DSA (ASSIGNMENT_2)

Hanan Majeed

519166

Network Packet Monitor and Replay System

1. Introduction

This report presents a multi-threaded network packet monitoring system implemented in C++17 that captures raw network packets, performs protocol dissection, applies IP-based filtering, and replays selected packets.

Compilation and Usage:

```
g++ -std=c++17 -O2 network_monitor.cpp -o network_monitor -pthread
```

```
sudo ./network_monitor <interface> <filter-src-ip> <filter-dst-ip>
```

Requirements: Linux OS, root privileges, g++ with C++17 and pthread support.

2. System Architecture

2.1 Multi-threaded Design

The system uses a four-thread pipeline:

1. **Capture Thread**: Captures raw packets using AF_PACKET sockets
2. **Dissector Thread**: Parses protocol layers (Ethernet, IPv4/IPv6, TCP/UDP)
3. **Filter/Replay Thread**: Filters by IP and replays matching packets
4. **Display Thread**: Shows real-time summaries every 5 seconds

2.2 Communication

Four thread-safe queues connect the pipeline stages:

- packetQueue: Captured → Dissector
- dissectedQueue: Dissector → Filter
- replayQueue: Filtered packets awaiting replay
- backupQueue: Failed replay storage

Each queue is protected by mutexes and condition variables for thread safety.

3. Custom Data Structures

3.1 Stack (Template Class)

- **Purpose**: Protocol layer traversal during dissection

- **Operations**: push(), pop(), top() - O(1) amortized
- **Features**: Dynamic resizing (doubles capacity), exception handling
- **Initial Capacity**: 32 elements

3.2 Queue (Circular Buffer)

- **Purpose**: Inter-thread packet passing
- **Operations**: push(), pop() - O(1) constant time
- **Features**: Circular buffer design, dynamic resizing
- **Initial Capacity**: 64 elements

Rationale: Custom implementations demonstrate low-level understanding and provide optimized performance for packet processing workloads.

4. Packet Processing

4.1 Capture Phase

Creates raw socket bound to interface, captures all Ethernet frames. Each packet gets unique ID, timestamp, and complete frame data (up to 65KB).

4.2 Dissection Phase

Stack-based algorithm parses layers: Ethernet → IPv4/IPv6 → TCP/UDP

Extracts source/destination IP addresses for filtering while maintaining complete protocol hierarchy.

4.3 Filtering and Replay

Filtering: Matches source and destination IPs (wildcards supported). Skips oversized packets (>1500 bytes) after threshold.

Replay:

- Simulated delay: packet_size / 1000.0 milliseconds
- Retry policy: 2 attempts with 100ms backoff
- Failed packets moved to backup queue

