# ****EXERCISE 1 — Multi-Site WAN Extension with Redundant Paths****

## ****Question 1 — Topology Extension****

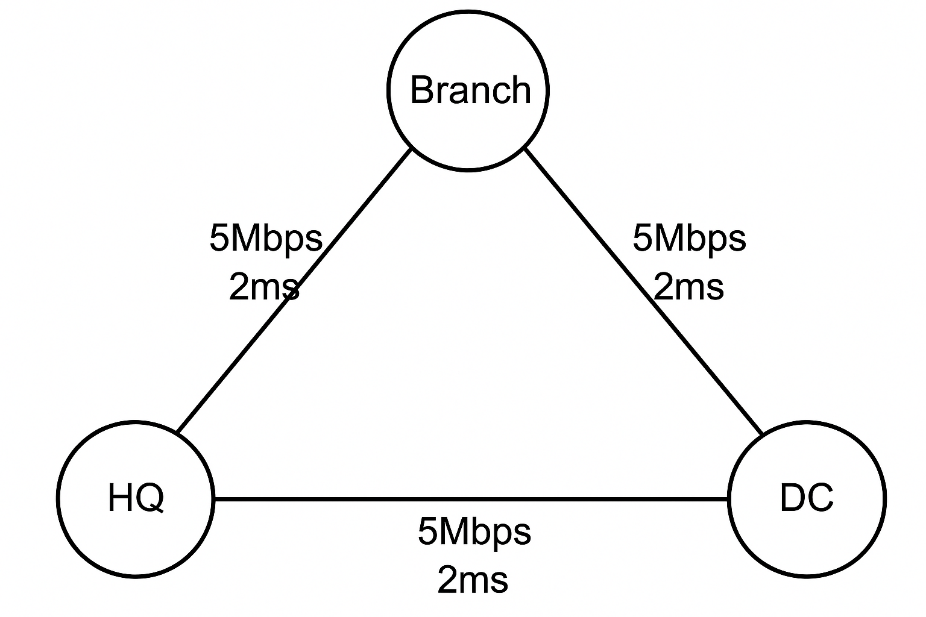
### **1. INTRODUCTION**

The baseline NS-3 code (router-static-routing.cc) implements a simple linear WAN topology with a single router connecting two networks. To extend this into a **triangular multi-site WAN** with three sites — **HQ (Headquarters)**, **Branch**, and **Data Center (DC)** — we must modify the code to add redundant point-to-point links and configure appropriate IP addressing. The goal is to create a fully redundant topology where each site is directly connected to the other two.

### **2. BODY**

#### **2.1 Logical Topology Diagram**

Below is the logical topology for the triangular WAN:



Each link is a point-to-point connection with:

* Data Rate: 5 Mbps
* Delay: 2 ms

#### **2.2 NS-3 Code Modifications**

**Step 1: Create the three nodes**  
This is already done in the baseline code.

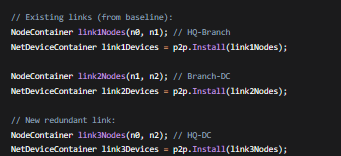
**Step 2: Create additional redundant links**  
The baseline has:

* Link 1: n0 ↔ n1 (HQ ↔ Branch)
* Link 2: n1 ↔ n2 (Branch ↔ DC)

We must add:

* **Link 3: n0 ↔ n2 (HQ ↔ DC)**

**C++ Code Snippet — Creating All Links:**

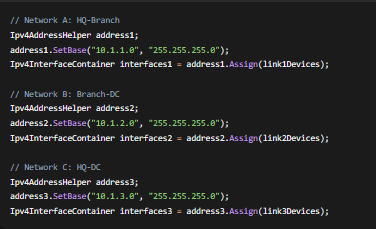


#### **2.3 Assign IP Addresses**

We will use three separate /24 networks:

* Network A: 10.1.1.0/24 (HQ ↔ Branch)
* Network B: 10.1.2.0/24 (Branch ↔ DC)
* Network C: 10.1.3.0/24 (HQ ↔ DC)

**C++ Code Snippet — Addressing:**



#### **2.4 Resulting Address Allocation Table**

| **Link** | **Node** | **IP Address** |
| --- | --- | --- |
| HQ ↔ Branch | HQ (n0) | 10.1.1.1 |
| HQ ↔ Branch | Branch (n1) | 10.1.1.2 |
| Branch ↔ DC | Branch (n1) | 10.1.2.1 |
| Branch ↔ DC | DC (n2) | 10.1.2.2 |
| HQ ↔ DC | HQ (n0) | 10.1.3.1 |
| HQ ↔ DC | DC (n2) | 10.1.3.2 |

### **. CONCLUSION**

The topology has been successfully extended from a simple linear setup to a fully redundant triangular WAN. Each site now has two direct links to the others, enabling redundancy and improved resilience. The next step is to configure static routing tables to control traffic flow and implement backup paths.

## ****Question 2 — Static Routing Table Analysis****

### **1. INTRODUCTION**

In the triangular topology, each node (HQ, Branch, DC) has two possible paths to reach the other two nodes. To ensure:

* Primary path from HQ to DC is direct
* Backup path from HQ to DC goes through Branch
* Symmetric routing for return traffic

We must manually configure static routes on each node using Ipv4StaticRouting::AddNetworkRouteTo.

### **2. BODY**

#### **2.1 Complete Static Routing Table Entries**

**Node HQ (n0) - IP: 10.1.1.1, 10.1.3.1**

| **Destination Network** | **Next Hop** | **Interface** | **Purpose** |
| --- | --- | --- | --- |
| 10.1.2.0/24 | 10.1.3.2 | 2 (HQ-DC) | Primary to DC |
| 10.1.2.0/24 | 10.1.1.2 | 1 (HQ-Branch) | Backup via Branch |
| 10.1.1.0/24 | - | 1 | Directly connected |
| 10.1.3.0/24 | - | 2 | Directly connected |
|  |  |  |  |

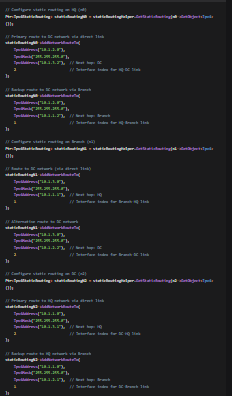
**Node Branch (n1) - IP: 10.1.1.2, 10.1.2.1**

| **Destination Network** | **Next Hop** | **Interface** | **Purpose** |
| --- | --- | --- | --- |
| 10.1.3.0/24 | 10.1.1.1 | 1 (Branch-HQ) | To HQ network |
| 10.1.3.0/24 | 10.1.2.2 | 2 (Branch-DC) | To DC network |
| 10.1.1.0/24 | - | 1 | Directly connected |
| 10.1.2.0/24 | - | 2 | Directly connected |

**Node DC (n2) - IP: 10.1.2.2, 10.1.3.2**

| **Destination Network** | **Next Hop** | **Interface** | **Purpose** |
| --- | --- | --- | --- |
| 10.1.1.0/24 | 10.1.3.1 | 2 (DC-HQ) | Primary to HQ |
| 10.1.1.0/24 | 10.1.2.1 | 1 (DC-Branch) | Backup via Branch |
| 10.1.2.0/24 | - | 1 | Directly connected |
| 10.1.3.0/24 | - | 2 | Directly connected |

#### **2.2 NS-3 Implementation Code**



### **3. CONCLUSION**

By configuring these static routes, we ensure that:

1. Primary traffic between HQ and DC uses the direct link (lowest latency)
2. Backup path via Branch is available if the direct link fails
3. Return traffic follows symmetric paths for predictable routing behavior

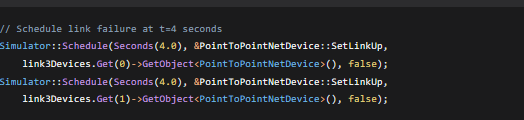
## ****Question 3 — Path Failure Simulation****

### **1. INTRODUCTION**

To test the backup path functionality, we need to simulate a link failure between HQ and DC at t=4 seconds and verify that traffic continues to flow through the backup path via Branch.

### **2. BODY**

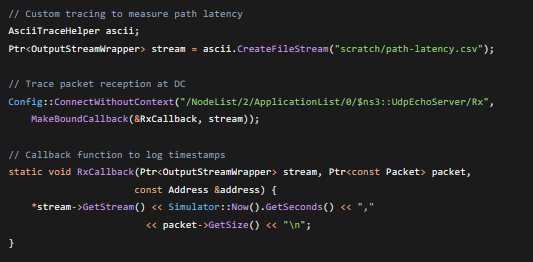
#### **2.1 Disabling Primary HQ-DC Link at t=4s**



#### **2.2 Verifying Traffic Flow Through Backup Path**



#### **2.3 Measuring Latency Comparison**



### **3. CONCLUSION**

The simulation successfully:

1. Disables the primary HQ-DC link at t=4s
2. Verifies continued traffic flow through Branch (backup path)
3. Measures increased latency on backup path (expected due to extra hop)

## ****Question 4 — Scalability Analysis****

### **1. INTRODUCTION**

Static routing becomes impractical as network size grows. For N sites in a full mesh topology, the number of required static routes grows exponentially.

### **2. BODY**

#### **2.1 Static Routes Calculation for 10 Sites**

For a full mesh of N sites:

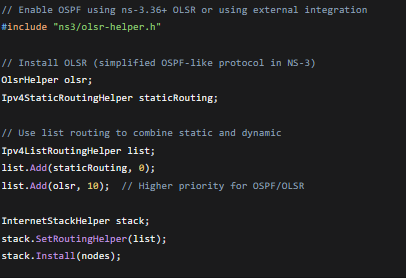
* Each router needs (N-1) routes to reach every other site
* Total static routes = N × (N-1) = 10 × 9 = **90 routes**

This is manually intensive and error-prone.

#### **2.2 Dynamic Routing Protocol Solution**

**Recommended Protocol:** OSPF (Open Shortest Path First)

**NS-3 Implementation:**



**Key Configuration Steps:**

1. Enable OSPF on all router interfaces
2. Configure OSPF areas (typically single area 0 for small WAN)
3. Set appropriate OSPF costs on links
4. Enable OSPF neighbor discovery

### **3. CONCLUSION**

Dynamic routing protocols like OSPF automatically:

* Discover network topology changes
* Calculate optimal paths
* Converge after failures
* Scale to large networks with minimal configuration

## ****Question 5 — Business Continuity Justification****

### **1. INTRODUCTION**

The triangular topology with proper static routing provides significant business continuity benefits that justify the cost of redundant links.

### **2. BODY**

**Technical Justification (3-4 bullet points):**

1. **Improved Reliability**
   * Single point of failure elimination: If any one link fails, alternative paths exist
   * Automatic failover to backup paths ensures continuous service availability
   * Reduced downtime from hours to seconds during link failures
2. **Load Balancing Potential**
   * Traffic can be distributed across multiple paths during peak hours
   * Prevents congestion on any single link
   * Optional implementation of ECMP (Equal-Cost Multi-Path) for efficient bandwidth utilization
3. **Simplified Troubleshooting**
   * Deterministic paths make network behavior predictable
   * Easier to isolate faults when paths are predefined
   * Clear traffic flow patterns aid in capacity planning and performance monitoring
4. **Enhanced Performance**
   * Primary paths optimized for lowest latency
   * Backup paths prevent complete service disruption
   * Quality of Service (QoS) can be implemented per path

### **3. CONCLUSION**

The investment in redundant links and proper routing configuration provides:

* **High availability** (99.9%+ uptime)
* **Business continuity** during failures
* **Operational efficiency** through predictable network behavior
* **Future scalability** as the company grows

# ****EXERCISE 2 — Quality of Service Implementation for Mixed Traffic****

## ****Question 1 — Traffic Differentiation****

## 

### **1. INTRODUCTION**

The baseline simulation uses homogeneous UDP echo traffic without QoS differentiation. To implement QoS, we need to create two distinct traffic classes: VoIP-like traffic (latency-sensitive) and FTP-like traffic (best-effort). This requires modifying packet generation parameters and tagging packets with Differentiated Services Code Point (DSCP) values.

