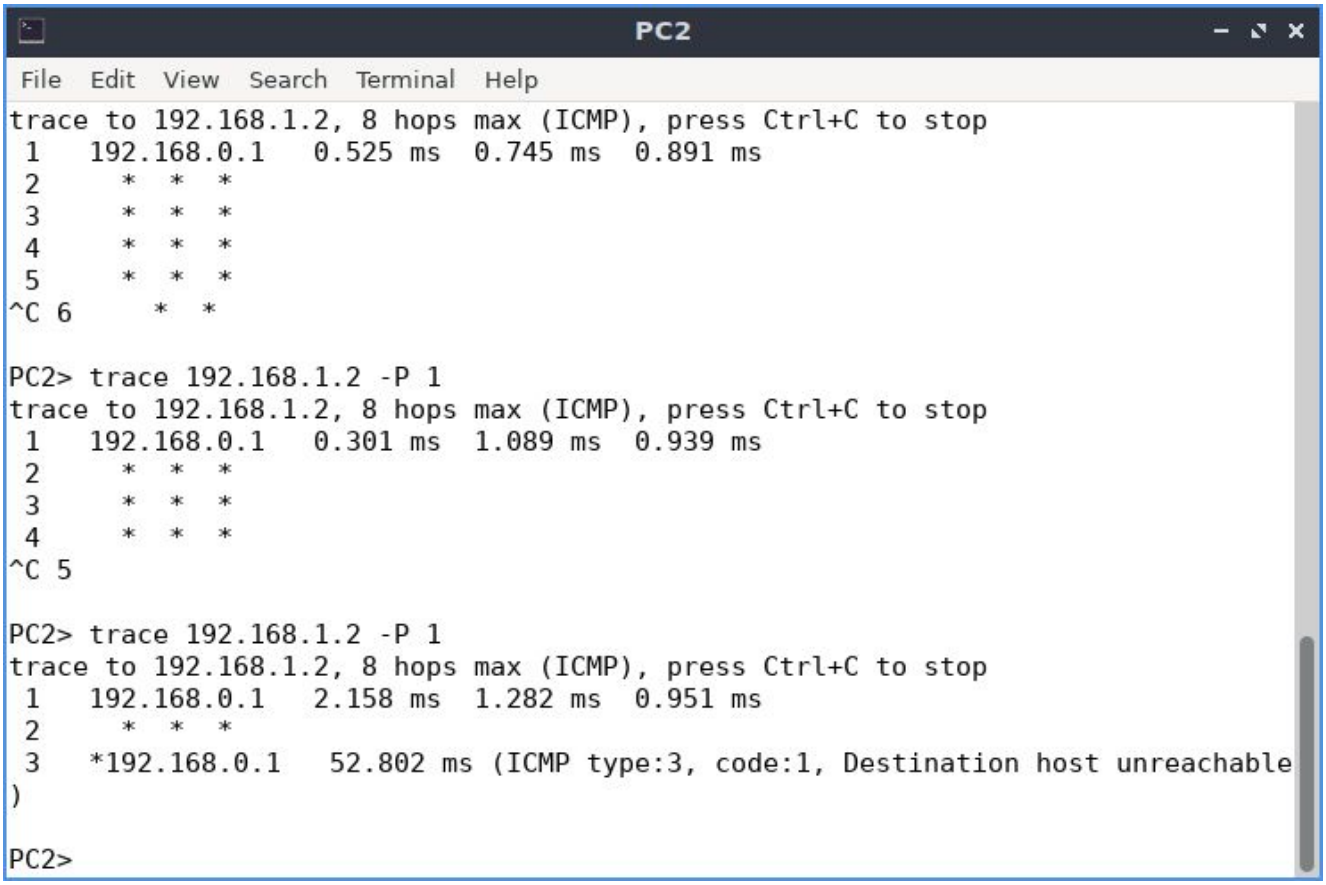
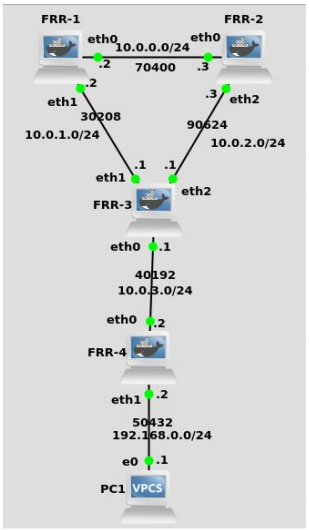


Figure 44: When the link between FRR-1 and FRR-3 in Topology B fails, using the trace command the destination becomes unreachable.




```
FRR-3
File  Edit  View  Search  Terminal  Help
Interface        Bandwidth  Delay      Peers  Xmit Queue   Mean    Pacing Time   Multicast    Pending  Hello     Holdtime
                                              Un/Reliable  SRTT    Un/Reliable   Flow Timer   Routes
eth1             100000     18         1      0 / 0        0       0             0            0        5        15
eth2             100000     254        1      0 / 0        0       0             0            0        5        15
eth0             100000     57         1      0 / 0        0       0             0            0        5        15
FRR-3# show ip eigrp topology

EIGRP Topology Table for AS(1)/ID(10.0.3.1)

Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply
       r - reply Status, s - sia Status

P  10.0.0.0/24, 2 successors, FD is 30720, serno: 0
       via 16.161.201.34 (30720/28160), eth1
       via 224.195.201.34 (30720/28160), eth2
       via 10.0.1.2 (32768/28160), eth1
       via 10.0.2.3 (93184/28160), eth2
P  10.0.1.0/24, 2 successors, FD is 28160, serno: 0
       via Connected, eth1
       via Connected, eth1
P  10.0.2.0/24, 2 successors, FD is 28160, serno: 0
       via Connected, eth2
       via Connected, eth2
P  10.0.3.0/24, 2 successors, FD is 28160, serno: 0
       via Connected, eth0
       via Connected, eth0
P  192.168.0.0/24, 1 successors, FD is 30720, serno: 0
       via 0.0.0.0 (30720/28160), eth0
       via 10.0.3.2 (42752/28160), eth0
FRR-3# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, F - PBR,
       f - OpenFabric,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

E   10.0.0.0/24 [90/30720] via 16.161.201.34, eth1 inactive, weight 1, 00:03:28
                          via 224.195.201.34, eth2 inactive, weight 1, 00:03:28
E   10.0.1.0/24 [90/28160] is directly connected, eth1, weight 1, 00:03:28
                          is directly connected, eth1, weight 1, 00:03:28
C>* 10.0.1.0/24 is directly connected, eth1, 00:05:05
E   10.0.2.0/24 [90/28160] is directly connected, eth2, weight 1, 00:03:28
                          is directly connected, eth2, weight 1, 00:03:28
C>* 10.0.2.0/24 is directly connected, eth2, 00:05:05
E   10.0.3.0/24 [90/28160] is directly connected, eth0, weight 1, 00:03:28
                          is directly connected, eth0, weight 1, 00:03:28
C>* 10.0.3.0/24 is directly connected, eth0, 00:05:05
E>* 192.168.0.0/24 [90/30720] is directly connected, eth0, weight 1, 00:03:28
```

Figure 45: On the left, topology C: loop arrangement of three routers connected to another router, which is connected to a PC, each link labelled with metrics altered from the default values. On the right, the topology and routing tables for FRR-3, including incorrect FD for the target subnet and spurious interface IP addresses.

Figure 46