Department of Electronic & Telecommunication Engineering,
University of Moratuwa, Sri Lanka.

Routing Protocol Design Report

Network Surgeons

220163X	Fernando D.S.
220423V	Nethkalana W.K.S.
220429U	Nimantha K.L.W.O.
220443G	Paranayapa D.R.L.W.S.

EN2150 - Communication Network Engineering

Contents

1 Introduction

Routing protocols are essential for determining the best paths for data to travel across networks. Widely used protocols like RIP, OSPF, IS-IS, and BGP have made large-scale communication over the Internet possible. However, these protocols still have several limitations, such as slow convergence, limited scalability, poor fault tolerance, and vulnerability to security threats.

This report presents the design of a new routing protocol aimed at overcoming major limitations found in existing protocols. The proposed protocol focuses on improving performance in terms of scalability, efficiency, convergence behavior, fault tolerance, and security. The report includes a summary of the weaknesses of current protocols, a detailed explanation of the new design, simulation results comparing performance, and an analysis of security and scalability features.

2 Summary of Existing Protocols and Their Limitations

The design of any modern routing protocol must be firmly rooted in a comprehensive understanding of the strengths and limitations of existing solutions. Over the past several decades, various routing protocols have been developed to support the growing needs of different types of networks—ranging from small local area networks (LANs) to the vast and complex interconnections of the global Internet. Among these protocols, **RIP**, **OSPF**, **IS-IS**, and **BGP** represent some of the most significant and widely deployed technologies.

However, none of these protocols are universally optimal. Each was developed with a specific set of assumptions and architectural goals, and over time, many of their design constraints have become limitations in modern environments that demand high scalability, flexibility, automation, and robust security. In this section, we systematically examine each protocol in depth, discussing their core functionalities and highlighting the limitations that restrict their effectiveness in evolving network contexts.

2.1 RIP (Routing Information Protocol)

RIP is one of the earliest distance-vector routing protocols developed for IP networks. It operates by having routers exchange their entire routing tables with their immediate neighbors at regular intervals (typically every 30 seconds). Although RIP's simplicity made it a viable solution in the early days of networking, it is now considered obsolete for all but the most basic network topologies.

Key Limitations:

- **Scalability Issues:** RIP supports a maximum hop count of 15, meaning any destination more than 15 hops away is considered unreachable.
- **Slow Convergence:** RIP is slow to adapt to network changes due to its reliance on periodic updates and lack of rapid convergence mechanisms.
- **Classful Routing (RIPv1):** The original RIP version does not support subnet masks or CIDR, resulting in inefficient address utilization.
- **Weak Security Model:** RIPv1 lacks authentication, and RIPv2's basic MD5-based mechanisms are insufficient for high-security environments.

2.2 OSPF (Open Shortest Path First)

OSPF is a link-state routing protocol designed to address RIP's shortcomings. It introduces more intelligent routing through a link-state database and shortest-path computation using Dijkstra's algorithm.

Key Limitations:

- **High Computational Overhead:** The SPF algorithm becomes resource-intensive in large, flat networks.

- **Hierarchical Design Requirements:** OSPF mandates the use of a backbone area (Area 0), which increases design complexity.
- **LSA Flooding:** Link State Advertisements can cause excessive control traffic in dynamic environments.
- **Bandwidth Metric Inaccuracy:** Interface cost must often be manually tuned to reflect real bandwidth, introducing potential configuration errors.

2.3 IS-IS (Intermediate System to Intermediate System)

IS-IS is another link-state protocol with operational similarities to OSPF but was originally designed for the OSI model. It has found a niche in large-scale service provider networks.

Key Limitations:

- **Steep Learning Curve:** IS-IS requires understanding OSI concepts and uses a different configuration model than IP-based protocols.
- **Layer 2 Operation:** IS-IS operates directly over Layer 2, which may hinder integration with IP-based monitoring tools.
- **Rigid Area Design:** Routers in IS-IS are bound to a single area, limiting design flexibility compared to OSPF's interface-based model.

2.4 BGP (Border Gateway Protocol)

BGP is the de facto interdomain routing protocol used across the Internet. It supports extensive policy control but is complex and lacks certain IGP features.

Key Limitations:

- **Unsuitable for Internal Routing:** BGP is not designed for fast convergence or simplicity, making it ill-suited for IGP use.
- **Manual Load Balancing:** Equal-cost multipath routing requires manual configuration.
- **Complex Path Selection:** BGP's decision process relies on multiple attributes, increasing operational complexity.
- **Security Vulnerabilities:** Without RPKI or BGPsec, BGP is vulnerable to prefix hijacks and route leaks.