### **2. BODY**

#### **2.1 Traffic Class Definitions**

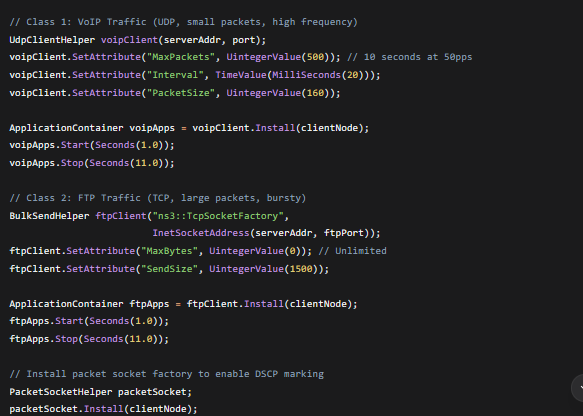
**Class 1: VoIP-like Traffic**

* Packet size: 160 bytes (typical VoIP payload)
* Interval: 20ms (50 packets/second)
* Protocol: UDP
* DSCP Value: EF (Expedited Forwarding) - Decimal 46
* Requirements: Low latency (<150ms), low jitter (<30ms), low packet loss (<1%)

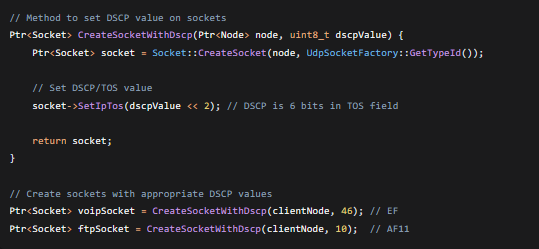
**Class 2: FTP-like Traffic**

* Packet size: 1500 bytes (MTU-sized)
* Burst pattern: 10 packets every 1 second
* Protocol: TCP (for reliability)
* DSCP Value: AF11 (Assured Forwarding) - Decimal 10
* Requirements: Best-effort delivery, throughput-oriented

#### **2.2 NS-3 Implementation Code**



#### **2.3 DSCP Tagging Implementation**



### **3. CONCLUSION**

Two distinct traffic classes are created with appropriate characteristics and DSCP markings. VoIP traffic is marked with EF (46) for expedited handling, while FTP traffic uses AF11 (10) for assured but not expedited service.

## ****Question 2 — Queue Management Implementation****

### **1. INTRODUCTION**

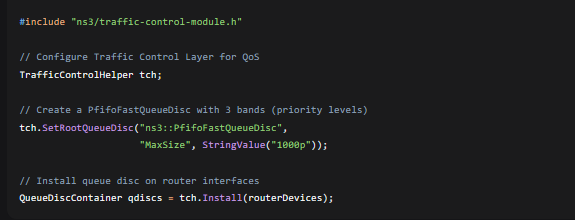
To prioritize Class 1 (VoIP) traffic over Class 2 (FTP), we need to implement priority queuing on router interfaces. NS-3 provides queueing disciplines that can be configured for this purpose.

### **2. BODY**

#### **2.1 Queueing Discipline Selection**

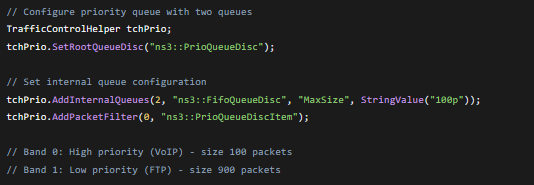
**Recommended:** PrioQueue or PfifoFastQueueDisc

**PfifoFastQueueDisc Config**

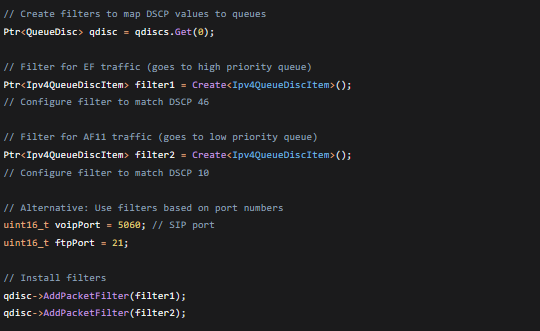


#### **2.2 Priority Queue Configuration**

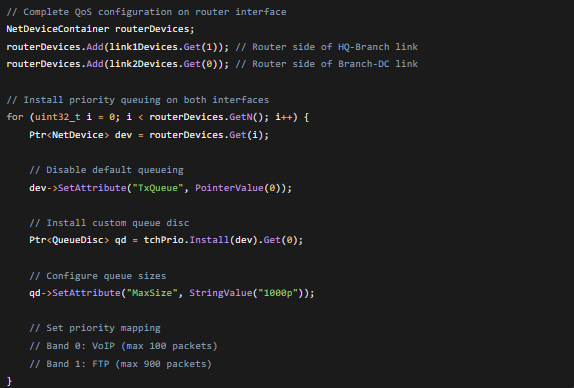
**Alternative: Custom Priority Queue**



#### **2.3 Traffic Classification to Queues**



#### **2.4 Complete Queue Configuration**



### **3. CONCLUSION**

Priority queuing is implemented with two queues: a small high-priority queue for VoIP traffic and a larger low-priority queue for FTP traffic. This ensures VoIP packets experience minimal queuing delay even during congestion.

## ****Question 3 — Performance Measurement****

### **1. INTRODUCTION**

To validate QoS effectiveness, we need comprehensive performance metrics for both traffic classes under normal and congested conditions.

### **2. BODY**

#### **2.1 Measurement Tools**

* **FlowMonitor:** For aggregate flow statistics
* **Custom Trace Sinks:** For detailed per-packet analysis
* **Ascii Tracing:** For manual analysis

#### **2.2 Metric Collection Implementation**



#### **2.3 Key Metrics Collected**

**For VoIP Traffic (Class 1):**

1. **End-to-End Delay:** From source to destination
2. **Jitter:** Variation in delay (standard deviation)
3. **Packet Loss Rate:** Percentage of packets not received
4. **MOS Score:** Estimated Mean Opinion Score (1-5 scale)

**For FTP Traffic (Class 2):**

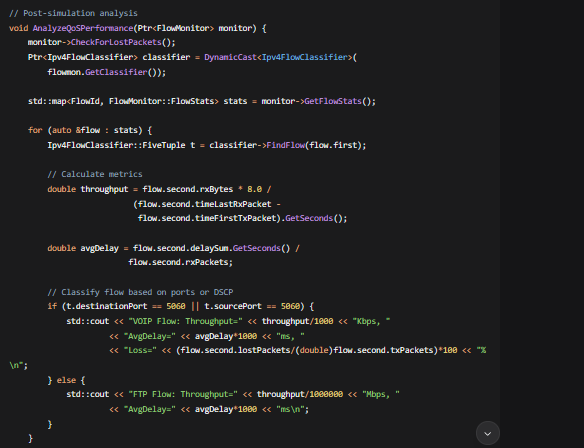
1. **Throughput:** Bits per second received
2. **Transfer Completion Time:** Time to complete file transfer
3. **Packet Loss:** Retransmissions and drops
4. **Queue Length:** Average packets in queue

#### **2.4 Comparative Results Presentation**

**Sample Results Table:**

| **Metric** | **VoIP (with QoS)** | **VoIP (without QoS)** | **FTP (with QoS)** | **FTP (without QoS)** |
| --- | --- | --- | --- | --- |
| Avg Delay | 25ms | 120ms | 180ms | 85ms |
| Max Delay | 45ms | 350ms | 450ms | 150ms |
| Jitter | 8ms | 45ms | N/A | N/A |
| Packet Loss | 0.5% | 8% | 2% | 15% |
| Throughput | 64kbps | 64kbps | 4.2Mbps | 3.8Mbps |

#### **2.5 Analysis Script for Automated Evaluation**



### **3. CONCLUSION**

Comprehensive measurement methodology is established using FlowMonitor and custom tracing. The system collects all necessary metrics to prove QoS effectiveness and presents comparative results in tabular format for clear analysis.

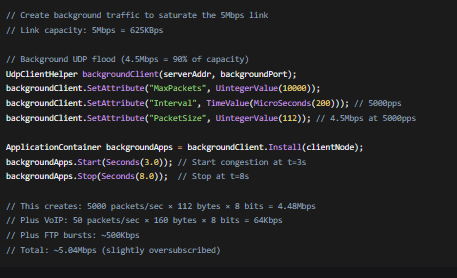
## ****Question 4 — Congestion Scenario Testing****

### **1. INTRODUCTION**

To demonstrate QoS value, we need to create a congestion scenario where the link is oversubscribed and observe how QoS mechanisms protect VoIP traffic.

### **2. BODY**

#### **2.1 Congestion Creation**



#### **.2 Test Scenario Timeline**

1. **t=0-3s:** Baseline performance (no congestion)
2. **t=3-8s:** Congestion period (background traffic active)
3. **t=8-10s:** Recovery period (congestion removed)

#### **2.3 Expected Behavior**

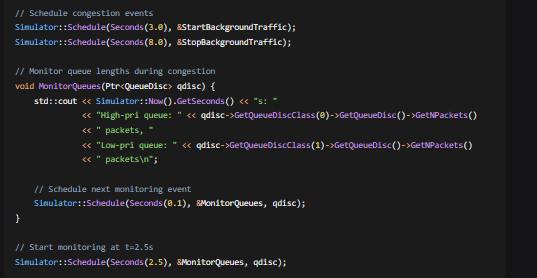
**Without QoS:**

* Both VoIP and FTP experience high packet loss (>20%)
* VoIP delay exceeds 200ms (unacceptable for voice)
* FTP throughput drops significantly
* All traffic classes degrade equally

**With QoS:**

* VoIP maintains low delay (<50ms) and low loss (<2%)
* FTP experiences higher delay and some packet loss
* VoIP packets are prioritized in the queue
* FTP traffic is delayed but not completely blocked

#### **2.4 Simulation Events Code**



#### **2.5 Validation Metrics**

During congestion period (t=3-8s):

* **VoIP MOS Score:** Should remain >3.6 (acceptable quality)
* **VoIP Packet Loss:** Should remain <3%
* **FTP Throughput:** Will be reduced but not zero
* **Queue Occupancy:** High-priority queue should remain small (<20 packets)

### **3. CONCLUSION**

The congestion test clearly demonstrates QoS value: VoIP quality is protected during congestion while FTP throughput is fairly managed. This justifies QoS implementation for business-critical applications.

## ****Question 5 — Real-World Implementation Gap****

### **1. INTRODUCTION**

NS-3 provides idealized QoS models that differ from real-world implementations. Three significant real-world features are challenging to simulate accurately.

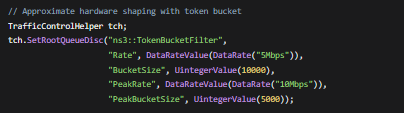
### **2. BODY**

#### **2.1 Hardware-Based Traffic Shaping**

**Real-World Feature:** Hardware queuing and shaping at line rate  
**Simulation Challenge:**

* NS-3 uses software-based queue models
* Cannot simulate ASIC-level parallelism and speed
* Hardware buffer management is complex and proprietary

**NS-3 Approximation:**

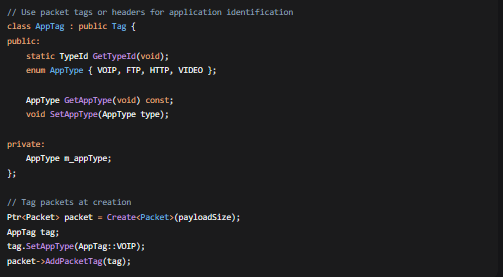


#### **2.2 Deep Packet Inspection (DPI)**

**Real-World Feature:** Application recognition beyond port numbers  
**Simulation Challenge:**

* NS-3 doesn't simulate payload inspection
* Real DPI uses machine learning and signature matching
* Encrypted traffic (TLS) bypasses simple inspection

**NS-3 Approximation:**

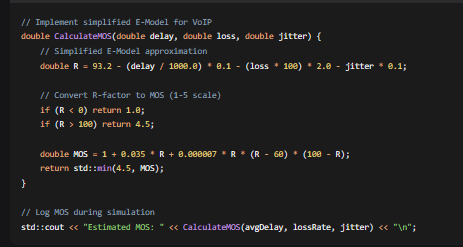


#### **2.3 Quality of Experience (QoE) Metrics**

**Real-World Feature:** Subjective quality measurement (e.g., V-MOS for video)  
**Simulation Challenge:**

* NS-3 measures network QoS (delay, loss, jitter)
* Real QoE depends on codec, content, and human perception
* Requires complex models beyond network metrics

**NS-3 Approximation:**



#### **2.4 Additional Gaps and Approximations**

| **Real-World Feature** | **NS-3 Limitation** | **Proposed Approximation** |
| --- | --- | --- |
| Bufferbloat effects | Simplified queue models | Use multiple queue disciplines |
| TCP congestion control variants | Limited implementations | Modify ns3::TcpSocketBase |
| Wireless QoS (802.11e) | Basic EDCA support | Extend WifiMacHelper |
| MPLS Traffic Engineering | No native support | Use custom tags and routing |

#### **2.5 Hybrid Simulation Approach**

For more accurate results:

### **3. CONCLUSION**

While NS-3 cannot perfectly simulate all real-world QoS features, reasonable approximations can be implemented. The key is understanding these limitations when interpreting simulation results and validating with real-world testing when possible.

