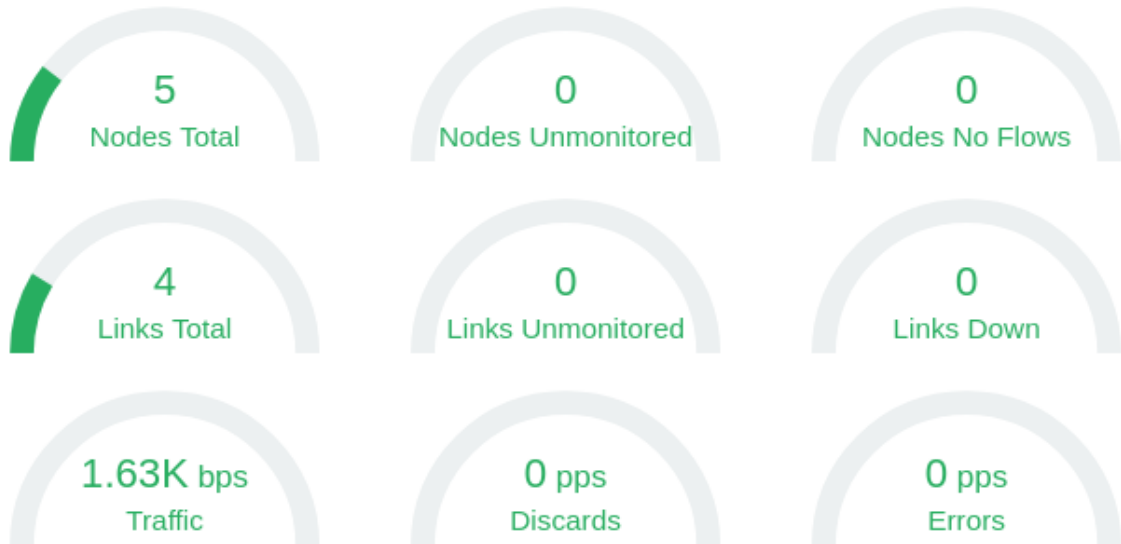
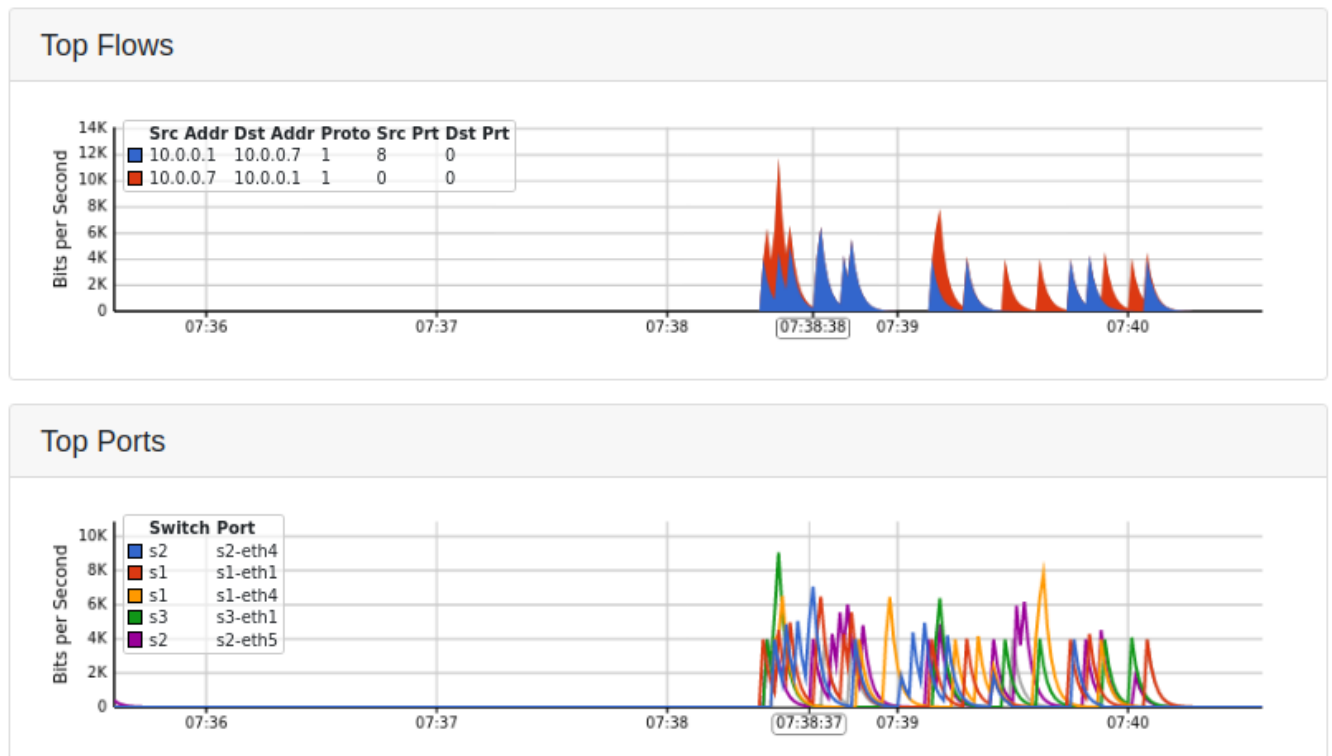