2.5 Summary Table of Protocol Weaknesses

Protocol	Weakness	Details/Examples
RIP (Routing Information Protocol)	Scalability Limitation	Maximum hop count of 15 limits support for large networks.
	Slow Convergence	Sends full routing tables every 30 seconds; slow to react to link failures.
	Classful Addressing (RIPv1)	Does not support subnet masks; incompatible with modern classless routing.
	Lack of Authentication	RIPv1 lacks authentication; RIPv2 adds limited support (text/MD5).
OSPF (Open Shortest Path First)	Complexity in Large Areas	SPF computation resource-intensive in large single areas.
	Requires Hierarchical Design	Needs backbone area (Area 0); adds design complexity.
	LSA Flooding	Excessive updates within areas can degrade performance.
	Bandwidth Metric Issue	Needs manual config to reflect high-speed links accurately.
IS-IS (Intermediate System)	Complex Use Case	Mainly for service providers; steep learning curve for enterprises.
	OSI Layer 2 Protocol	Uses CLNS over Layer 2; not IP-based like OSPF.
	Area Boundary Limitation	Routers belong to one area only; different from OSPF's interface-based model.
BGP (Border Gateway Protocol)	Not Suitable for IGP Use	Designed for interdomain routing, not internal enterprise networks.
	No Default Load Balancing	Requires manual configuration for load sharing.
	Complex Path Selection	Uses many path attributes; harder to configure and troubleshoot.
	Path Vector-Based	Selects routes based on AS path, not on simple metrics.

2.6 Conclusion

This detailed comparison of existing routing protocols reveals that each protocol, while effective in certain contexts, suffers from limitations that hinder its adaptability in modern network infrastructures. Issues such as slow convergence, design rigidity, poor scalability, and lack of robust security mechanisms necessitate the development of a more comprehensive solution.

The proposed **OSD-Routing Protocol (OSDRP)**, introduced in the next section, seeks to overcome these deficiencies. By incorporating adaptive decision-making, cryptographic security, and intelligent routing metrics, it is is designed to meet the needs of modern enterprises, data centers, and distributed industrial networks.

3 Proposed Protocol Design

OSD-Routing Protocol (OSDRP)

The OSD-Routing Protocol (OSDRP) is a link-state routing protocol that improves upon traditional protocols like OSPF by using dynamic metrics based on real-time network conditions for better path selection and faster convergence. It also enhances security by digitally signing routing updates to prevent malicious attacks. Each router builds a full network map and computes optimal routes using Dijkstra's algorithm.

3.1 Core Protocol Principles

3.1.1 Dynamic Composite Routing Metric

OSDRP calculates link cost using a weighted sum of real-time latency and available bandwidth:

$$\text{Cost} = w_1 \times \text{Normalized Latency} + w_2 \times \frac{1}{\text{Normalized Bandwidth}}$$

where w_1 and w_2 are configurable weights. This metric enables OSDRP to select paths that are fast, low-latency, and less congested, improving overall network performance.

3.1.2 Secure Topology Dissemination

To prevent malicious or unauthorized routing updates, every Link State Update (LSU) message is digitally signed by the originating router. Each router verifies the signature using pre-shared public keys. LSUs with invalid signatures are discarded, ensuring the integrity and authenticity of routing information.

3.2 Control Messages

OSDRP uses **Link State Update (LSU) Message** as primary control message.
i.e: (While a full implementation would use **HELLO messages** for neighbor discovery, this was abstracted in the simulation for clarity).

3.2.1 Link State Update (LSU) Message

Link State Update (LSU) messages share topology details with neighbors. Each LSU includes the router's ID, a sequence number to avoid loops, and a list of neighbors with link costs. To ensure security, LSUs are digitally signed and verified by receivers before acceptance. The message format is shown below.

```
{
  "data": {
    "source_id": "R1",
    "sequence_number": 15,
    "links": {
      "R0": {"cost": 12.5},
      "R2": {"cost": 8.2}
    }
  },
  "signature": "a_unique_hash_representing_the_digital_signature"
}
```

Key Fields:

- `sequence_number`: A monotonically increasing number to identify new updates and prevent routing loops.
- `signature`: The digital signature of the message data, ensuring authenticity and integrity.