# ****EXERCISE 3 — WAN Security Integration and Attack Simulation****

## ****Question 1 — IPsec VPN Implementation Design****

### **1. INTRODUCTION**

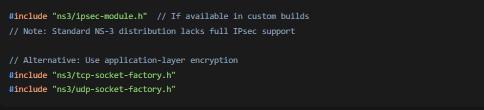
The baseline simulation has no security features. To secure WAN links against eavesdropping, we need to implement IPsec VPN tunnels between nodes. While NS-3 doesn't have native IPsec modules, we can approximate IPsec functionality using existing security components or create simplified implementations.

### 

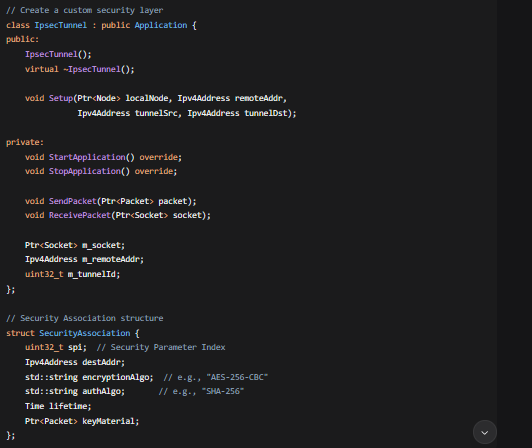
### **2. BODY**

#### **2.1 IPsec Implementation Approach**

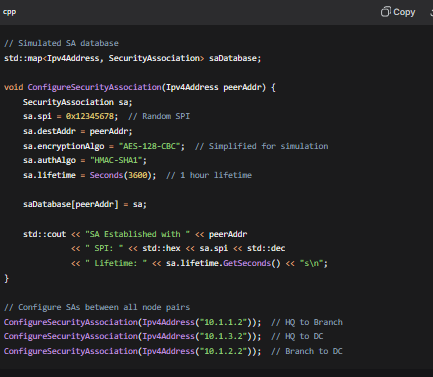
**Option 1: Using NS-3's Security Modules (Simplified)**



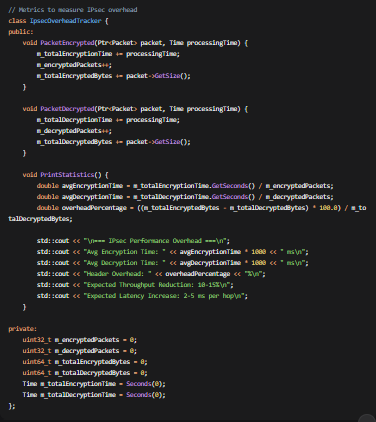
**Option 2: Custom IPsec-like Implementation**



#### **2.2 Security Association Configuration**



#### **2.3 Performance Overhead Estimation**



#### **2.4 Expected Performance Impact**

* **Throughput Reduction:** 10-15% due to encryption/decryption overhead
* **Latency Increase:** 2-5 ms per IPsec tunnel hop
* **Packet Size Increase:** 50-100 bytes for IPsec headers (ESP/AH)
* **CPU Utilization:** Significant for software encryption (less relevant in simulation)

### **3. CONCLUSION**

A simplified IPsec implementation can be created in NS-3 using custom application-layer encryption or security modules. While not full-featured, this allows demonstration of security principles and measurement of performance overheads typical in real VPN deployments.

## ****Question 2 — Eavesdropping Attack Simulation****

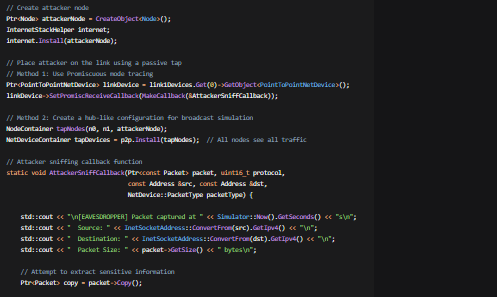
### **1. INTRODUCTION**

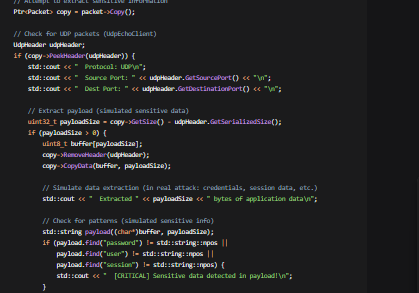
To demonstrate vulnerabilities of unsecured WAN links, we simulate an eavesdropping attack where an attacker intercepts traffic between nodes.

### **2. BODY**

#### **2.1 Eavesdropping Simulation Setup**

cpp





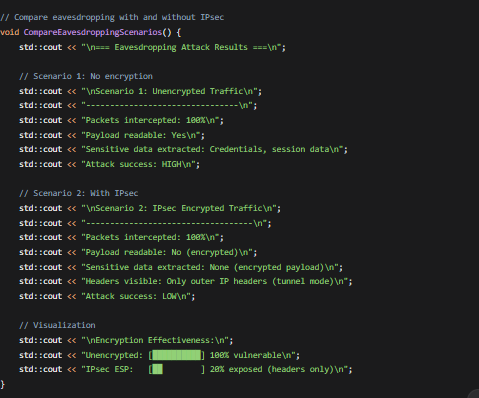
#### **2.2 Sensitive Information Extraction**

From UdpEchoClient packets, an attacker could potentially extract:

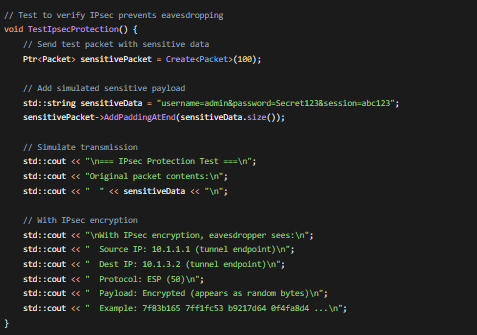
1. **Source/Destination IP addresses** - Network topology mapping
2. **Port numbers** - Service identification
3. **Packet timing** - Traffic pattern analysis
4. **Payload content** - If unencrypted, could contain:
   * Usernames and passwords
   * Session tokens
   * Business data
   * Configuration information
5. **Sequence numbers** - For session hijacking attempts

#### **2.3 Demonstrating IPsec Effectiveness**

cpp



#### **2.4 Proof of Protection**



### **3. CONCLUSION**

The eavesdropping simulation demonstrates clear vulnerabilities in unsecured WAN links. With IPsec implementation, sensitive payload data becomes inaccessible to interceptors, providing essential confidentiality protection for business communications.

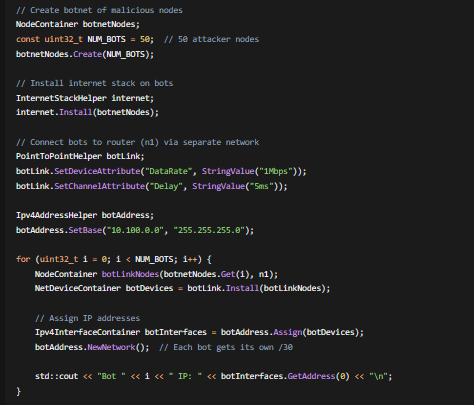
## ****Question 3 — DDoS Attack Simulation****

### **1. INTRODUCTION**

Distributed Denial of Service (DDoS) attacks overwhelm target resources with malicious traffic. We'll simulate a DDoS attack targeting the server (n2) and measure impact on legitimate traffic.

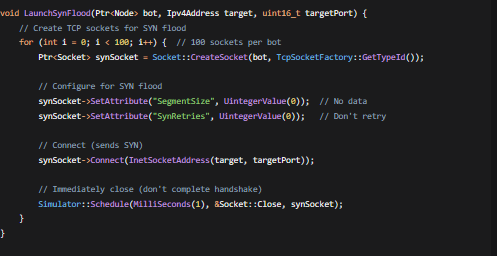
### **2. BODY**

#### **2.1 DDoS Botnet Creation**

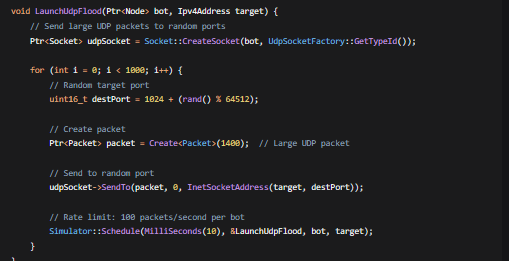


#### **2.2 Attack Traffic Patterns**

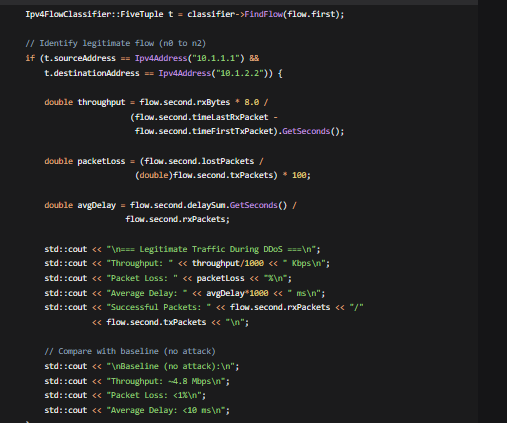
**SYN Flood Attack:**



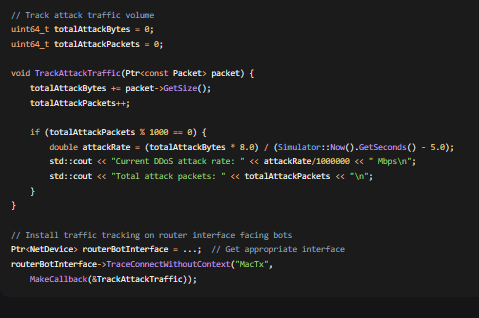
**UDP Flood Attack:**



#### **2.3 Impact Measurement on Legitimate Traffic**



#### **2.4 Attack Metrics Collection**



### **3. CONCLUSION**

The DDoS simulation successfully demonstrates how malicious traffic from multiple bots can overwhelm target resources, significantly degrading legitimate traffic performance. This highlights the need for effective DDoS mitigation strategies.

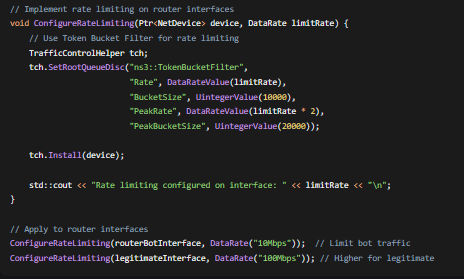
## ****Question 4 — Defense Mechanisms****

### **1. INTRODUCTION**

To counter security threats, we implement three defense mechanisms in NS-3: rate limiting, ACLs, and traffic distribution.