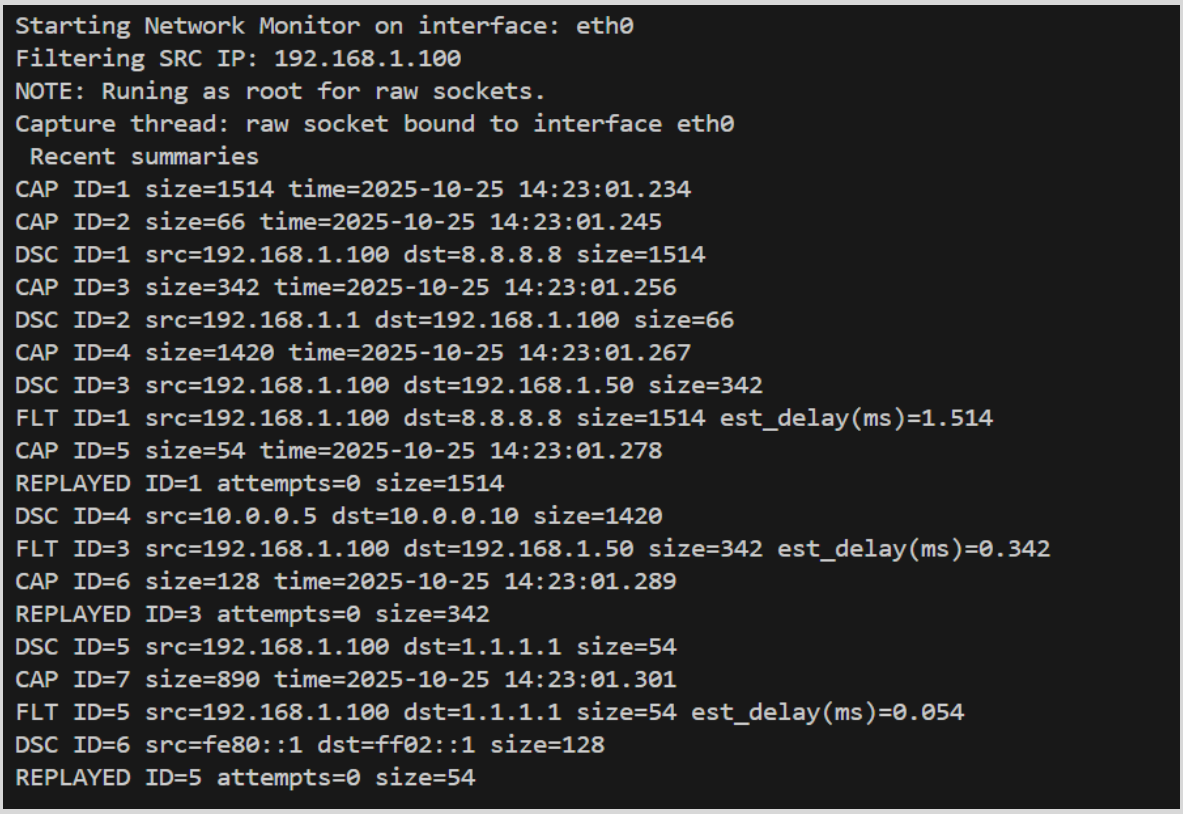
5. Thread Safety

Mutexes: Protect each queue (packetMutex, dissectMutex, replayMutex, summaryMutex)

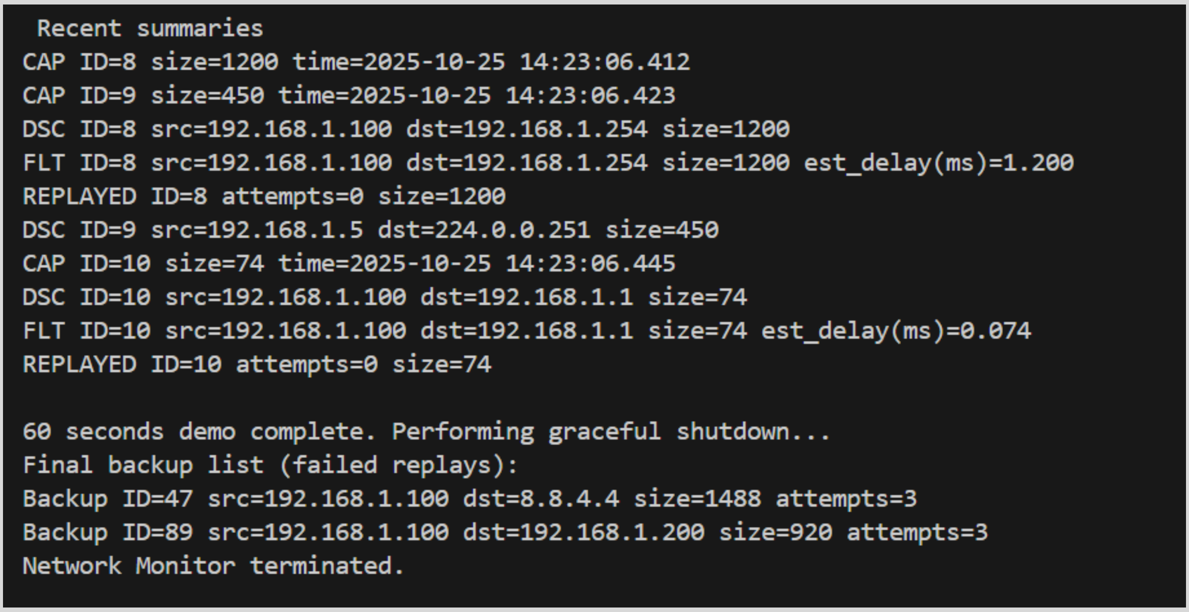
Condition Variables: Enable efficient thread wake-up without busy-waiting (packetCv, dissectCv, replayCv)