LAMPIRAN A HASIL SIMULASI PERCOBAAN

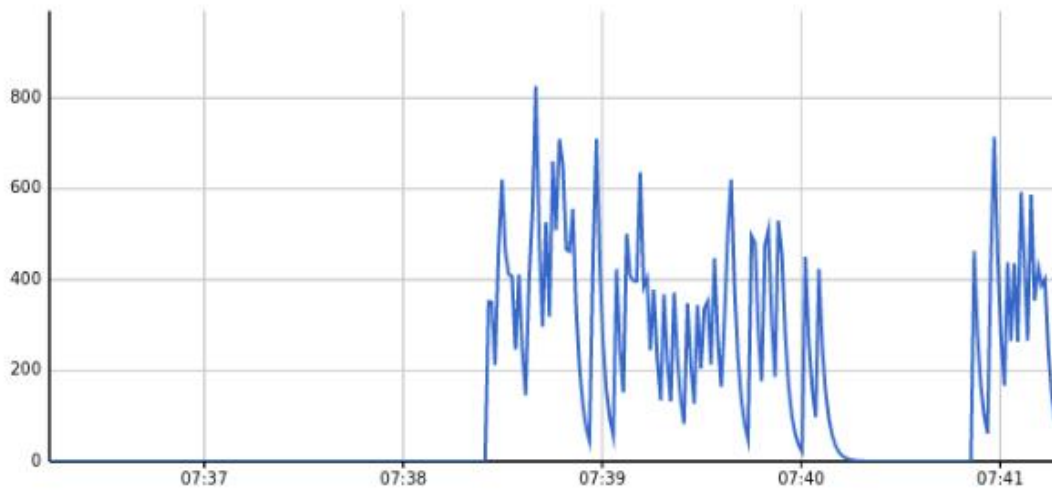
A-1 Lalu Lintas Normal



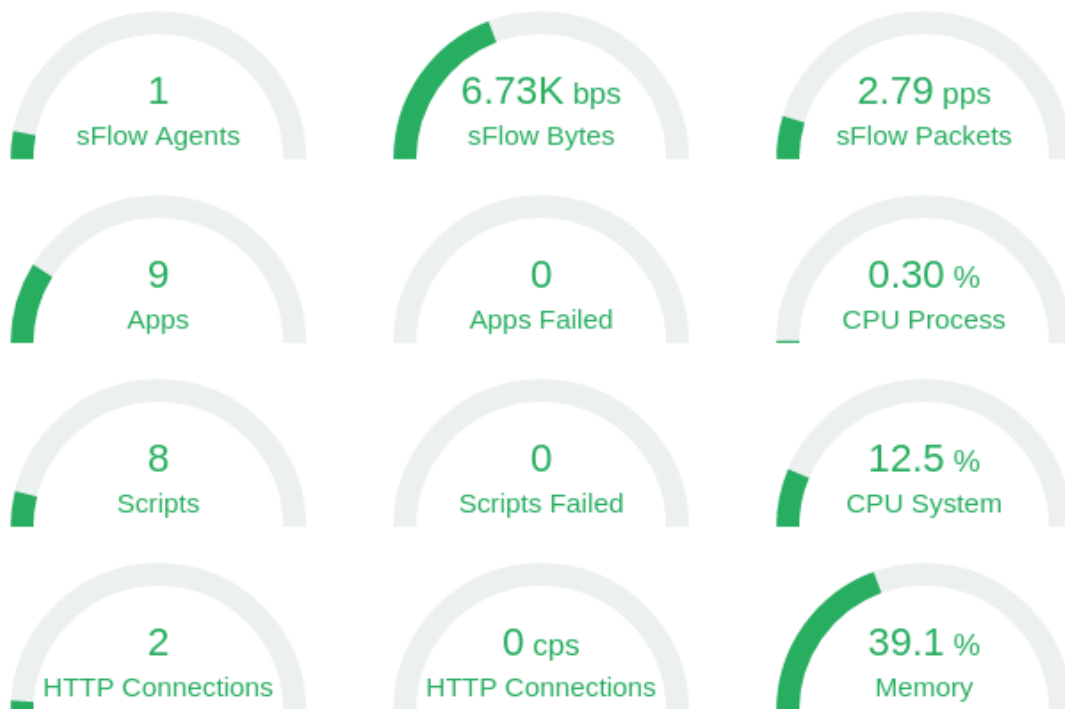
Gambar Lampiran 1.1 Dashboard Sflow RT Pada Lalu Lintas Normal



Gambar Lampiran 1.2 Flow dan Port Saat Lalu lintas Normal



Gambar Lampiran 1.3 Jumlah Flow saat Lalu Lintas Normal



Gambar Lampiran 1.4 Resource Saat Lalu Lintas Normal

```
"Node: h7"
root@ivan-virtual-machine:/home/ivan/Documents/DDoS Mitigation V5# iperf -c 10.0.0.1
-----
Client connecting to 10.0.0.1, TCP port 5001
TCP window size: 85.3 KByte (default)
-----
[ 1] local 10.0.0.7 port 41774 connected with 10.0.0.1 port 5001
[ ID] Interval      Transfer    Bandwidth
[ 1] 0.0000-10.0076 sec 6.24 GBytes 5.36 Gbits/sec
root@ivan-virtual-machine:/home/ivan/Documents/DDoS Mitigation V5#
```

Gambar Lampiran 1.5 Throughput Lalu Lintas Normal

```
"Node: h1"
64 bytes from 10.0.0.7: icmp_seq=82 ttl=64 time=0.052 ms
64 bytes from 10.0.0.7: icmp_seq=83 ttl=64 time=0.067 ms
64 bytes from 10.0.0.7: icmp_seq=84 ttl=64 time=0.102 ms
64 bytes from 10.0.0.7: icmp_seq=85 ttl=64 time=0.065 ms
64 bytes from 10.0.0.7: icmp_seq=86 ttl=64 time=0.065 ms
64 bytes from 10.0.0.7: icmp_seq=87 ttl=64 time=0.139 ms
64 bytes from 10.0.0.7: icmp_seq=88 ttl=64 time=0.084 ms
64 bytes from 10.0.0.7: icmp_seq=89 ttl=64 time=0.126 ms
64 bytes from 10.0.0.7: icmp_seq=90 ttl=64 time=0.174 ms
64 bytes from 10.0.0.7: icmp_seq=91 ttl=64 time=0.188 ms
64 bytes from 10.0.0.7: icmp_seq=92 ttl=64 time=0.110 ms
64 bytes from 10.0.0.7: icmp_seq=93 ttl=64 time=0.071 ms
64 bytes from 10.0.0.7: icmp_seq=94 ttl=64 time=0.128 ms
64 bytes from 10.0.0.7: icmp_seq=95 ttl=64 time=0.070 ms
64 bytes from 10.0.0.7: icmp_seq=96 ttl=64 time=0.233 ms
64 bytes from 10.0.0.7: icmp_seq=97 ttl=64 time=0.057 ms
64 bytes from 10.0.0.7: icmp_seq=98 ttl=64 time=0.174 ms
64 bytes from 10.0.0.7: icmp_seq=99 ttl=64 time=0.126 ms
64 bytes from 10.0.0.7: icmp_seq=100 ttl=64 time=8.50 ms

--- 10.0.0.7 ping statistics ---
100 packets transmitted, 100 received, 0% packet loss, time 101332ms
rtt min/avg/max/mdev = 0.052/0.280/8.495/1.100 ms
root@ivan-virtual-machine:/home/ivan/Documents/DDoS Mitigation V5#
```