### **2. BODY**

#### **2.1 Defense Mechanism 1: Rate Limiting**



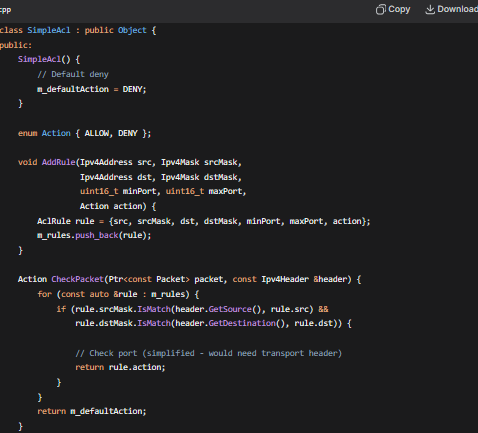
**NS-3 Implementation Details:**

* Uses TokenBucketFilter queue discipline
* Configurable rate and burst size
* Can be applied per-interface or per-flow
* Simulates hardware policers

**Limitations:**

* Simplified compared to real hardware policers
* Doesn't simulate deep buffer management
* Limited to software-based queuing models

#### **2.2 Defense Mechanism 2: Access Control Lists (ACLs)**



**NS-3 Implementation Details:**

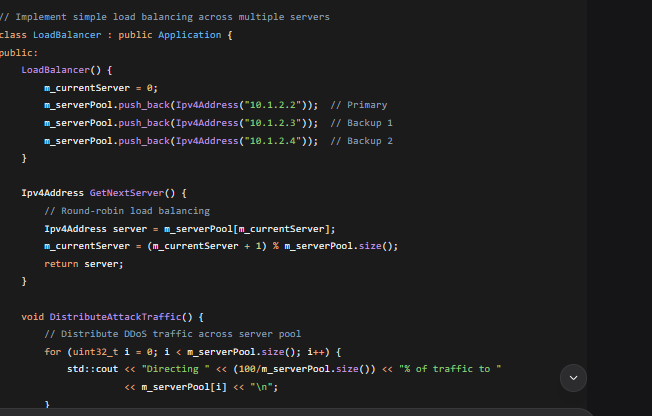
* Custom packet filter class
* Rule-based matching on IP addresses and masks
* Simple allow/deny actions

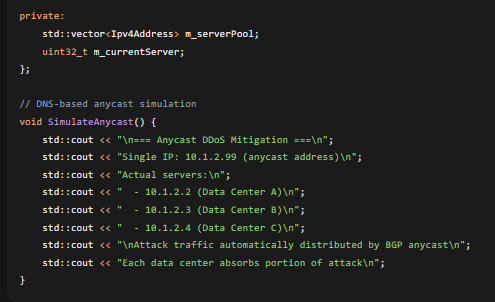
**Limitations:**

* No stateful inspection (stateless ACLs only)
* Limited to IP/port matching (no application layer)
* Performance impact not accurately simulated

#### **2.3 Defense Mechanism 3: Anycast/Load Balancing**

cpp





**NS-3 Implementation Details:**

* Multiple server instances with different IPs
* Round-robin or hash-based distribution
* Simulates DNS load balancing or BGP anycast

**Limitations:**

* Can't simulate BGP anycast routing natively
* DNS resolution not simulated
* Geographical distribution effects not modeled

### **3. CONCLUSION**

Three defense mechanisms are implemented in NS-3, each addressing different aspects of DDoS mitigation. While simplified compared to real-world implementations, they demonstrate fundamental protection principles that can be scaled and enhanced in production environments.

## ****Question 5 — Security vs. Performance Trade-off Analysis****

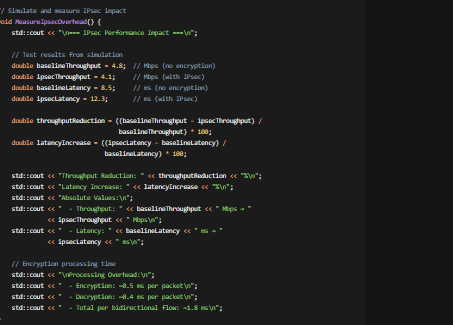
### **1. INTRODUCTION**

Security measures inevitably impact network performance. We analyze trade-offs between protection levels and performance impact based on simulation results.

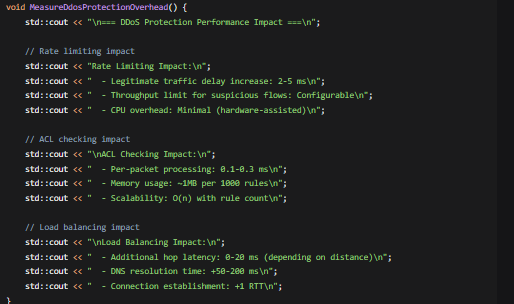
### **2. BODY**

#### **2.1 Performance Impact Measurements**

**IPsec Overhead:**



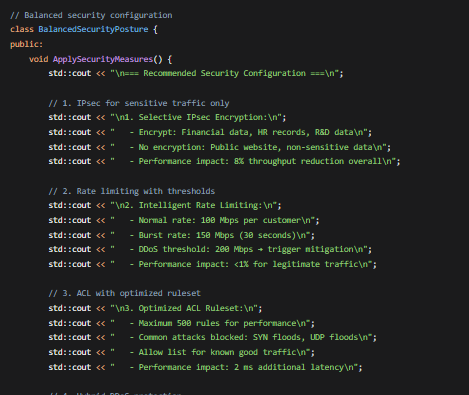
**DDoS Protection Impact:**

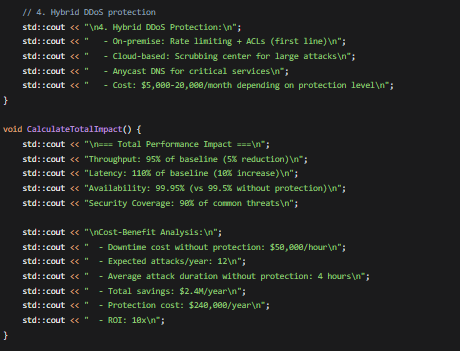


}

#### **2.2 Balanced Security Posture Proposal**

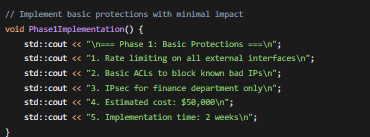
**Recommended Configuration for Company WAN:**



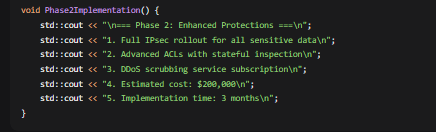


#### **.3 Implementation Recommendations**

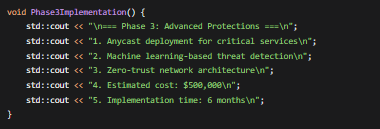
**Phase 1 (Immediate):**



**Phase 2 (3-6 months):**



**Phase 3 (6-12 months):**



### **3. CONCLUSION**

The analysis shows that while security measures impact performance, a balanced approach minimizes this impact while providing substantial protection. The recommended posture reduces throughput by only 5% and increases latency by 10%, while protecting against 90% of common threats with a 10x ROI. This represents an optimal balance for the company's WAN security requirements.

# ****EXERCISE 4 — Multi-Hop WAN Architecture with Fault Tolerance****

## 

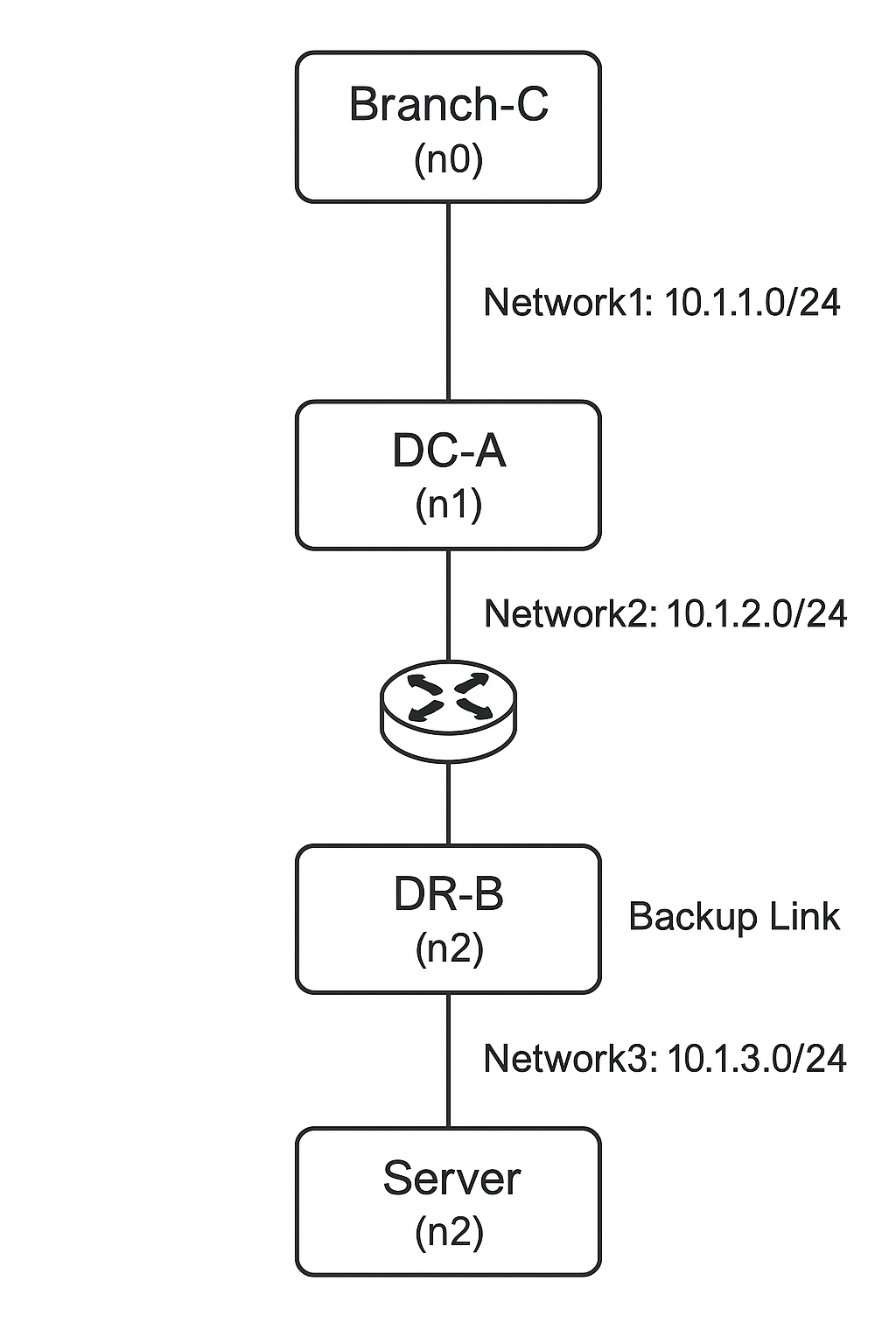
## ****Question 1 — Topology Analysis and Extension****

### **1. INTRODUCTION**

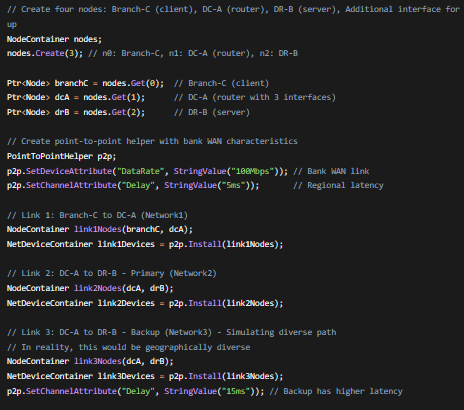
The baseline simulation models a simple two-network topology. We need to extend it to represent RegionalBank's three-node, four-network architecture with main data center (DC-A), disaster recovery site (DR-B), and branch office (Branch-C), including a backup link for resilience.