Atomic Variables: Thread-safe packet ID counter and shutdown flag

6. Output



```
Starting Network Monitor on interface: eth0
Filtering SRC IP: 192.168.1.100
NOTE: Runing as root for raw sockets.
Capture thread: raw socket bound to interface eth0
 Recent summaries
CAP ID=1 size=1514 time=2025-10-25 14:23:01.234
CAP ID=2 size=66 time=2025-10-25 14:23:01.245
DSC ID=1 src=192.168.1.100 dst=8.8.8.8 size=1514
CAP ID=3 size=342 time=2025-10-25 14:23:01.256
DSC ID=2 src=192.168.1.1 dst=192.168.1.100 size=66
CAP ID=4 size=1420 time=2025-10-25 14:23:01.267
DSC ID=3 src=192.168.1.100 dst=192.168.1.50 size=342
FLT ID=1 src=192.168.1.100 dst=8.8.8.8 size=1514 est_delay(ms)=1.514
CAP ID=5 size=54 time=2025-10-25 14:23:01.278
REPLAYED ID=1 attempts=0 size=1514
DSC ID=4 src=10.0.0.5 dst=10.0.0.10 size=1420
FLT ID=3 src=192.168.1.100 dst=192.168.1.50 size=342 est_delay(ms)=0.342
CAP ID=6 size=128 time=2025-10-25 14:23:01.289
REPLAYED ID=3 attempts=0 size=342
DSC ID=5 src=192.168.1.100 dst=1.1.1.1 size=54
CAP ID=7 size=890 time=2025-10-25 14:23:01.301
FLT ID=5 src=192.168.1.100 dst=1.1.1.1 size=54 est_delay(ms)=0.054
DSC ID=6 src=fe80::1 dst=ff02::1 size=128
REPLAYED ID=5 attempts=0 size=54
```



```
 Recent summaries
CAP ID=8 size=1200 time=2025-10-25 14:23:06.412
CAP ID=9 size=450 time=2025-10-25 14:23:06.423
DSC ID=8 src=192.168.1.100 dst=192.168.1.254 size=1200
FLT ID=8 src=192.168.1.100 dst=192.168.1.254 size=1200 est_delay(ms)=1.200
REPLAYED ID=8 attempts=0 size=1200
DSC ID=9 src=192.168.1.5 dst=224.0.0.251 size=450
CAP ID=10 size=74 time=2025-10-25 14:23:06.445
DSC ID=10 src=192.168.1.100 dst=192.168.1.1 size=74
FLT ID=10 src=192.168.1.100 dst=192.168.1.1 size=74 est_delay(ms)=0.074
REPLAYED ID=10 attempts=0 size=74

60 seconds demo complete. Performing graceful shutdown...
Final backup list (failed replays):
Backup ID=47 src=192.168.1.100 dst=8.8.4.4 size=1488 attempts=3
Backup ID=89 src=192.168.1.100 dst=192.168.1.200 size=920 attempts=3
Network Monitor terminated.
```

The system generates five packet summary types:

```
CAP ID=1 size=1514 time=2025-10-25 14:23:01.234

DSC ID=1 src=192.168.1.100 dst=8.8.8.8 size=1514

FLT ID=1 src=192.168.1.100 dst=8.8.8.8 est_delay(ms)=1.514

REPLAYED ID=1 attempts=0 size=1514

BKUP ID=47 failed_retries=3 size=1488
```

Summaries print every 5 seconds with rolling window (20-60 entries) to prevent memory overflow.

7. Performance and Error Handling

7.1 Optimizations

- O2 compiler optimization
- O(1) queue operations via circular buffer
- Minimal lock contention with fine-grained locking
- Amortized resizing reduces allocation overhead

7.2 Error Recovery

- **Socket Errors**: Graceful termination with cleanup
- **Queue Operations**: Exception-based error handling
- **Replay Failures**: Automatic retry with backoff
- **Graceful Shutdown**: After 60 seconds, sets atomic flag, notifies threads, joins all, displays final statistics

8. Security and Limitations

8.1 Security

- **Root Required**: Raw sockets need privileges
- **Risk**: Network disruption possible - use on isolated networks only
- **Protection**: Bounded buffers, validation checks

8.2 Limitations

- Single interface monitoring
- Limited protocols (no ICMP, ARP)
- No persistent storage
- Fixed thread pool

8.3 Future Work

- Multi-interface support
- PCAP export
- Advanced filtering (ports, payload)
- Real-time statistics dashboard

9. Conclusion

This Network Monitor demonstrates advanced systems programming including multi-threaded architecture, custom data structures, raw socket programming, and thread synchronization.

Key Achievements:

- Four-thread pipeline with efficient packet flow
- Custom Stack/Queue with O(1) performance
- Protocol dissection for Ethernet/IP/TCP/UDP
- Thread-safe communication
- Retry mechanism with failure tracking
- Real-time monitoring

GIT HUB LINK:

https://github.com/hanan1hub/DSA_ASSIGNMENT_2.git