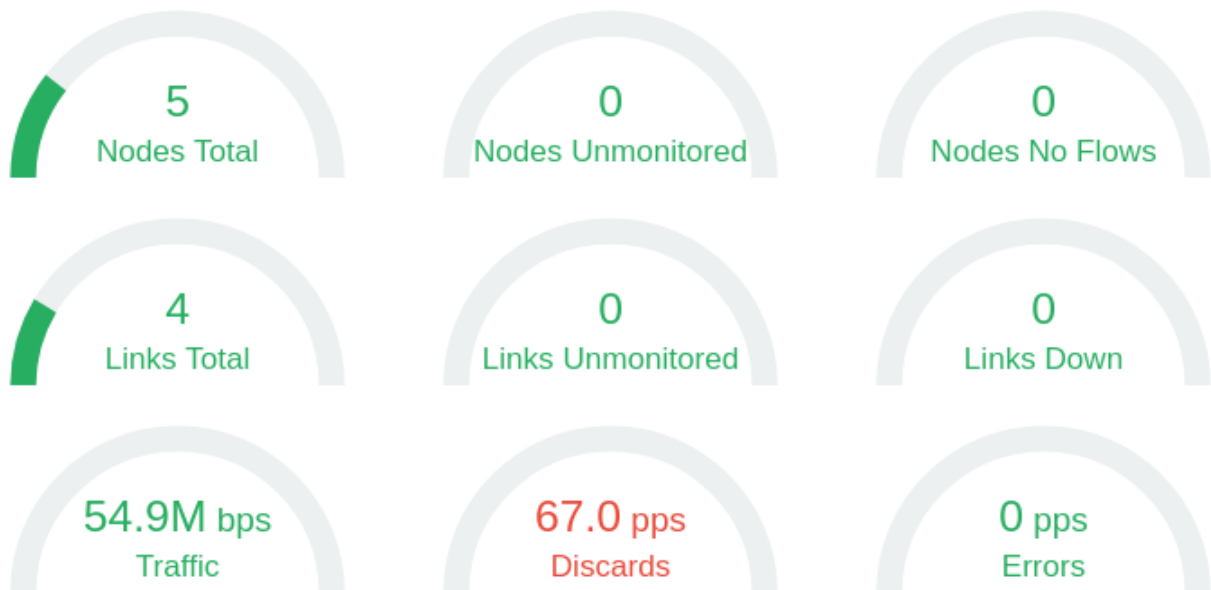
Gambar Lampiran 1.6 Packet Loss dan Latensi Saat Lalu Lintas Normal

```
Terminal
Traffic is mostly Normal.
-----
Traffic Analysis:
Normal traffic count: 264
DDoS traffic count: 0
Total traffic analyzed: 264
Traffic is mostly Normal.
-----
Traffic Analysis:
Normal traffic count: 2
DDoS traffic count: 0
Total traffic analyzed: 2
Traffic is mostly Normal.
-----
Traffic Analysis:
Normal traffic count: 2
DDoS traffic count: 0
Total traffic analyzed: 2
Traffic is mostly Normal.
-----
```

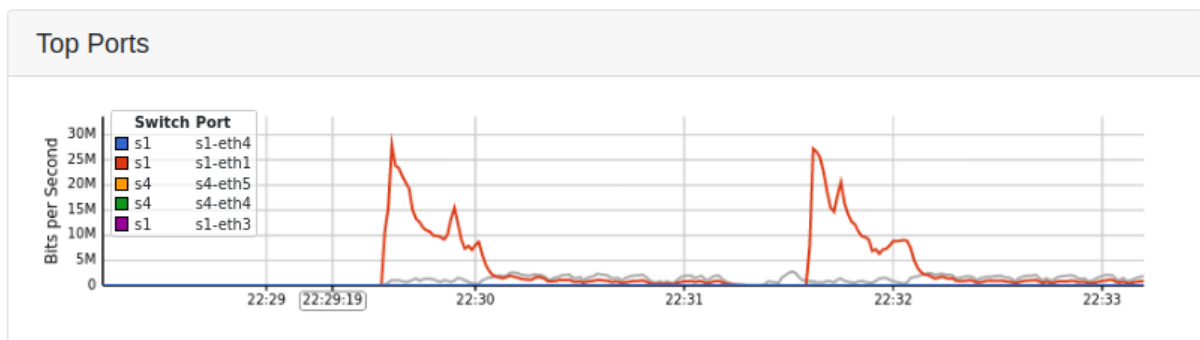
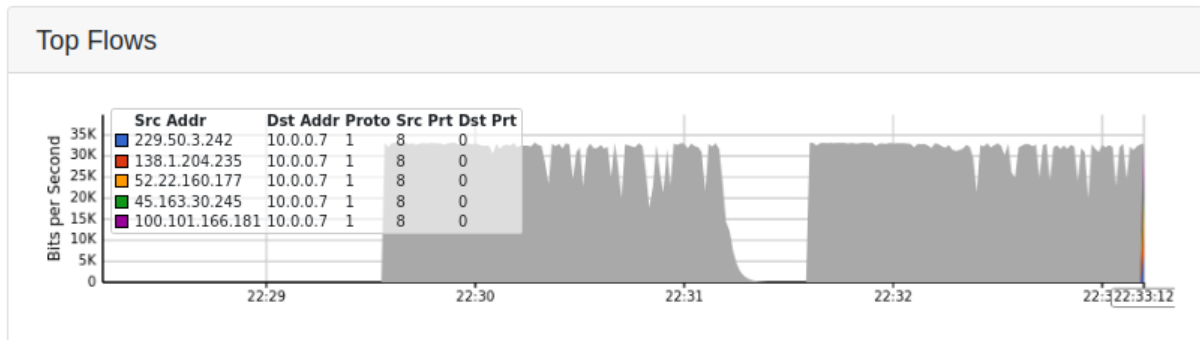
Gambar Lampiran 1.7 Hasil Deteksi Lalu Lintas Normal

A-2 Lalu Lintas DDoS Selama DDoS Terjadi

1. ICMP Flood



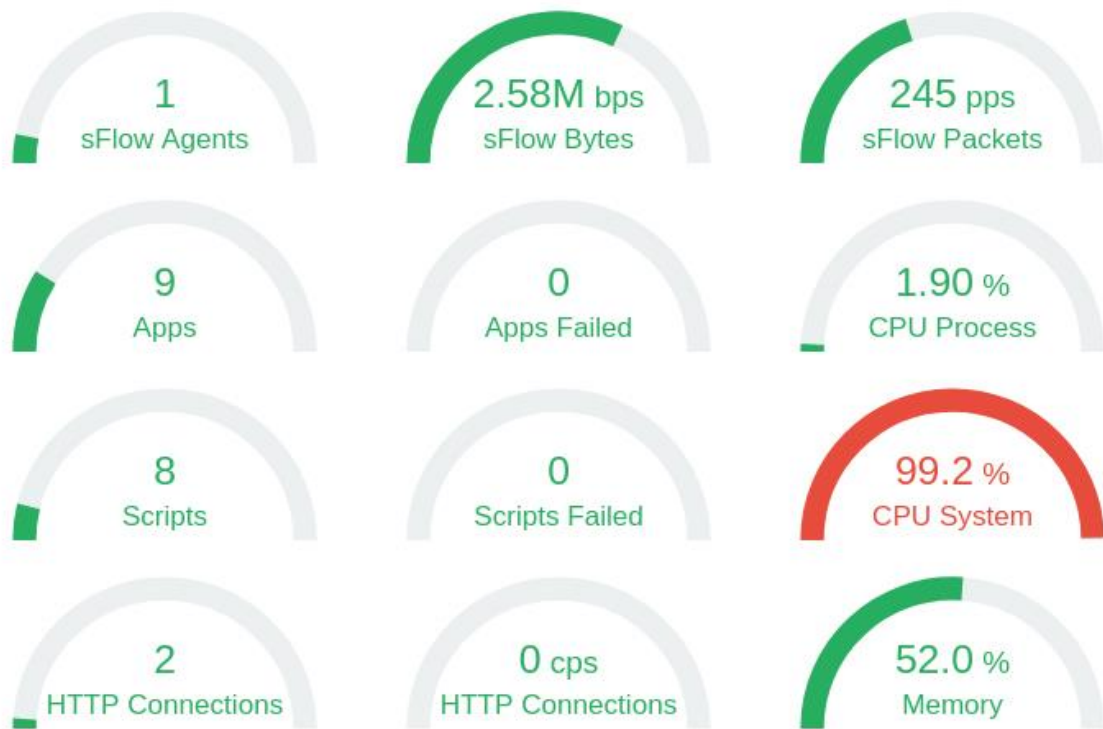
Gambar Lampiran 2.1 Dashboard Sflow RT Pada Lalu Lintas ICMP Flood Selama DDoS Terjadi