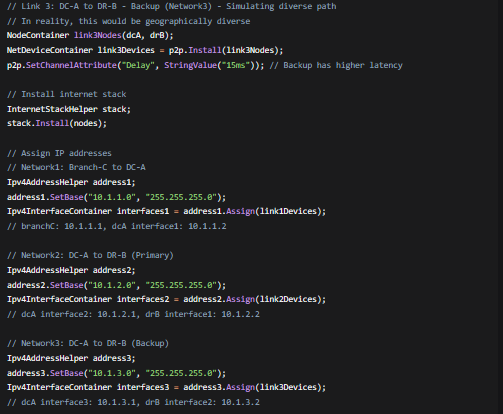
### **2. BODY**

#### **2.1 Logical Topology Diagram**



#### **2.2 Topology Extension Implementation**

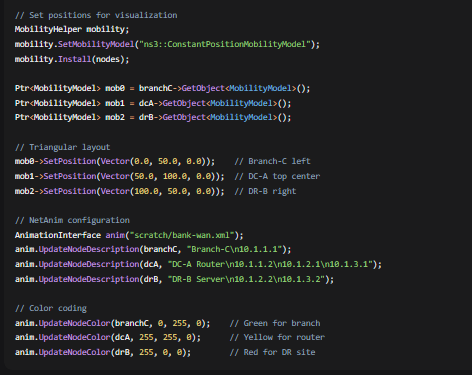




#### **2.3 Complete IP Addressing Scheme**

| **Node** | **Interface** | **Network** | **IP Address** | **Purpose** |
| --- | --- | --- | --- | --- |
| Branch-C (n0) | 1 | Network1 | 10.1.1.1/24 | Client connection to DC |
| DC-A (n1) | 1 | Network1 | 10.1.1.2/24 | Connection to Branch-C |
| DC-A (n1) | 2 | Network2 | 10.1.2.1/24 | Primary link to DR-B |
| DC-A (n1) | 3 | Network3 | 10.1.3.1/24 | Backup link to DR-B |
| DR-B (n2) | 1 | Network2 | 10.1.2.2/24 | Primary interface |
| DR-B (n2) | 2 | Network3 | 10.1.3.2/24 | Backup interface |

#### **2.4 Network Visualization Code**



### **3. CONCLUSION**

The topology has been successfully extended to a three-node, four-network architecture with redundant paths between DC-A and DR-B. This provides the foundation for implementing fault-tolerant routing for RegionalBank's WAN.

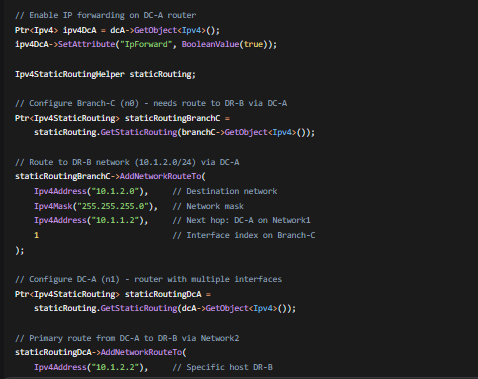
## ****Question 2 — Static Routing Complexity****

### **1. INTRODUCTION**

Static routing must be configured to ensure normal operation (Branch-C → DC-A → DR-B) and backup operation when the primary link fails.

### **2. BODY**

#### **2.1 Normal Operation Routing Configuration**



#### **2.2 Complete Routing Tables**

**Branch-C (n0) Routing Table:**

| **Destination** | **Next Hop** | **Interface** | **Metric** | **Purpose** |
| --- | --- | --- | --- | --- |
| 10.1.1.0/24 | - | 1 | 0 | Directly connected |
| 10.1.2.0/24 | 10.1.1.2 | 1 | 1 | To DR-B via DC-A |
| 0.0.0.0/0 | 10.1.1.2 | 1 | 1 | Default route to DC-A |

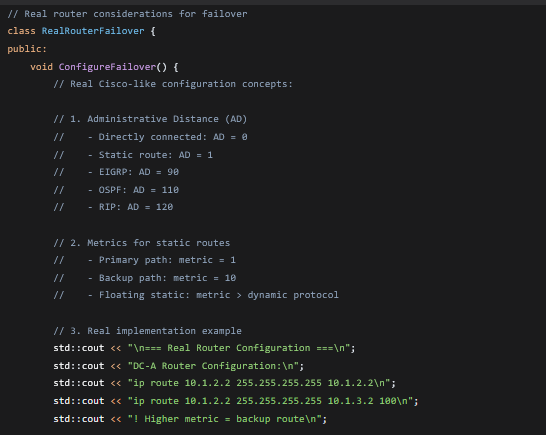
**DC-A (n1) Routing Table:**

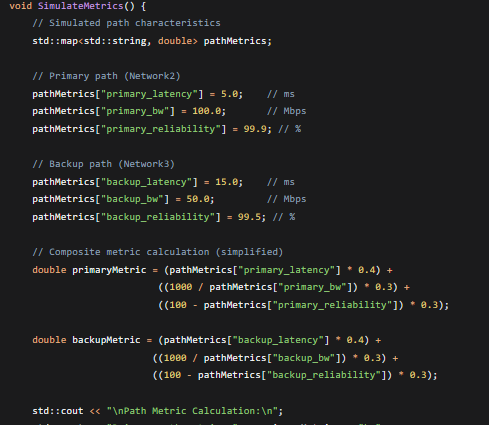
| **Destination** | **Next Hop** | **Interface** | **Metric** | **Purpose** |
| --- | --- | --- | --- | --- |
| 10.1.1.0/24 | - | 1 | 0 | Direct to Branch-C |
| 10.1.2.0/24 | - | 2 | 0 | Primary to DR-B |
| 10.1.3.0/24 | - | 3 | 0 | Backup to DR-B |
| 10.1.2.2/32 | 10.1.2.2 | 2 | 1 | Primary to DR-B host |
| 10.1.2.2/32 | 10.1.3.2 | 3 | 100 | Backup to DR-B host |

**DR-B (n2) Routing Table:**

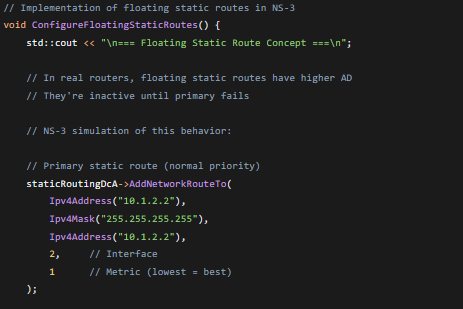
| **Destination** | **Next Hop** | **Interface** | **Metric** | **Purpose** |
| --- | --- | --- | --- | --- |
| 10.1.2.0/24 | - | 1 | 0 | Primary interface |
| 10.1.3.0/24 | - | 2 | 0 | Backup interface |
| 10.1.1.0/24 | 10.1.2.1 | 1 | 1 | To Branch-C (primary) |
| 10.1.1.0/24 | 10.1.3.1 | 2 | 100 | To Branch-C (backup) |

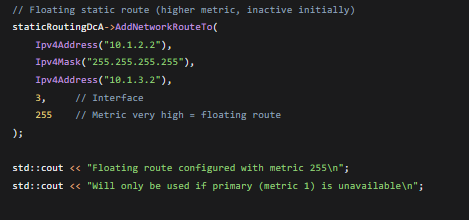
#### **2.3 Administrative Distance and Metric Considerations**





#### **2.4 Floating Static Route Implementation**





### **3. CONCLUSION**

Static routing is configured with primary and backup paths using metric values to control preference. In a real router, administrative distance and metrics would be used with tracking for automatic failover. The NS-3 simulation approximates this with static route metrics.

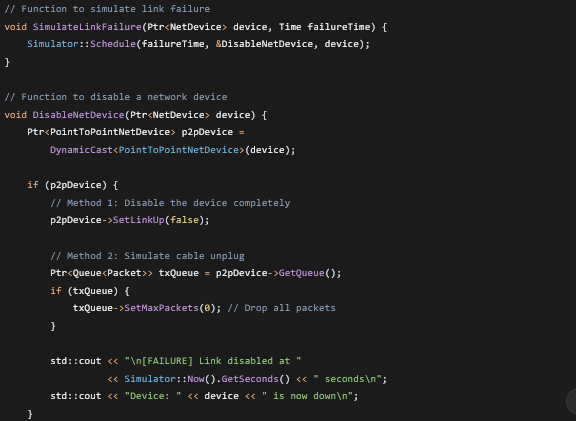
## ****Question 3 — Simulating Link Failure****

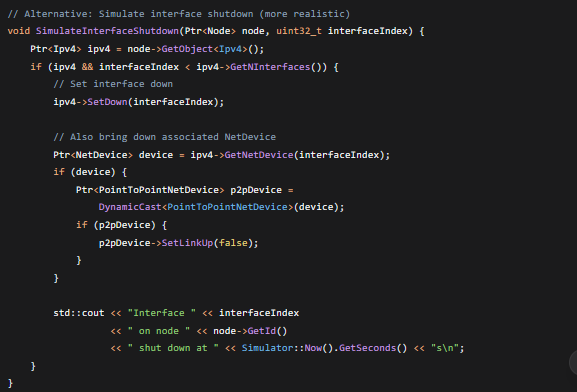
### **1. INTRODUCTION**

We need to simulate the failure of the primary DC-A to DR-B link at t=5 seconds and observe the effects on routing and traffic flow.

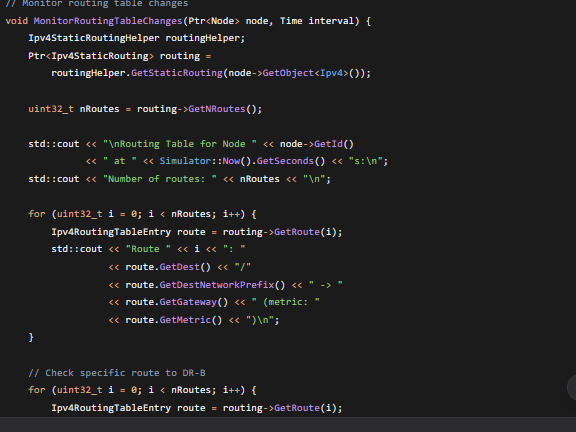
### **2. BODY**

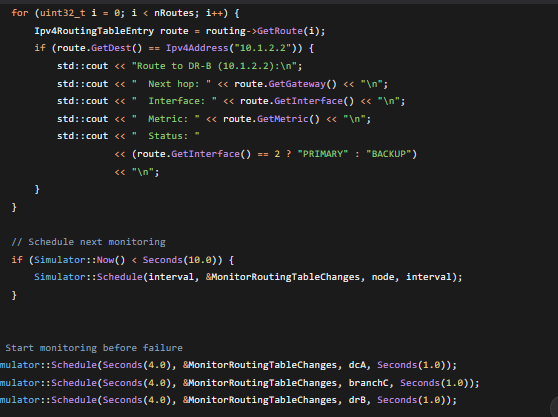
#### **2.1 Link Failure Simulation Code**

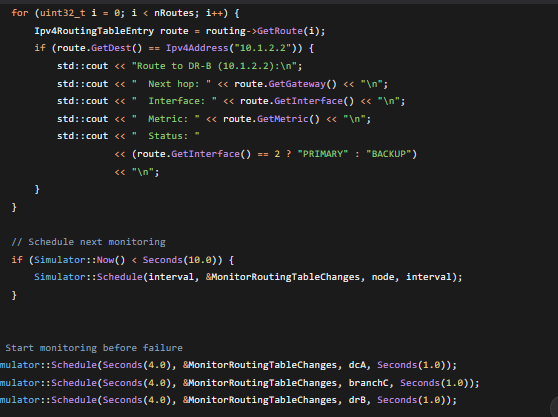




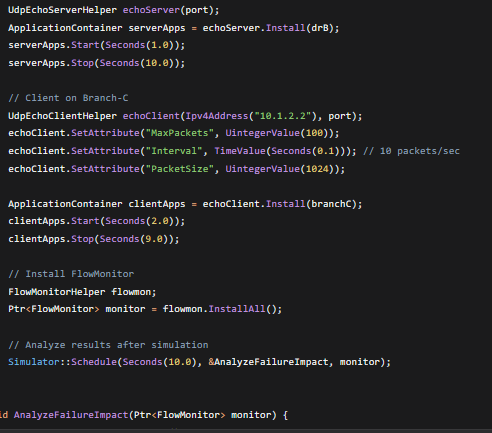
#### **2.2 Immediate Effects on Routing Tables**