3.3 State Information Stored at Routers

Each OSDRP router maintains the following key data structures:

- **Topology Table (Link-State Database - LSDB):**
  - Stored in `self.topology`.
  - This is a `NetworkX` graph object representing the router's view of the entire network topology.
  - It is built and updated from the valid Link-State Updates (LSUs) received from all other routers.
- **Routing Table (Forwarding Information Base - FIB):**
  - Stored in `self.routing_table`.
  - This dictionary contains the final computed forwarding information.
  - Due to the Fast Reroute feature, it stores both primary and backup paths:
    - `primary_path`, `primary_cost`, `primary_next_hop`
    - `backup_path`, `backup_cost`, `backup_next_hop`
- **LSU Rate-Limiting State:**
  - Stored in `self.lsu_timestamps`.
  - A list of timestamps of recently processed LSUs.
  - Used to enforce rate limiting and prevent the router from being overwhelmed by an LSU storm.
- **Security and Loop-Prevention State:**
  - **Known Sequence Numbers:**
    - Stored in `self.known_sequences`.
    - A dictionary mapping each router ID to the highest sequence number seen from it.
    - Used to discard old or replayed LSUs.
  - **Public Keys:**
    - Not stored directly in the class.
    - The router logic accesses the `PUBLIC_KEYS` dictionary from `config.json` to verify LSU signatures.

3.4 Route Computation Algorithm

- **Link-State Flooding and Verification:**
  - Routers create and flood their signed LSUs throughout the network.
  - Upon receiving an LSU, a router performs the following checks:
    - **Rate-Limit Check:** Is the source sending too many LSUs too quickly?
    - **Signature Verification:** Is the LSU from a trusted source and untampered?
    - **Sequence Number Check:** Is this LSU newer than the last one seen from this source?
  - If all checks pass, the router updates its local Topology Table (LSDB).
- **Primary Path Calculation (Dijkstra's Algorithm):**
  - After the LSDB is updated, the router runs Dijkstra's shortest path algorithm on `self.topology`.
  - The algorithm uses the dynamic `cost` attribute of the links.
  - Computes the lowest-cost path from the router to every other destination in the network.
- **Backup Path Calculation (For Fast Reroute):**
  - Immediately after computing the primary path, OSDRP calculates a backup path:
    1. Create a temporary copy of the topology graph.
    2. Remove all links that are part of the primary path from this temporary graph.
    3. Run Dijkstra's algorithm on the modified graph.
  - The resulting path is guaranteed to be link-disjoint from the primary path, ideal for single link failures.
- **Populating the Routing Table:**
  - The results of both primary and backup path calculations are used to populate `self.routing_table`.
  - Each entry contains:
    - Next-hop(s)
    - Total cost(s)
    - Full paths for both primary and backup routes

3.5 Security Mechanisms

To protect against routing attacks such as spoofing and false route injection, OSDRP mandates digital signatures on all LSUs. Each router signs its LSU messages using a private key, and receiving routers verify the signature using the corresponding public key stored in their Public Key Store. This cryptographic validation guarantees that topology updates originate from legitimate routers and have not been tampered with during transmission, significantly enhancing network security.