Gambar Lampiran 2.2 Flow dan Port Saat Lalu lintas ICMP Flood Selama DDoS Terjadi



Gambar Lampiran 2.3 Jumlah Flow saat Lalu Lintas ICMP Flood Selama DDoS Terjadi



Gambar Lampiran 2.4 Resource Saat Lalu Lintas ICMP Flood Selama DDoS Terjadi

```

"Node: h10"
root@ivan-virtual-machine:/home/ivan/Documents/DDoS Detection# iperf -c 10.0.0.2
tcp connect failed: No route to host
-----
Client connecting to 10.0.0.2, TCP port 5001
TCP window size: -1.00 Byte (default)
-----
[ 1] local 0.0.0.0 port 0 connected with 10.0.0.2 port 5001
root@ivan-virtual-machine:/home/ivan/Documents/DDoS Detection#

```

Gambar Lampiran 2.5 Throughput Lalu Lintas ICMP Flood Selama DDoS Terjadi

```
"Node: h10"
From 10.0.0.10 icmp_seq=54 Destination Host Unreachable
From 10.0.0.10 icmp_seq=55 Destination Host Unreachable
From 10.0.0.10 icmp_seq=56 Destination Host Unreachable
From 10.0.0.10 icmp_seq=57 Destination Host Unreachable
From 10.0.0.10 icmp_seq=58 Destination Host Unreachable
From 10.0.0.10 icmp_seq=59 Destination Host Unreachable
From 10.0.0.10 icmp_seq=60 Destination Host Unreachable
From 10.0.0.10 icmp_seq=61 Destination Host Unreachable
From 10.0.0.10 icmp_seq=62 Destination Host Unreachable
From 10.0.0.10 icmp_seq=63 Destination Host Unreachable
From 10.0.0.10 icmp_seq=64 Destination Host Unreachable
From 10.0.0.10 icmp_seq=65 Destination Host Unreachable
From 10.0.0.10 icmp_seq=66 Destination Host Unreachable
From 10.0.0.10 icmp_seq=67 Destination Host Unreachable
From 10.0.0.10 icmp_seq=68 Destination Host Unreachable
From 10.0.0.10 icmp_seq=69 Destination Host Unreachable
From 10.0.0.10 icmp_seq=70 Destination Host Unreachable
From 10.0.0.10 icmp_seq=71 Destination Host Unreachable
From 10.0.0.10 icmp_seq=72 Destination Host Unreachable

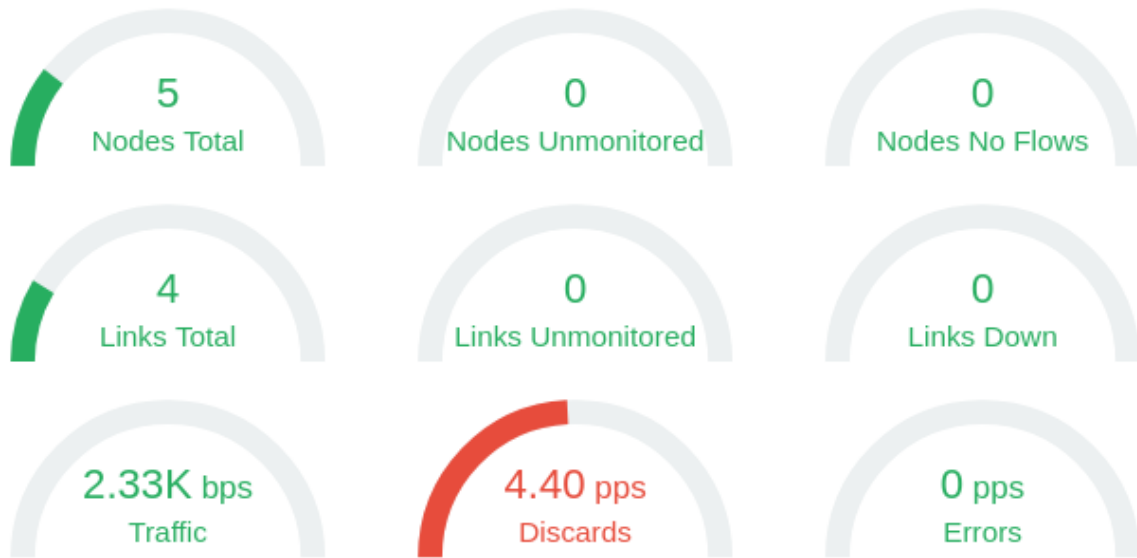
--- 10.0.0.2 ping statistics ---
100 packets transmitted, 0 received, +72 errors, 100% packet loss, time 101358ms
pipe 4
root@ivan-virtual-machine:/home/ivan/Documents/DDoS Detection#
```

Gambar Lampiran 2.6 Packet Loss dan Latensi Saat Lalu Lintas ICMP Flood Selama DDoS Terjadi