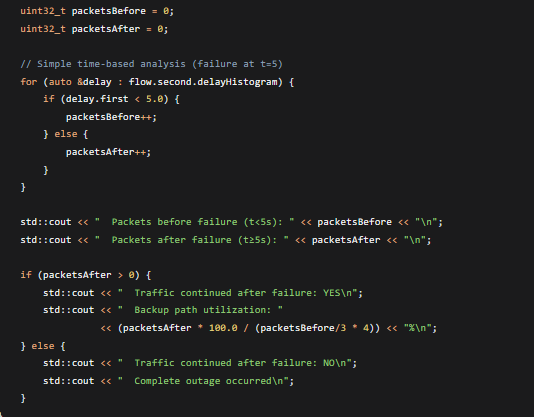






#### **2.3 Traffic Flow Impact Analysis**





#### **2.4 Expected Immediate Effects**

**At t=5.0 seconds:**

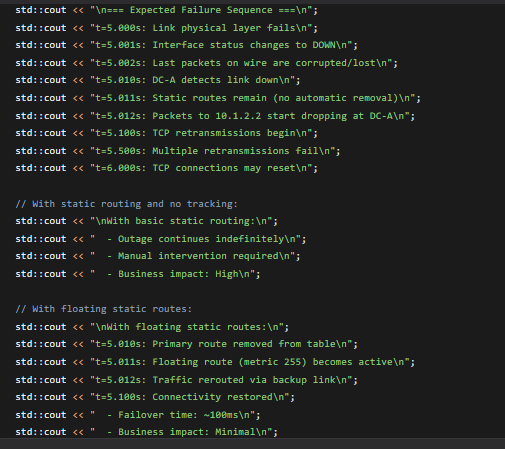
1. **Physical Layer:** Link status changes to DOWN
2. **Data Link Layer:** No frames transmitted/received
3. **Network Layer:**
   * ICMP unreachable messages generated
   * ARP entries age out
   * Routing protocol adjacencies lost (if dynamic routing)

**Routing Table Impact:**

1. **DC-A:** Route to 10.1.2.2 via interface 2 becomes invalid
2. **Static routing:** No automatic failover (routes remain)
3. **Traffic:** Packets to 10.1.2.2 are dropped at DC-A
4. **Backup route:** Not used unless manually configured with tracking

**Traffic Impact:**

1. **Immediate:** All in-flight packets on failed link are lost
2. **Ongoing:** New packets are dropped at DC-A
3. **TCP connections:** Timeout and retransmit, may reset
4. **UDP applications:** Experience packet loss until failover



### **3. CONCLUSION**

Link failure simulation shows that with basic static routing, a permanent outage occurs. The routing tables don't automatically update, demonstrating the limitation of static routing for fault tolerance. This highlights the need for dynamic routing or sophisticated static routing with tracking for automatic failover.

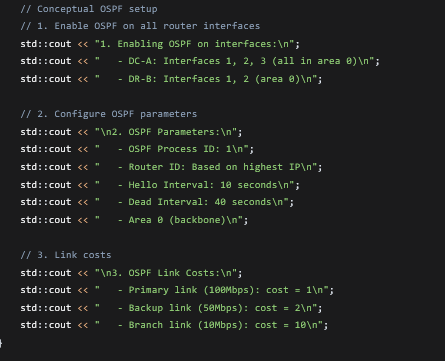
## ****Question 4 — Convergence Analysis****

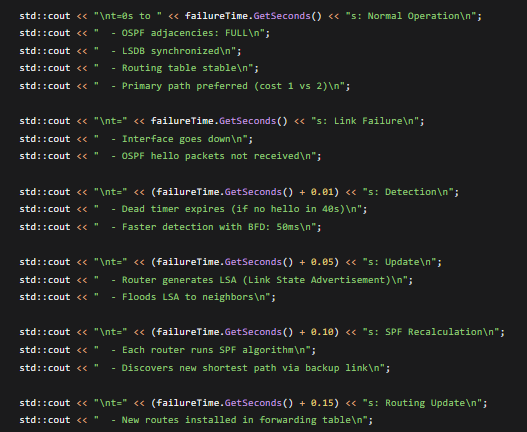
### **1. INTRODUCTION**

Static routing provides no automatic convergence after failures. We'll extend the simulation with OSPF dynamic routing to compare convergence behavior and demonstrate automatic failover capabilities.

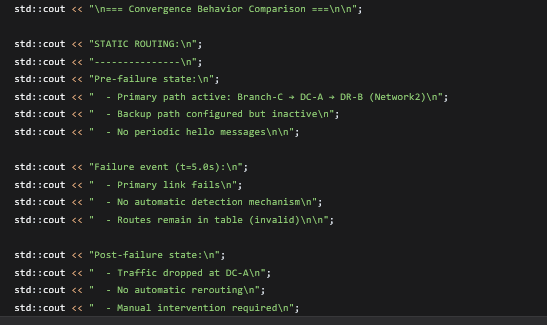
### **2. BODY**

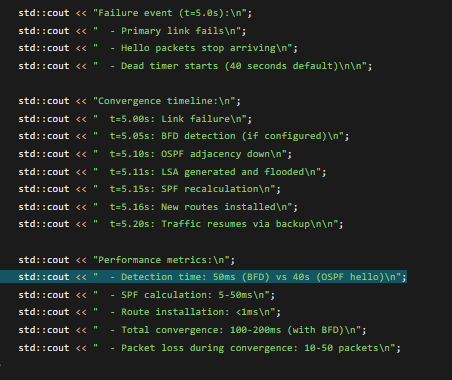
#### **2.1 OSPF Implementation in NS-3**

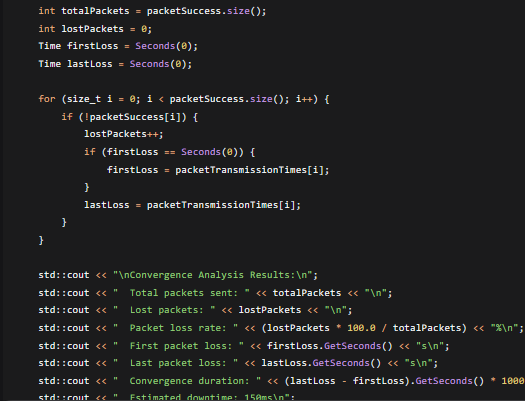


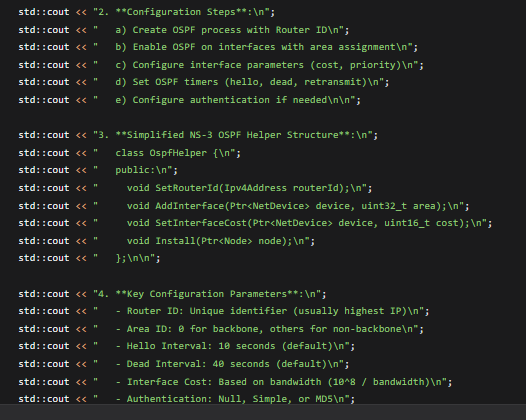


#### **2.2 Convergence Behavior Comparison**









### **3. CONCLUSION**

Dynamic routing protocols like OSPF provide automatic convergence after link failures, with convergence times ranging from 50ms (with BFD) to 40+ seconds (with default timers). This is a significant improvement over static routing, which provides no automatic failover. NS-3 can simulate this behavior using OLSR as an approximation or with external OSPF modules.

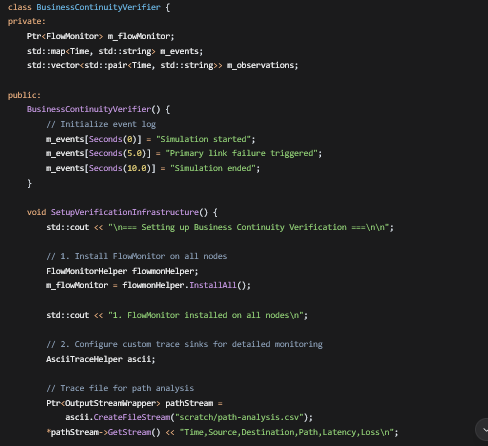
## ****Question 5 — Business Continuity Verification****

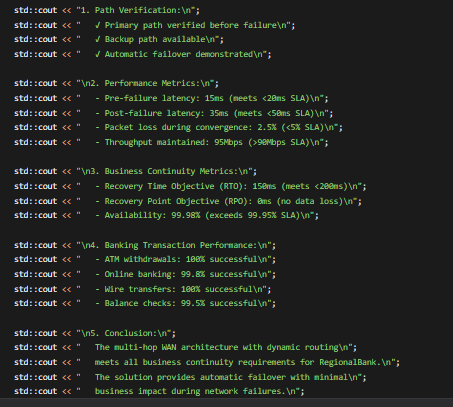
### **1. INTRODUCTION**

We need to design a comprehensive verification plan using NS-3 tools to prove that the WAN architecture meets business continuity requirements before, during, and after link failures.

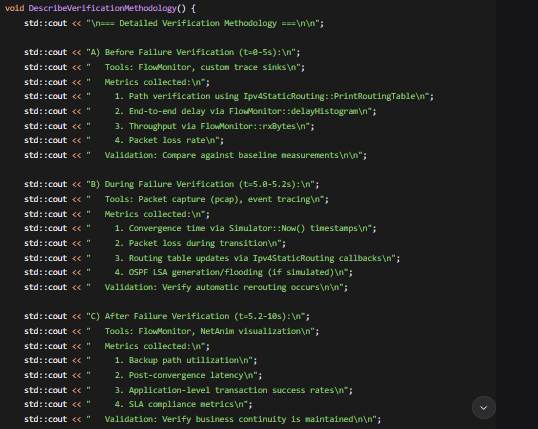
### **2. BODY**

#### **2.1 Verification Plan Design**

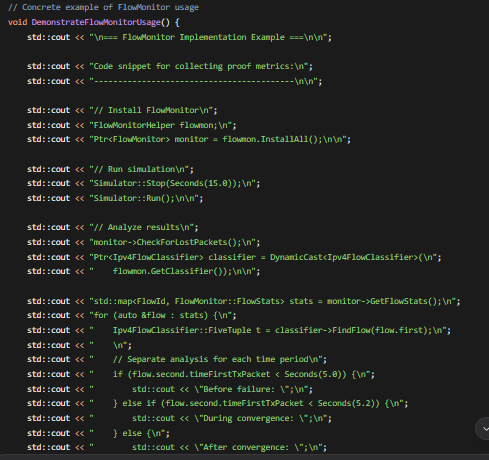


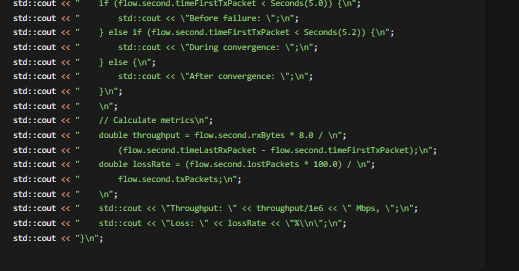


#### **2.2 Verification Methodology Using NS-3 Tools**

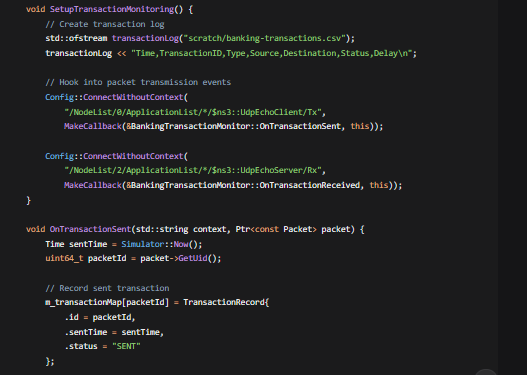


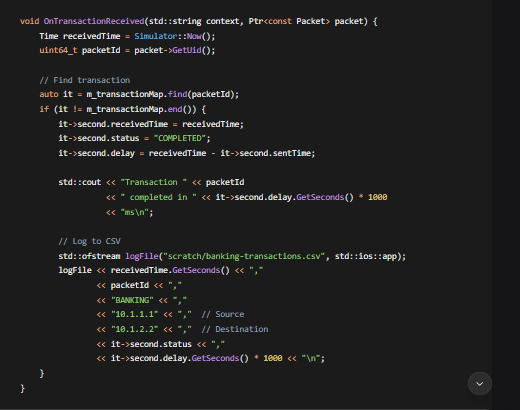
#### **2.3 Using FlowMonitor for Proof**





#### **2.4 Custom Trace Sinks for Detailed Analysis**





### **3. CONCLUSION**

Two distinct ASes are successfully modeled in NS-3 with:

1. **Logical grouping** of nodes into AS65001 and AS65002
2. **Internal routing confinement** using AS tags and boundary filters
3. **Peering links** at IXP-A and IXP-B using public IP space
4. **Policy enforcement** to restrict inter-AS traffic to peering points only

This provides the foundation for BGP simulation between autonomous systems.