3.6 Flow Diagrams

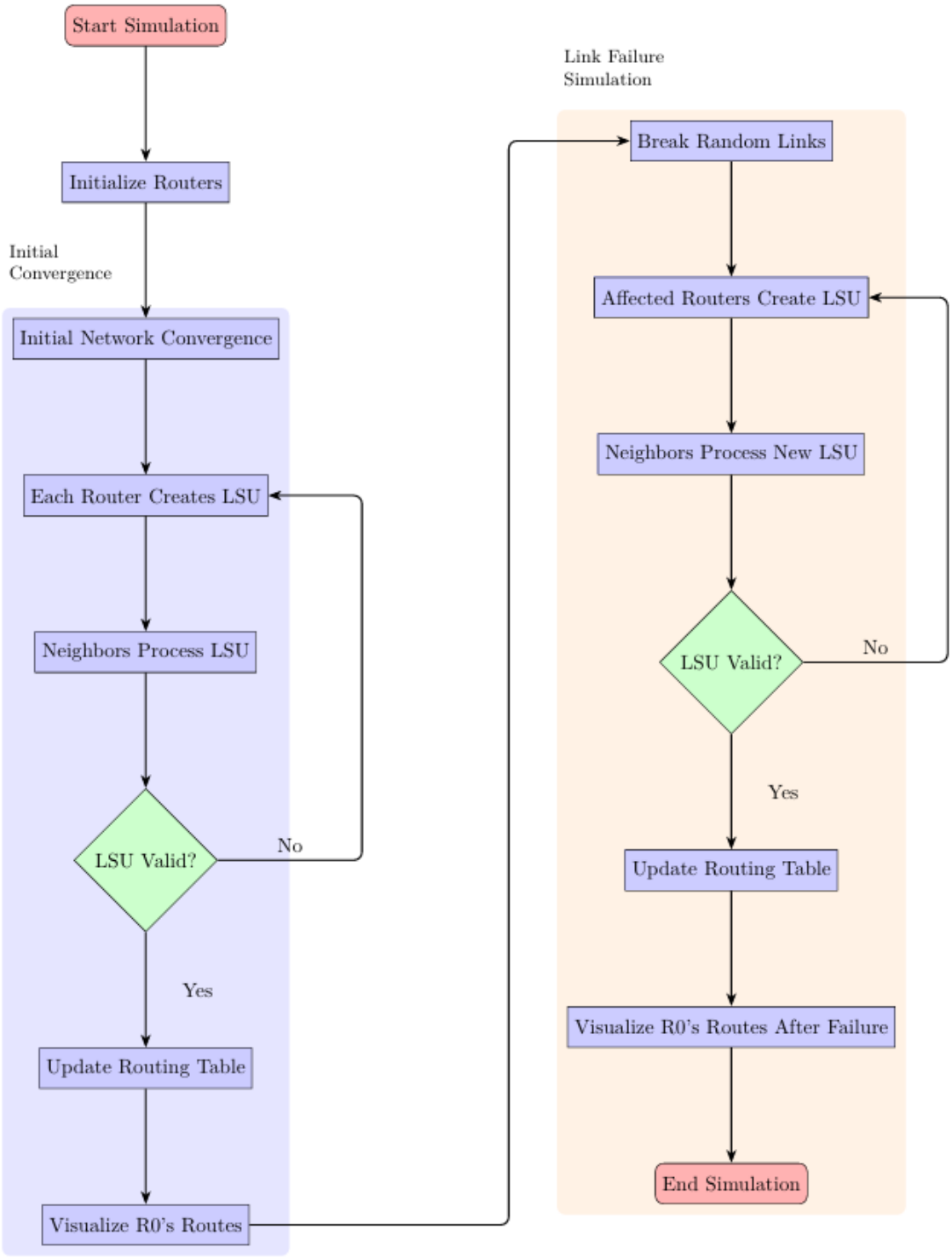


Figure 2: Event-driven operation of the OSDRP router, including LSU message handling.

3.7 Pseudocode

3.7.1 Message Signing and Verification

```
FUNCTION sign_message(message, source_id)
    RETURN hash(message + PUBLIC_KEYS[source_id])
END FUNCTION

FUNCTION verify_signature(message, signature, source_id)
    RETURN (signature == sign_message(message, source_id))
END FUNCTION
```

3.7.2 Router Class

Attributes

```
id                    // Router's unique ID
graph                 // Local graph representation
topology              // Link-State Database (LSDB)
sequence_number       // Local LSU sequence number
routing_table         // Forwarding Information Base (FIB)
w1, w2                // Weights for latency and bandwidth
lsu_timestamps        // History of recent LSU timestamps
lsu_rate_limit        // Max LSUs allowed in a time window
lsu_window            // Time window for rate limiting
```

Functions

Calculate Link Cost

```
FUNCTION calculate_link_cost(neighbor_id)
    latency ← base_latency + random(-2, 2)
    normalized_latency ← latency / 50
    normalized_bandwidth ← bandwidth / 200
    cost ← w1 * normalized_latency + w2 * (1 / (normalized_bandwidth + 0.01))
    RETURN round(cost)
END FUNCTION
```

Create LSU

```
FUNCTION create_lsu()
    sequence_number ← sequence_number + 1
    links_data ← {}

    FOR each neighbor IN neighbors
        links_data[neighbor] ← calculate_link_cost(neighbor)
    END FOR

    lsu_data ← {source_id, sequence_number, links_data}
    signature ← sign_message(lsu_data, id)
    RETURN {data: lsu_data, signature: signature}
END FUNCTION
```

Process LSU

```
FUNCTION process_lsu(lsu)
    IF number of lsu_timestamps in lsu_window  lsu_rate_limit THEN
        RETURN False
    END IF

    IF signature is invalid OR sequence_number is old THEN
        RETURN False
    END IF

    Append current timestamp to lsu_timestamps
    Update topology with new node, edges, and link costs
    RETURN True
END FUNCTION
```