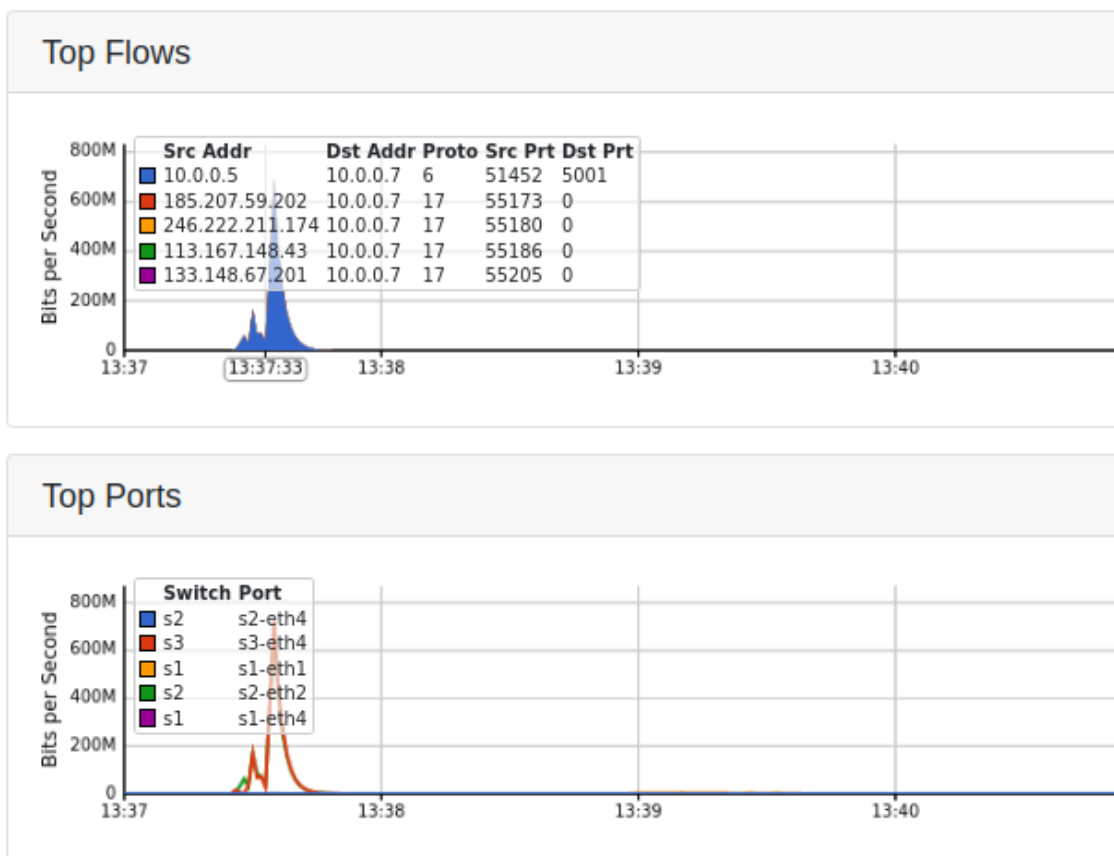
```
Terminal
Traffic Analysis:
Normal traffic count: 0
DDoS traffic count: 39
Total traffic analyzed: 39
DDoS traffic detected.
Potential victim host: h7
-----
Traffic Analysis:
Normal traffic count: 0
DDoS traffic count: 190
Total traffic analyzed: 190
DDoS traffic detected.
Potential victim host: h7
-----
Traffic Analysis:
Normal traffic count: 0
DDoS traffic count: 103
Total traffic analyzed: 103
DDoS traffic detected.
Potential victim host: h7
-----
```

Gambar Lampiran 2.7 Hasil Deteksi Lalu Lintas ICMP Flood Selama DDoS Terjadi

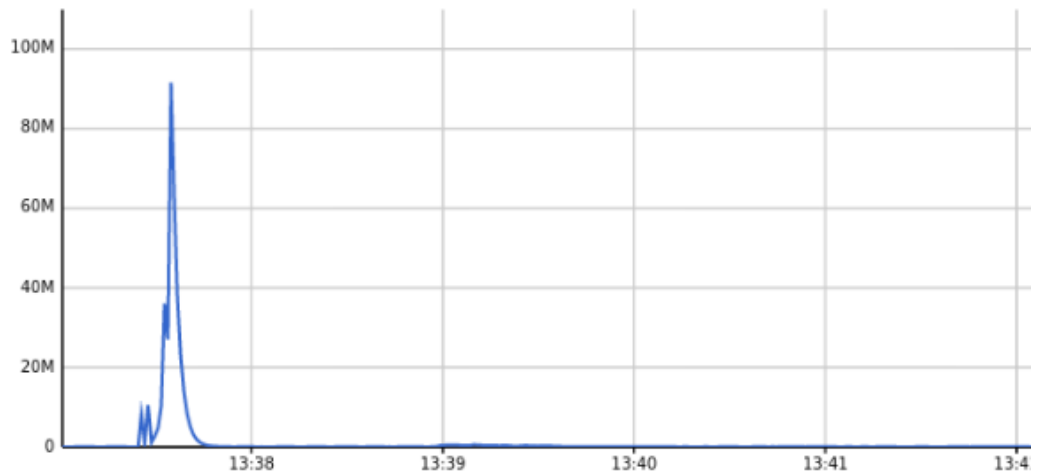
2. UDP Flood



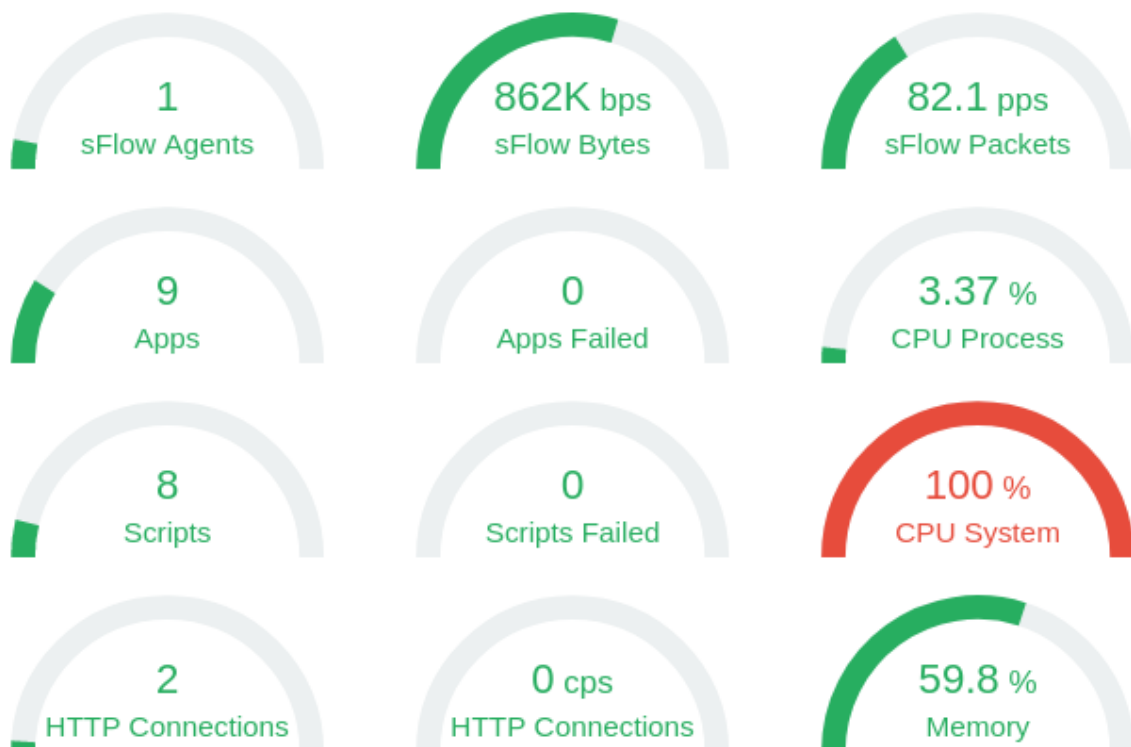
Gambar Lampiran 3.1 Dashboard Sflow RT Pada Lalu Lintas UDP Flood Selama DDoS Terjadi