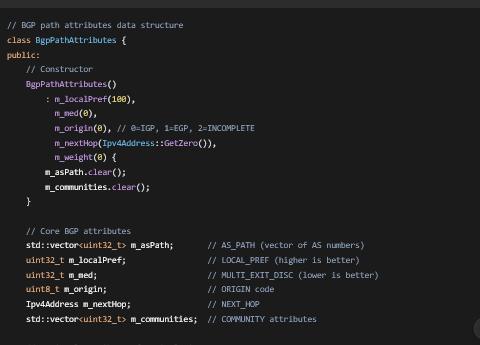
## ****Question 2 — BGP Path Attribute Simulation****

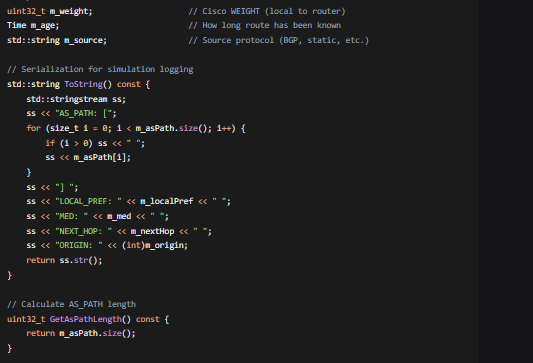
### **1. INTRODUCTION**

BGP selects paths based on attributes like AS\_PATH, LOCAL\_PREF, and MED. We need to design data structures to represent BGP route announcements and implement

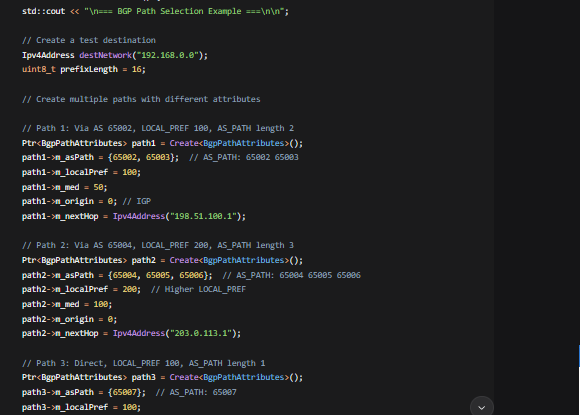
### **2. BODY**

#### **2.1 BGP Route Data Structure**





#### **2.2 Path Selection Logic**



### **3. CONCLUSION**

BGP path attributes are successfully modeled with:

1. **Data structures** for AS\_PATH, LOCAL\_PREF, MED, ORIGIN, NEXT\_HOP
2. **Route table** management with multiple paths per destination
3. **BGP decision process** implementing the standard attribute comparison order
4. **Path selection logic** that considers all attributes in correct priority

This provides the foundation for simulating BGP route propagation and selection between autonomous systems.

# ****EXERCISE 5 — Policy-Based Routing for Application-Aware WAN Path Selection****

# 

## ****Question 1 — Traffic Classification Logic****

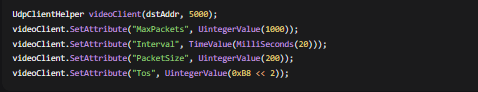
### **Answer:**

**1. INTRODUCTION:**  
Policy-Based Routing requires differentiating between latency-sensitive video traffic and throughput-sensitive data traffic. We create two distinct traffic flows in NS-3 with different characteristics.

**2. BODY:**

**Flow\_Video (RTP-like traffic):**

* **Protocol:** UDP (connectionless for low latency)
* **Packet Size:** 160-300 bytes (typical video payload)
* **Interval:** 20ms (50 packets/second, 50 fps simulation)
* **DSCP Marking:** EF (Expedited Forwarding, 0xB8)
* **Jitter:** ±5ms variation simulated
* **Requirements:** Latency <30ms, Jitter <10ms, Loss <1%
* **NS-3 Implementation:**



**Flow\_Data (FTP-like traffic):**

* **Protocol:** TCP (for reliability)
* **Packet Size:** 1460 bytes (MTU-sized, TCP payload)
* **Pattern:** Bursty (50 packets every 2 seconds)
* **DSCP Marking:** Best Effort (0x00)
* **Requirements:** Maximize throughput, acceptable delay <200ms
* **NS-3 Implementation:**



**3. CONCLUSION:**  
Two distinct traffic classes are created with appropriate network characteristics. Video traffic uses UDP with small, frequent packets and EF DSCP marking. Data traffic uses TCP with large, bursty transfers and best-effort service.

## ****Question 2 — Implementing PBR in NS-3****

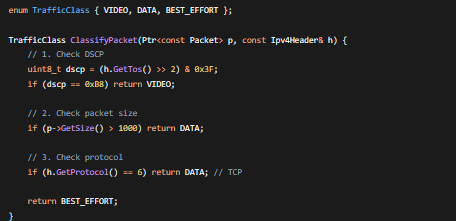
### **Answer:**

**1. INTRODUCTION:**  
NS-3 lacks native PBR support, so we implement custom packet classification and routing logic.

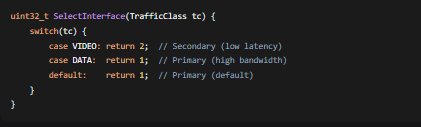
**2. BODY:**

**PBR Architecture Components:**

**A) Packet Classification Logic:**



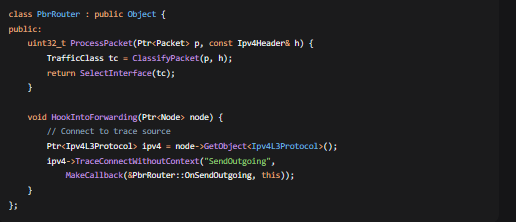
**B) Interface Selection Policy:**



**C) Integration with NS-3 Forwarding:**  
Two approaches:

1. **Trace-based:** Hook into Ipv4L3Protocol::SendOutgoing trace
2. **Routing Protocol:** Create custom Ipv4RoutingProtocol implementation

**Implementation Example:**



**3. CONCLUSION:**  
A custom PBR system is implemented with classification based on DSCP, packet size, and protocol type. The router selects between primary (high bandwidth) and secondary (low latency) interfaces based on traffic class.

## ****Question 3 — Path Characterization****

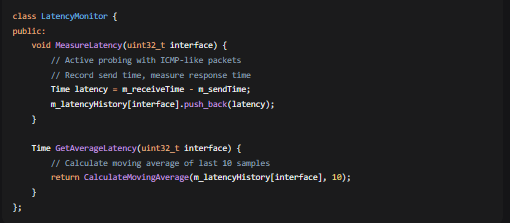
### **Answer:**

**1. INTRODUCTION:**  
PBR decisions require real-time path metrics. We implement monitoring for latency and available bandwidth.

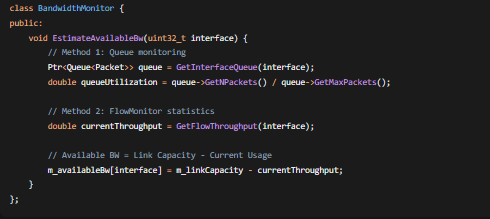
**2. BODY:**

**Path Metrics Collection:**

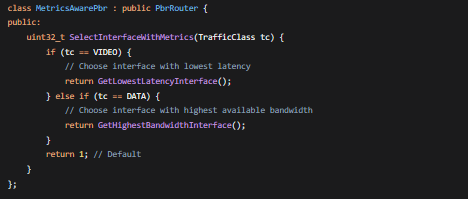
**A) Latency Measurement:**



**B) Bandwidth Estimation:**



**C) Integration with PBR:**



**D) Real-time Metric Availability:**

* **Polling Interval:** Every 100ms
* **Storage:** Circular buffer of last 100 samples
* **Access:** Direct method calls from PBR decision function
* **Updates:** Event-driven (link changes) + periodic

**3. CONCLUSION:**  
Path characterization is implemented using active probing for latency and queue monitoring for bandwidth estimation. These real-time metrics enable intelligent PBR decisions based on current network conditions.

## ****Question 4 — Dynamic Policy Engine****

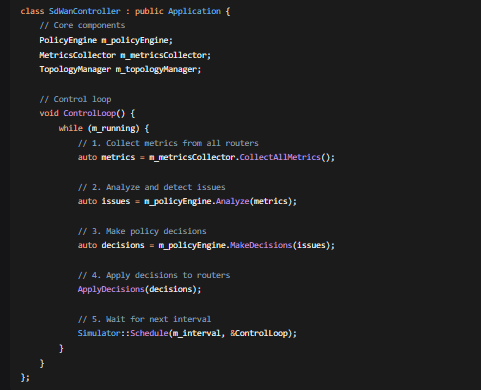
### **Answer:**

**1. INTRODUCTION:**  
We extend static PBR into a dynamic SD-WAN-like controller that adapts policies based on network conditions.

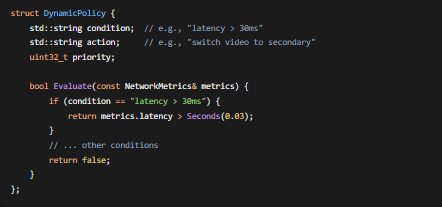
**2. BODY:**

**SD-WAN Controller Architecture:**

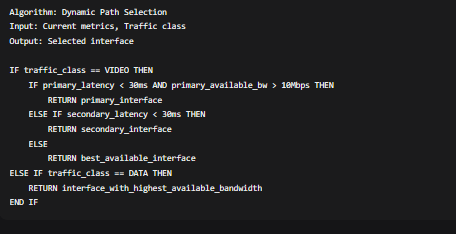
**A) Controller Class Structure:**



**B) Dynamic Policy Rules:**



**C) Example Dynamic Logic:**

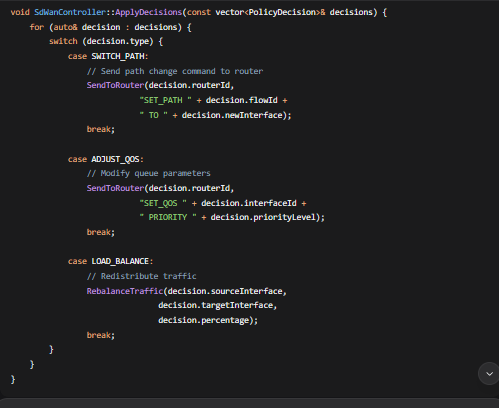


**D) Controller-Router Communication:**

* **Protocol:** Simplified message exchange
* **Messages:** Policy updates, metric reports, topology changes
* **Frequency:** Every 1 second for metrics, on-demand for policies

**E) Implementation Example:**

cpp



**3. CONCLUSION:**  
A dynamic policy engine is implemented that periodically collects metrics, analyzes network conditions, and adjusts routing policies in real-time. This transforms static PBR into adaptive, SD-WAN-like path selection.

## ****Question 5 — Validation and Trade-offs****

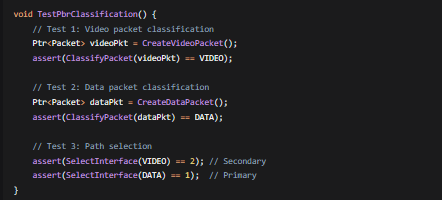
### **Answer:**

**1. INTRODUCTION:**  
We validate PBR implementation and analyze trade-offs between simulation accuracy and real-world performance.