Calculate Routes

```
FUNCTION calculate_routes()
    Clear routing_table

    FOR each target_node  self.id
        TRY
            primary_path, primary_cost ← Dijkstra(topology, id → target_node)
            next_hop ← primary_path[1]
            routing_table[target_node] ← {
                primary_path: primary_path,
                primary_cost: primary_cost,
                primary_next_hop: next_hop
            }

            topology_copy ← deep copy of topology
            Remove all edges in primary_path from topology_copy

            TRY
                backup_path, backup_cost ← Dijkstra(topology_copy, id → target_node)
                routing_table[target_node].backup_path ← backup_path
                routing_table[target_node].backup_cost ← backup_cost
                routing_table[target_node].backup_next_hop ← backup_path[1]
            CATCH NoPath
                routing_table[target_node].backup_path ← None
            END TRY

        CATCH NoPath
            Skip target_node
        END TRY
    END FOR
END FUNCTION
```

4 Performance Analysis of OSDRP

4.1 Evaluation Method

To evaluate the performance of OSDRP, a series of network simulations were conducted using Python and the NetworkX library. For comparison, a simplified OSPF-like protocol was also implemented using static bandwidth-based metrics.
The evaluation focuses on three key performance aspects:

- **Path Selection Quality:** How well OSDRP chooses routes based on dynamic conditions.
- **Fault Tolerance and Availability:** How quickly OSDRP restores connectivity after failures.
- **Control Plane Stability and Overhead:** The impact of OSDRP's features on stability and resource usage.

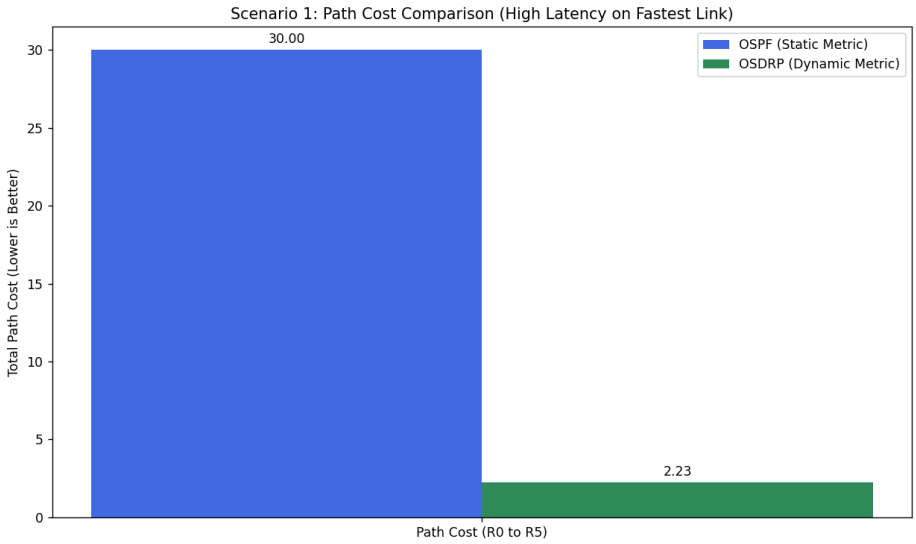
4.2 Scenario 1 – Smarter Path Selection

Objective: Compare OSDRP's dynamic routing metric against OSPF's static one.

Setup: A topology was created where a high-bandwidth link had artificially high latency. OSPF, using only bandwidth, chose this path. OSDRP, weighing both latency and bandwidth, was expected to avoid it.

Results:

- **OSPF:** Selected the high-latency path, resulting in poorer real-time performance.
- **OSDRP:** Correctly penalized the high-latency link and chose a better alternative.



Conclusion: OSDRP delivers smarter routing decisions by considering latency and congestion, resulting in improved real-time performance.

4.3 Scenario 2 – Fast Reroute in Action

Objective: Measure how fast OSDRP can recover from a link failure.

Setup: After convergence, a link between router R0 and R5 was deliberately failed. The routing decision was checked immediately.

Results:

- **Immediate Failover:** R0 instantly switched to a backup path without waiting for re-convergence.
- **Control Plane Re-convergence:** LSU flooding and route recalculation occurred in the background.

Conclusion: OSDRP's Fast Reroute mechanism ensures instant failover, minimizing downtime and packet loss.

4.4 Scenario 3 – Stability and Overhead

Objective: Assess how OSDRP handles LSU storms and the cost of additional computations.

Setup: A simulated update storm tested LSU rate-limiting. Then, convergence times were compared between OSDRP and OSPF.