Gambar Lampiran 3.2 Flow dan Port Saat Lalu lintas UDP Flood Selama DDoS Terjadi



Gambar Lampiran 3.3 Jumlah Flow saat Lalu Lintas UDP Flood Selama DDoS Terjadi



Gambar Lampiran 3.4 Resource Saat Lalu Lintas UDP Flood Selama DDoS Terjadi

```
"Node: h10"
root@ivan-virtual-machine:/home/ivan/Documents/DDoS Detection V3# iperf -c 10.0.0.2
tcp connect failed: No route to host
-----
Client connecting to 10.0.0.2, TCP port 5001
TCP window size: -1.00 Byte (default)
-----
[ 1] local 0.0.0.0 port 0 connected with 10.0.0.2 port 5001
root@ivan-virtual-machine:/home/ivan/Documents/DDoS Detection V3#
```

Gambar Lampiran 3.5 Throughput Lalu Lintas UDP Flood Selama DDoS Terjadi

```
"Node: h10"
From 10.0.0.10 icmp_seq=82 Destination Host Unreachable
From 10.0.0.10 icmp_seq=83 Destination Host Unreachable
From 10.0.0.10 icmp_seq=84 Destination Host Unreachable
From 10.0.0.10 icmp_seq=85 Destination Host Unreachable
From 10.0.0.10 icmp_seq=86 Destination Host Unreachable
From 10.0.0.10 icmp_seq=87 Destination Host Unreachable
From 10.0.0.10 icmp_seq=88 Destination Host Unreachable
From 10.0.0.10 icmp_seq=89 Destination Host Unreachable
From 10.0.0.10 icmp_seq=90 Destination Host Unreachable
From 10.0.0.10 icmp_seq=91 Destination Host Unreachable
From 10.0.0.10 icmp_seq=92 Destination Host Unreachable
From 10.0.0.10 icmp_seq=93 Destination Host Unreachable
From 10.0.0.10 icmp_seq=94 Destination Host Unreachable
From 10.0.0.10 icmp_seq=95 Destination Host Unreachable
From 10.0.0.10 icmp_seq=96 Destination Host Unreachable
From 10.0.0.10 icmp_seq=97 Destination Host Unreachable
From 10.0.0.10 icmp_seq=98 Destination Host Unreachable
From 10.0.0.10 icmp_seq=99 Destination Host Unreachable
From 10.0.0.10 icmp_seq=100 Destination Host Unreachable

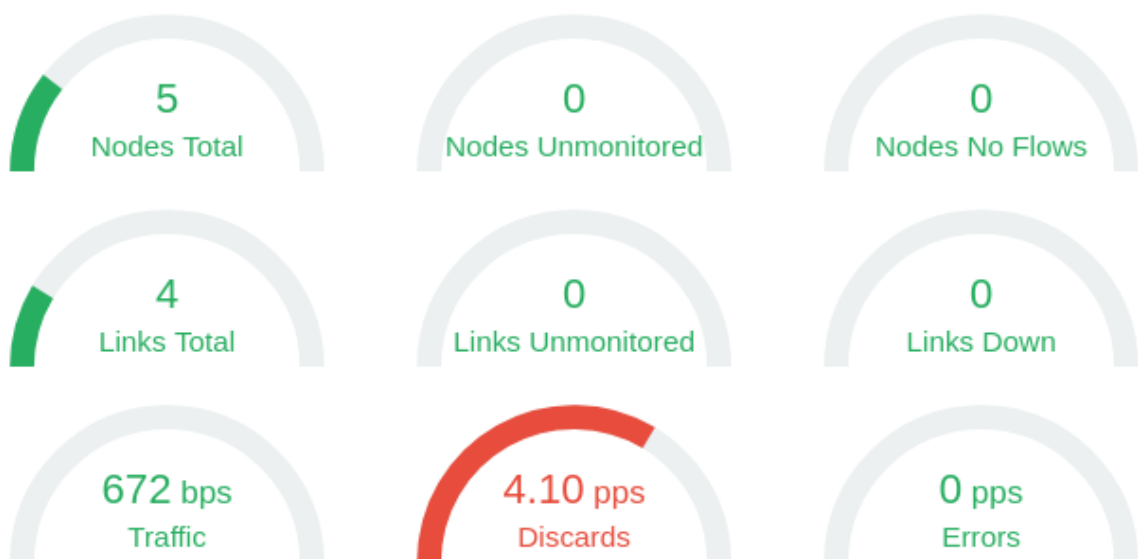
--- 10.0.0.2 ping statistics ---
100 packets transmitted, 0 received, +69 errors, 100% packet loss, time 101370ms
pipe 4
root@ivan-virtual-machine:/home/ivan/Documents/DDoS Detection V3#
```

Gambar Lampiran 3.6 Packet Loss dan Latensi Saat Lalu Lintas UDP Flood Selama DDoS Terjadi

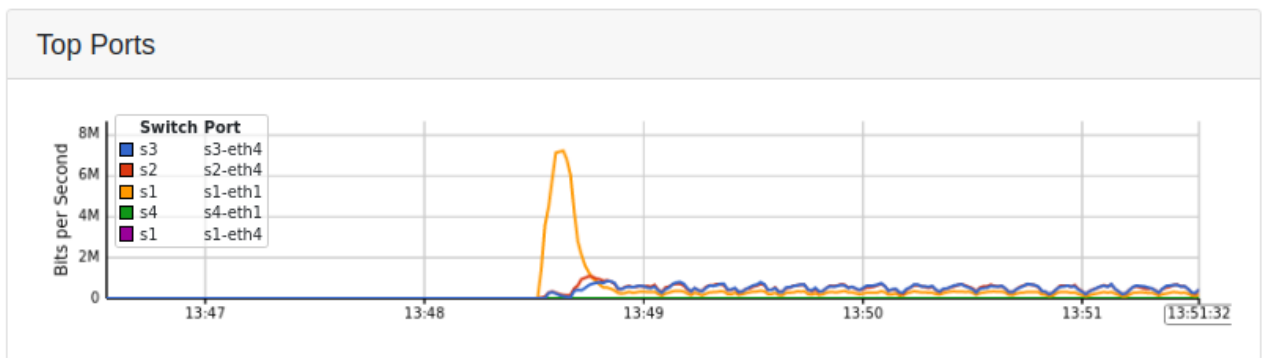
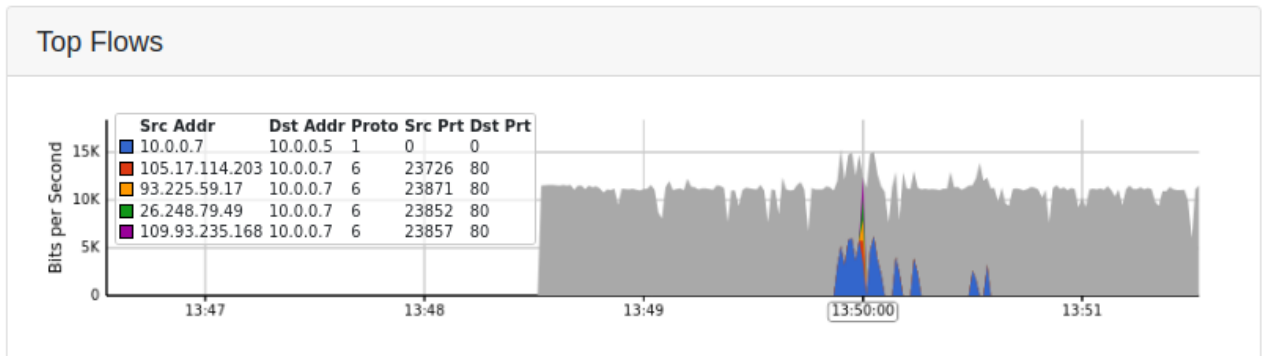
```
Terminal
Terminal x Terminal x v
Traffic Analysis:
Normal traffic count: 0
DDoS traffic count: 91
Total traffic analyzed: 91
DDoS traffic detected.
Potential victim host: h7
-----
Traffic Analysis:
Normal traffic count: 0
DDoS traffic count: 440
Total traffic analyzed: 440
DDoS traffic detected.
Potential victim host: h7
-----
Traffic Analysis:
Normal traffic count: 0
DDoS traffic count: 33
Total traffic analyzed: 33
DDoS traffic detected.
Potential victim host: h7
-----
```