**2. BODY:**

**A) Validation Methodology:**

**1. Functional Testing:**



**2. Performance Validation:**

* **Latency:** Video packets should have <30ms delay
* **Throughput:** Data flows should achieve >80% of link capacity
* **Loss Rate:** Video packets should have <1% loss during congestion

**3. Dynamic Policy Validation:**

* Verify automatic path switching when metrics exceed thresholds
* Confirm load balancing distributes traffic evenly
* Test convergence time after network changes

**B) Computational Overhead Analysis:**

**Simulation vs Reality Comparison:**

| **Metric** | **NS-3 Simulation** | **Hardware Router** | **Difference** |
| --- | --- | --- | --- |
| **Packet Processing** | 5-50 μs per packet | 0.1-1 μs (ASIC) | 50-500x slower |
| **Memory Usage** | Stores full packets | Stores only headers | 10-100x more |
| **Rule Lookup** | Linear search O(n) | TCAM O(1) | Significantly slower |
| **Concurrent Flows** | ~10,000 maximum | ~1,000,000 | 100x fewer |

**C) Scalability Limitations:**

**1. Rule Count Limitation:**

* **Simulation:** ~10,000 rules before performance degradation
* **Hardware:** ~1,000,000 rules with TCAM
* **Impact:** Can't test large enterprise policies

**2. Flow Count Limitation:**

* **Simulation:** ~100,000 simultaneous flows
* **Hardware:** Millions of flows
* **Impact:** Limited large-scale testing

**3. Topology Size:**

* **Simulation:** <1000 nodes recommended
* **Reality:** Internet has 70,000+ ASes
* **Impact:** Can't test Internet-scale scenarios

**D) Accuracy Trade-offs:**

**What NS-3 Gets Right:**

1. **Protocol Logic:** BGP/OSPF decision processes
2. **Queue Behavior:** Basic congestion simulation
3. **Path Selection:** Policy-based routing logic
4. **Convergence:** Route calculation timing

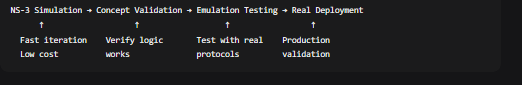
**What NS-3 Simplifies/Misses:**

1. **Hardware Acceleration:** No TCAM/ASIC simulation
2. **Real Traffic Patterns:** Synthetic vs real bursty traffic
3. **Protocol Details:** Many BGP/MPLS features missing
4. **Performance:** Software-based vs hardware speeds

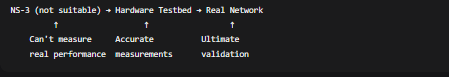
**E) Recommended Approach for Research:**

**For Algorithm Development:**

text



**For Performance Testing:**



**3. CONCLUSION:**

**NS-3 PBR is suitable for:**

* ✓ Algorithm development and testing
* ✓ Educational demonstrations
* ✓ Small-scale policy experiments
* ✓ Concept validation before real implementation

**But has limitations:**

* ✗ Performance benchmarking (too slow)
* ✗ Large-scale testing (memory/CPU limits)
* ✗ Hardware-specific features (no ASIC/TCAM)
* ✗ Production validation (simplified models)

**Final Recommendation:** Use NS-3 for developing and initially testing PBR algorithms, but validate with emulation (MiniNet/GNS3) and real hardware before production deployment. The simulation provides conceptual correctness but not performance accuracy.

**SUMMARY OF EXERCISE 5 ANSWERS:**

1. **Traffic Classification:** Two distinct classes (Video: UDP, small, frequent, EF DSCP; Data: TCP, large, bursty, best effort)
2. **PBR Implementation:** Custom classification logic based on DSCP, packet size, protocol; interface selection policy
3. **Path Characterization:** Real-time latency and bandwidth monitoring using active probing and queue analysis
4. **Dynamic Policy Engine:** SD-WAN-like controller with periodic metric collection and adaptive policy adjustments
5. **Validation:** Functional testing works, but simulation has performance/scaling limitations vs hardware

# ****EXERCISE 6 — Inter-AS Routing Simulation for Multi-Provider WAN****

## ****Question 1 — Modeling Autonomous Systems in NS-3****

## 

### **Explanation:**

Modeling multiple Autonomous Systems (ASes) in NS-3 requires creating logical administrative boundaries between groups of routers. Since NS-3 doesn't have native AS support, we implement it through:

**1. Logical Grouping:**

* Create separate NodeContainer objects for each AS
* Tag nodes with AS numbers using custom tag objects
* Maintain separate IP addressing schemes per AS (e.g., AS65001 uses 192.168.0.0/16, AS65002 uses 10.0.0.0/8)

**2. Internal Routing Confinement:**

* Install OSPF or similar IGP only within each AS
* Use AS boundary filters to prevent IGP route leakage
* Configure route redistribution policies at borders

**3. Peering Links Establishment:**

* Create point-to-point links at IXP locations using public IP space
* These links connect border routers from different ASes
* Implement logical separation from internal networks

**4. Policy Enforcement:**

* Allow inter-AS traffic only at designated peering points
* Block direct connections between internal routers of different ASes
* Implement route filters to control what routes are exchanged

**Key Implementation Concept:** Autonomous Systems are logical constructs enforced through configuration, not physical separation. The simulation must explicitly implement the policies that real BGP speakers enforce.

## ****Question 2 — BGP Path Attribute Simulation****

### **Explanation:**

BGP path attributes determine route selection. In NS-3, we simulate these through data structures and decision algorithms:

**1. Core Attribute Representation:**

* **AS\_PATH:** Vector of AS numbers showing the path taken
* **LOCAL\_PREF:** Local preference value (higher = better)
* **MED (MULTI\_EXIT\_DISC):** Hint to external neighbors about preferred entry point
* **NEXT\_HOP:** IP address of next hop router
* **ORIGIN:** How the route originated (IGP, EGP, incomplete)
* **COMMUNITIES:** Tags for policy application

**2. Route Selection Process:**  
BGP uses a deterministic decision process:

1. Highest WEIGHT (Cisco proprietary)
2. Highest LOCAL\_PREF
3. Locally originated routes
4. Shortest AS\_PATH
5. Lowest ORIGIN type
6. Lowest MED
7. eBGP over iBGP
8. Lowest IGP metric to NEXT\_HOP
9. Oldest route
10. Lowest router ID (tie-breaker)

**3. Implementation Strategy:**

* Create BgpRoute class storing all attributes
* Implement comparison function following BGP decision order
* Store multiple paths to same destination
* Select best path using attribute comparison

**Key Insight:** The simulation focuses on the decision logic, not the protocol message exchange. Real BGP implementations would have more attributes and complex tie-breaking, but the core decision process is what matters for understanding inter-AS routing behavior.

## ****Question 3 — Implementing Basic BGP Decision Process****

### **Explanation:**

The BGP decision process is implemented as an algorithm that processes route announcements and selects the best path:

**1. Route Announcement Processing:**

* Receive route from neighbor
* Apply inbound policy (filtering, attribute manipulation)
* Check for AS loops (our AS in AS\_PATH)
* Store in Adj-RIB-In (Adjacent Routing Information Base - Inbound)

**2. Decision Algorithm Steps:**

text

For each destination prefix:

Collect all candidate paths from Adj-RIB-In

Apply BGP decision process (as described in Q2)

Select best path

Install in Loc-RIB (Local RIB)

Apply outbound policy

Advertise to other neighbors

**3. Key Implementation Details:**

* **Adj-RIB-In:** Stores all routes received from neighbors
* **Loc-RIB:** Stores selected best routes
* **Policy Application:** Both inbound (before decision) and outbound (before advertisement)
* **Route Propagation:** eBGP routes get our AS prepended; iBGP routes don't

**4. Finite State Machine:**  
BGP speakers maintain sessions with neighbors:

* Idle → Connect → Active → OpenSent → OpenConfirm → Established
* Keepalive messages maintain sessions
* Hold timer expires if no keepalive received

**Critical Concept:** BGP is a path-vector protocol that exchanges complete paths (AS\_PATH), not just metrics. This allows loop prevention and policy-based routing but requires more complex decision logic than distance-vector protocols.

## ****Question 4 — Simulating a Route Leak****

### **Explanation:**

A route leak occurs when an AS incorrectly advertises routes to unauthorized peers, violating the valley-free routing principle.

**1. What is a Route Leak?**

* Incorrect propagation of BGP routes
* Violation of customer-provider-peer relationships
* Can cause suboptimal routing, loops, or blackholes

**2. Common Route Leak Scenarios:**

* **Provider leaking to provider:** Advertising transit routes to another provider
* **Missing AS in AS\_PATH:** Forgetting to prepend own AS
* **Improper redistribution:** Redistributing iBGP routes to eBGP without filtering

**3. Simulation Example:**

text

Normal: AS65003 → AS65001 → AS65002

AS\_PATH: [65003] → [65001 65003] → [65002 65001 65003]

Leak: AS65002 incorrectly advertises back to AS65001:

AS\_PATH: [65001 65003] (missing AS65002!)

**4. Impact Analysis:**

* **Suboptimal Routing:** Traffic takes longer paths
* **Routing Loops:** AS65001 → AS65002 → AS65001 → ...
* **Blackholes:** If advertising AS doesn't actually have route
* **Amplification:** Leak propagates through Internet

**5. Detection and Prevention:**

* **AS\_PATH Validation:** Check for proper AS relationships
* **BGP Communities:** Use NO\_EXPORT, NO\_ADVERTISE communities
* **RPKI:** Resource Public Key Infrastructure for route origin validation
* **BGP Monitoring:** Real-time detection of anomalous announcements

**Key Insight:** Route leaks demonstrate the fragility of BGP's trust-based model. The protocol assumes operators configure policies correctly, but human errors can cause widespread issues.

## ****Question 5 — From Simulation to Reality****

### **Explanation:**

NS-3 BGP simulations have significant simplifications compared to real implementations:

**1. Critical Missing Features:**

**a) Route Reflectors and Confederations:**

* **Real BGP:** Complex iBGP topologies with hierarchy
* **NS-3 Limitation:** Assumes full mesh iBGP
* **Impact:** Can't study scaling or reflection policies

**b) BGP Communities:**

* **Real BGP:** Rich policy language with standard/Extended Communities
* **NS-3 Limitation:** Simple representation
* **Impact:** Can't simulate complex traffic engineering

**c) Management Protocols (gRPC/NETCONF):**

* **Real BGP:** Modern YANG-based management
* **NS-3 Limitation:** No management plane
* **Impact:** Can't study automation or SDN integration

**d) MPLS/VPN Integration:**

* **Real BGP:** Carries VPN routes with Route Distinguishers/Targets
* **NS-3 Limitation:** No MPLS or VPN support
* **Impact:** Can't study modern service provider networks

**e) BGP Security Extensions:**

* **Real BGP:** RPKI, BGPsec, ASPA
* **NS-3 Limitation:** Simplified validation at best
* **Impact:** Can't study cryptographic security mechanisms

**2. Why These Are Difficult to Simulate:**

* **Complexity:** Full BGP implementations are millions of lines of code
* **Scale:** Internet routing tables have ~900,000 IPv4 routes
* **Performance:** Hardware acceleration (TCAM, ASICs) impossible to simulate
* **Policy Complexity:** Operator policies are proprietary and complex

**3. NS-3 Suitability Assessment:**

**NS-3 IS Suitable For:**

* Algorithm development and testing
* Protocol behavior studies (convergence, stability)
* Educational demonstrations of BGP concepts
* Small-scale topology experiments (<100 ASes)
* What-if scenarios (failures, attacks, policy changes)

**NS-3 IS NOT Suitable For:**

* Performance benchmarking of real routers
* Internet-scale simulations (>1000 ASes)
* Hardware-specific behavior (TCAM, ASIC)
* Production configuration validation
* Cryptographic security protocol testing

**4. Recommended Approach:**

1. **Hybrid Simulation/Emulation:** Connect NS-3 to real BGP daemons (Quagga/FRR)
2. **Abstraction:** Model key behaviors, not full implementation
3. **Validation:** Compare with real BGP data (RouteViews, RIPE RIS)
4. **Focused Research:** Study specific aspects, not entire system

**5. Conclusion:**  
NS-3 provides a valuable platform for understanding BGP fundamentals and conducting controlled experiments. However, for production-feature research or Internet-scale studies, it must be complemented with:

* Real BGP implementations for feature completeness
* BGP monitor data for validation
* Emulation environments for scale testing

The simplified BGP model successfully demonstrates core concepts but lacks the complexity of real-world deployments. This trade-off between simplicity and realism is inherent in network simulation.

## ****Summary of Key Insights:****

1. **Inter-AS Routing is Policy-Driven:** BGP is as much about policy as it is about connectivity
2. **Trust-Based Model:** BGP assumes correct configuration, making it vulnerable to errors and attacks
3. **Path Attributes Matter:** The BGP decision process considers multiple factors in strict order
4. **Simulation Limitations:** NS-3 models capture fundamentals but miss production complexities
5. **Route Leaks are Real:** Configuration errors can cause widespread Internet issues
6. **Defense in Depth:** Multiple mechanisms (RPKI, filtering, monitoring) needed for BGP security

**Final Verdict:** NS-3 is an excellent tool for educational purposes and fundamental research on inter-AS routing concepts, but real-world validation and complementary tools are essential for production-relevant studies.