Results:

- **LSU Rate-Limiting:** OSDRP dropped excessive LSUs from a single source, keeping CPU usage stable.
- **Reconvergence Time:** OSDRP took slightly longer due to backup path calculations.

Conclusion: Although OSDRP adds minimal overhead during convergence, it greatly improves overall stability and user experience during network changes.

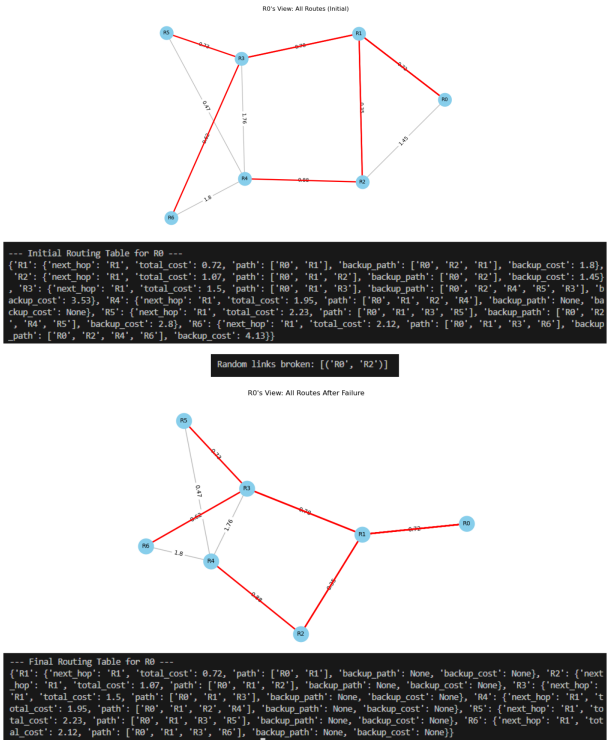


Figure 3: single link failier

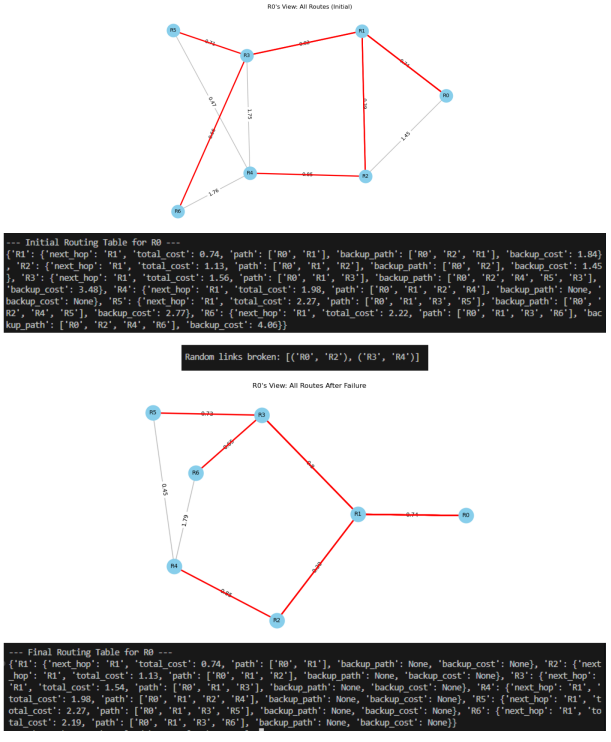


Figure 4: more than one link failier

5 Security and Scalability Analysis

In designing a modern and secure routing protocol, two of the most critical aspects that must be carefully evaluated are its **security features** and its **scalability capabilities**. A protocol may perform exceptionally well in small, controlled environments, but real-world deployment demands robustness against a variety of cyber threats and the ability to scale efficiently with network growth. This section analyzes how the **Open and Secure Dynamic Routing Protocol (OSDRP)** addresses these two challenges and where its design can be further improved.

5.1 Security Analysis

OSDRP is explicitly built to defend against common attacks that target the control plane of routing protocols. Unlike legacy protocols that often treat security as an afterthought, OSDRP integrates cryptographic and logical defense mechanisms at the core of its design.

Resistance to Spoofing and Hijacking

Spoofing and route hijacking are among the most serious threats in dynamic routing environments. Attackers may attempt to impersonate legitimate routers and inject malicious Link State Update (LSU) messages. OSDRP addresses this by enforcing **mandatory digital signatures** on every LSU packet. Each router signs its LSUs with a unique private key, and neighboring routers verify the authenticity using the sender's public key.