Gambar Lampiran 3.7 Hasil Deteksi Lalu Lintas UDP Flood Selama DDoS Terjadi

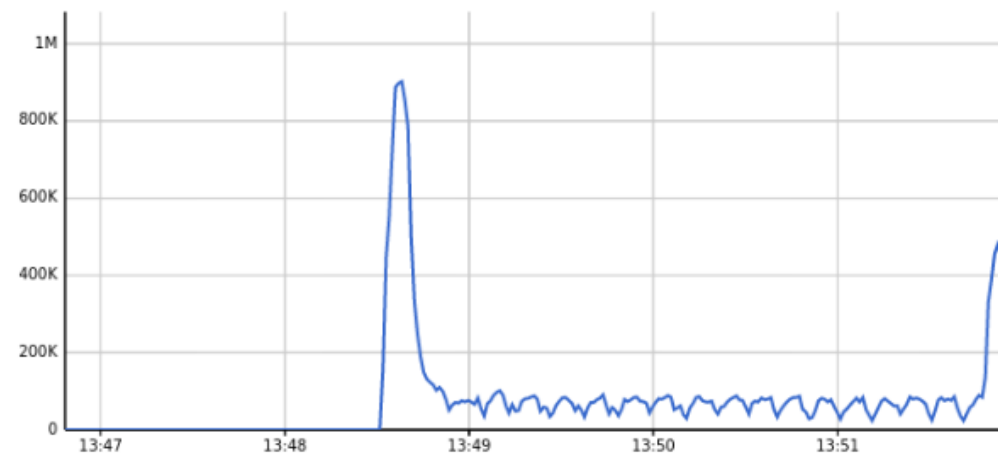
3. TCP SYN Flood



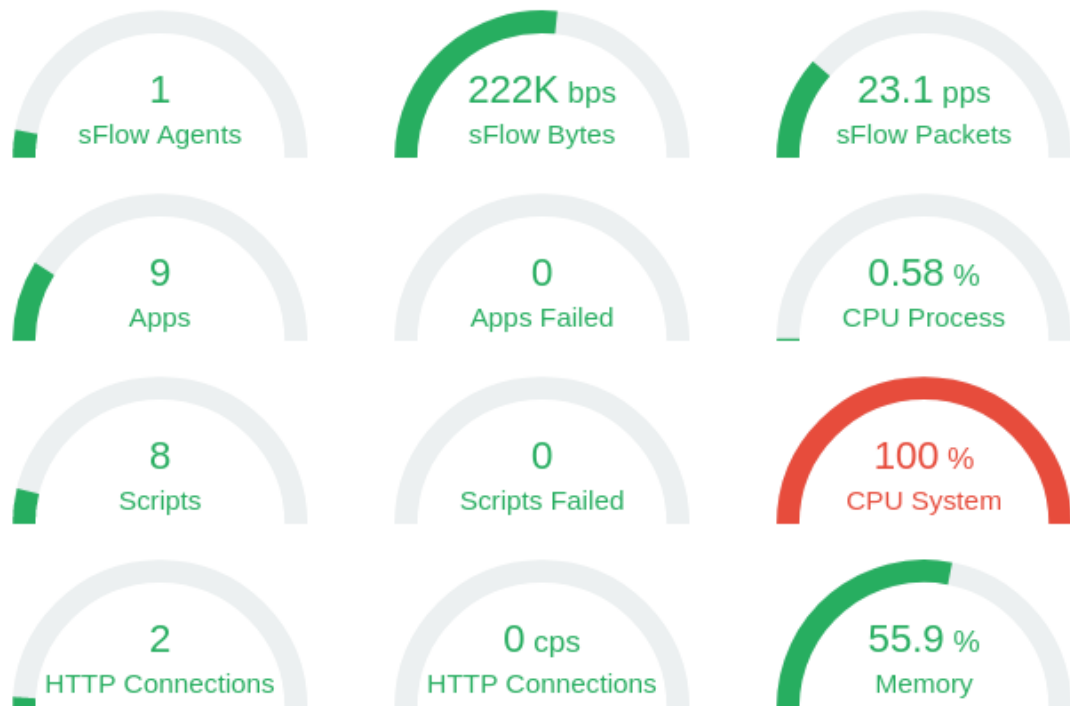
Gambar Lampiran 4.1 Dashboard Sflow RT Pada Lalu Lintas TCP SYN Flood Selama DDoS Terjadi



Gambar Lampiran 4.2 Flow dan Port Saat Lalu lintas TCP SYN Flood Selama DDoS Terjadi



Gambar Lampiran 4.3 Jumlah Flow saat Lalu Lintas TCP SYN Flood Selama DDoS Terjadi



Gambar Lampiran 4.4 Resource Saat Lalu Lintas TCP SYN Flood Selama DDoS Terjadi

```

"Node: h10"
root@ivan-virtual-machine:/home/ivan/Documents/DDoS Detection V3# iperf -c 10.0
.0.2
tcp connect failed: No route to host
-----
Client connecting to 10.0.0.2, TCP port 5001
TCP window size: -1.00 Byte (default)
-----
[ 1] local 0.0.0.0 port 0 connected with 10.0.0.2 port 5001
root@ivan-virtual-machine:/home/ivan/Documents/DDoS Detection V3#