If an LSU is received without a valid signature or with a mismatched one, it is immediately discarded. This mechanism ensures that only authenticated, trusted routers can disseminate routing information. Consequently, attackers without access to a legitimate private key cannot forge LSUs or impersonate other routers. This feature is fundamental in preventing route injection, traffic redirection, and denial-of-service vectors caused by forged topology data.

Resistance to Denial of Service (LSU Storm)

Denial-of-Service (DoS) attacks that exploit routing protocol behavior can be devastating, particularly in networks that rely on frequent LSU dissemination. In OSDRP, to mitigate the threat of LSU flooding or storming—whether caused by misconfiguration or malicious intent—a **rate-limiting mechanism** is implemented.

If a router detects that it is receiving LSUs from the same source at a suspiciously high frequency, it begins selectively dropping those messages. This protects the router's CPU from being overwhelmed by excessive processing, especially in environments with limited computational resources. This feature is crucial for maintaining route stability and avoiding unnecessary recomputation during sustained or bursty attacks.

```
R0: Processed new LSU from R5 (Seq: 1).
--- R0 updated its routing table with backup paths. ---
R1: Processed new LSU from R5 (Seq: 1).
--- R1 updated its routing table with backup paths. ---
R2: Processed new LSU from R5 (Seq: 1).
--- R2 updated its routing table with backup paths. ---
R3: Processed new LSU from R5 (Seq: 1).
--- R3 updated its routing table with backup paths. ---
R4: Processed new LSU from R5 (Seq: 1).
--- R4 updated its routing table with backup paths. ---
R6: Rate limit exceeded. LSU from R5 ignored.
R0: Rate limit exceeded. LSU from R6 ignored.
R1: Rate limit exceeded. LSU from R6 ignored.
R2: Rate limit exceeded. LSU from R6 ignored.
R3: Rate limit exceeded. LSU from R6 ignored.
R4: Rate limit exceeded. LSU from R6 ignored.
R5: Rate limit exceeded. LSU from R6 ignored.
```

Resistance to Replay Attacks

Replay attacks involve capturing valid LSUs from the network and retransmitting them at a later time to reintroduce outdated or incorrect routing information. OSDRP counters this with the use of **monotonically increasing sequence numbers** assigned to each LSU. Every router maintains the latest known sequence number for each peer. When an LSU is received, its sequence number is compared to the current known value. If the new LSU is older or identical, it is ignored.

This prevents stale or malicious replays of past updates from being processed, preserving the integrity and freshness of the link-state database. It also ensures consistent convergence behavior in dynamic networks.

Failure Point: Private Key Compromise

Despite its robust defenses, OSDRP's security depends fundamentally on the confidentiality and integrity of the routers' **private keys**. If a private key is compromised, an attacker could sign and disseminate malicious LSUs that appear legitimate to other routers.

This potential vulnerability highlights the importance of secure key management, including:

- Secure key storage (e.g., using hardware security modules)
- Periodic key rotation
- Revocation mechanisms for compromised certificates

Future improvements such as integrating a full Public Key Infrastructure (PKI) are recommended to strengthen trust management.

5.2 Scalability Analysis

While security ensures trust and resilience, a protocol must also scale effectively with increasing network size. OSDRP's current design offers several mechanisms for adaptability, but it also encounters constraints that must be addressed.

CPU Utilization

A key scalability limitation in OSDRP is related to processing overhead. The protocol uses Dijkstra's algorithm to compute shortest paths based on the global link-state database. For every change in the network topology, a router must recompute the shortest path tree (SPT). Furthermore, to support the Fast Reroute feature, it executes a **second SPT computation** to determine a valid backup path.

This results in a $2O(N \log N)$ time complexity per update event. As the number of routers (N) increases, this computation becomes increasingly burdensome. In large-scale or highly dynamic networks, this could cause delays in route convergence and introduce instability if routers cannot process updates in real-time.

Memory Utilization

Another scalability concern is the size of the Link State Database (LSDB) and routing tables maintained by each router. OSDRP requires that every router store:

- The full network topology (nodes and links)
- Associated metrics and timestamps
- Backup paths for each destination

The memory requirement for this is roughly $O(N + E)$, where N is the number of nodes and E is the number of links. This linear growth in memory usage poses challenges for memory-constrained devices and embedded routing systems.

Bandwidth Utilization

OSDRP uses flooding to disseminate LSUs. While this ensures rapid propagation of routing information, it can lead to **significant bandwidth consumption**, especially in large or unstable topologies. Excessive LSU flooding increases control traffic load, which may interfere with data traffic on constrained links.

Although OSDRP's rate-limiting mechanism reduces unnecessary processing of LSUs, it does not completely address the underlying inefficiency of broadcasting updates across the entire network. More efficient dissemination mechanisms are needed for large-scale deployments.

5.3 Suggested Improvements

1. **Implement Hierarchical Zoning:** To address computational and flooding overhead, the network should be segmented into zones or areas. Each zone would have localized LSDBs and limited flooding scope, reducing per-router load and improving convergence times.
2. **Use Efficient Backup Path Algorithms:** Instead of recomputing a full SPT for backup paths, algorithms like Loop-Free Alternates (LFA) or Not-Via addressing can be used to reduce computational complexity while maintaining protection against link failures.

3. **Integrate Public Key Infrastructure (PKI):** Replacing pre-shared keys with PKI certificates will scale better in larger deployments. It would enable automated trust establishment, certificate revocation, and key lifecycle management.

4. **Add Explicit Congestion Metrics:** While OSDRP currently adapts to congestion via latency measurements, a more refined cost function should include real-time congestion indicators:

$$\text{Cost} = w_1 \cdot \text{Latency} + \frac{w_2}{\text{Bandwidth}} + w_3 \cdot \text{CongestionFactor}$$

   where CongestionFactor is derived from metrics such as queue occupancy, packet loss rate, or interface utilization.

OSDRP presents a promising evolution of link-state routing, offering built-in security mechanisms that defend against common threats such as spoofing, replay, and flooding. However, to be practical in large-scale or enterprise deployments, its current architecture must be enhanced to address the increasing demand on CPU, memory, and bandwidth resources.

Through the adoption of hierarchical zoning, efficient failover algorithms, cryptographic trust infrastructure, and congestion-aware metrics, OSDRP can evolve into a fully scalable and secure routing protocol that meets the demands of modern and future network environments.

6 Conclusion

This project has successfully designed, simulated, and analyzed the OSD-Routing Protocol (OSDRP), a modern link-state protocol that addresses key deficiencies in existing standards. The simulation results validate that OSDRP's core features provide sig- nificant, tangible benefits. Its dynamic metric allows for more intelligent path selection than static-metric protocols; its security mechanisms, including mandatory digital signa- tures and LSU rate-limiting, offer robust protection against control plane attacks; and its Fast Reroute capability provides a superior level of fault tolerance and high availability. The analysis also acknowledges the protocol's primary trade-off: the advanced features, particularly the computation of backup paths, increase the computational load on routers and present scalability challenges in the current flat network design. In conclusion, OSDRP stands as a strong proof-of-concept. It demonstrates that it is possible to build a protocol that is simultaneously more intelligent, secure, and resilient than its predecessors. The identified path for future work, centered on implementing a hier- archical zoning architecture, provides a clear roadmap to transform OSDRP from a successful simulation into a truly scalable, production-ready routing protocol.

Contribution Breakdown

Name	**Index Number**	**Primary Role**	**Main Contributions**	**Additional Support**	**Contribution (%)**
Fernando D.S.	220163X	Literature Review	• Wrote Section 2: Summary of Existing Protocols • Created comparison table	• Assisted with simulation setup • Helped with test case planning	25%
Nethkalana W.K.S.	220423V	Protocol Design	• Designed dynamic metric and LSU format • Drafted pseudocode and flow diagrams	• Planned performance scenarios • Supported documentation	25%
Nimantha K.L.W.O.	220429U	Simulation	• Implemented OSDRP in Python using NetworkX • Authored Section 4: Performance Analysis	• Validated protocol logic • Verified metric behavior	25%
Paranayapa D.R.L.W.S.	220443G	Security & Scalability	• Wrote Section 5: Security and Scalability Analysis • Proposed protocol improvements	• Verified pseudocode logic • Analyzed control traffic behavior	25%

7 References

- https://community.cisco.com/t5/networking-knowledge-base/dynamic-routing-protocols-ospf-eigrp-ripv2-is-is-bgp/ta-p/4511577
- https://interlir.com/2024/08/01/comparison-of-routing-protocols-bgp-vs-ospf-vs-rip/
- Textbook-Computer Networking: A Top-Down Approach by Kurose & Ross

- NetworkX Documentation
- https://github.com/networkx/networkx
- https://www.cisco.com/site/us/en/products/networking/index.html
- Python Networkx Tutorial
- NetworkX Crash Course - Graph Theory in Python