```

Gambar Lampiran 4.5 Throughput Lalu Lintas TCP SYN Flood Selama DDoS Terjadi

```
"Node: h10"
From 10.0.0.10 icmp_seq=81 Destination Host Unreachable
From 10.0.0.10 icmp_seq=82 Destination Host Unreachable
From 10.0.0.10 icmp_seq=83 Destination Host Unreachable
From 10.0.0.10 icmp_seq=84 Destination Host Unreachable
From 10.0.0.10 icmp_seq=85 Destination Host Unreachable
From 10.0.0.10 icmp_seq=86 Destination Host Unreachable
From 10.0.0.10 icmp_seq=87 Destination Host Unreachable
From 10.0.0.10 icmp_seq=88 Destination Host Unreachable
From 10.0.0.10 icmp_seq=89 Destination Host Unreachable
From 10.0.0.10 icmp_seq=90 Destination Host Unreachable
From 10.0.0.10 icmp_seq=91 Destination Host Unreachable
From 10.0.0.10 icmp_seq=92 Destination Host Unreachable
From 10.0.0.10 icmp_seq=93 Destination Host Unreachable
From 10.0.0.10 icmp_seq=94 Destination Host Unreachable
From 10.0.0.10 icmp_seq=95 Destination Host Unreachable
From 10.0.0.10 icmp_seq=96 Destination Host Unreachable
From 10.0.0.10 icmp_seq=97 Destination Host Unreachable
From 10.0.0.10 icmp_seq=98 Destination Host Unreachable
From 10.0.0.10 icmp_seq=99 Destination Host Unreachable

--- 10.0.0.2 ping statistics ---
100 packets transmitted, 0 received, +99 errors, 100% packet loss, time 101368ms
pipe 4
root@ivan-virtual-machine:/home/ivan/Documents/DDoS Detection V3#
```

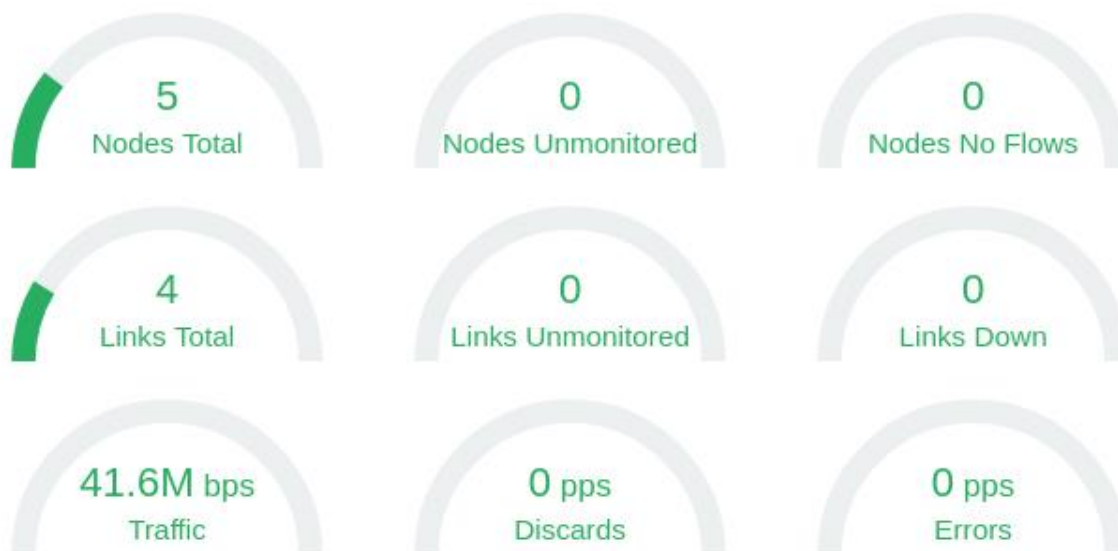
Gambar Lampiran 4.6 Packet Loss dan Latensi Saat Lalu Lintas TCP SYN Flood Selama DDoS Terjadi

```
Terminal
Traffic Analysis:
Normal traffic count: 0
DDoS traffic count: 41
Total traffic analyzed: 41
DDoS traffic detected.
Potential victim host: h7
-----
Traffic Analysis:
Normal traffic count: 0
DDoS traffic count: 321
Total traffic analyzed: 321
DDoS traffic detected.
Potential victim host: h7
-----
Traffic Analysis:
Normal traffic count: 0
DDoS traffic count: 199
Total traffic analyzed: 199
DDoS traffic detected.
Potential victim host: h7
-----
```

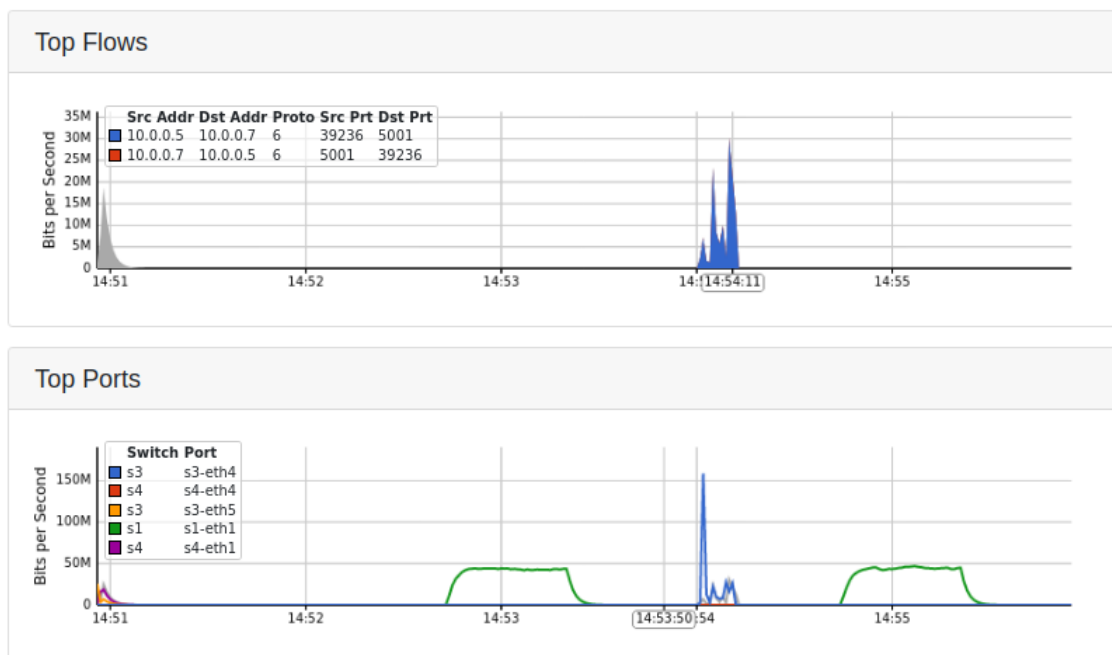
Gambar Lampiran 4.7 Hasil Deteksi Lalu Lintas TCP SYN Flood Selama DDoS Terjadi

A-3 Lalu Lintas DDoS Setelah Mitigasi

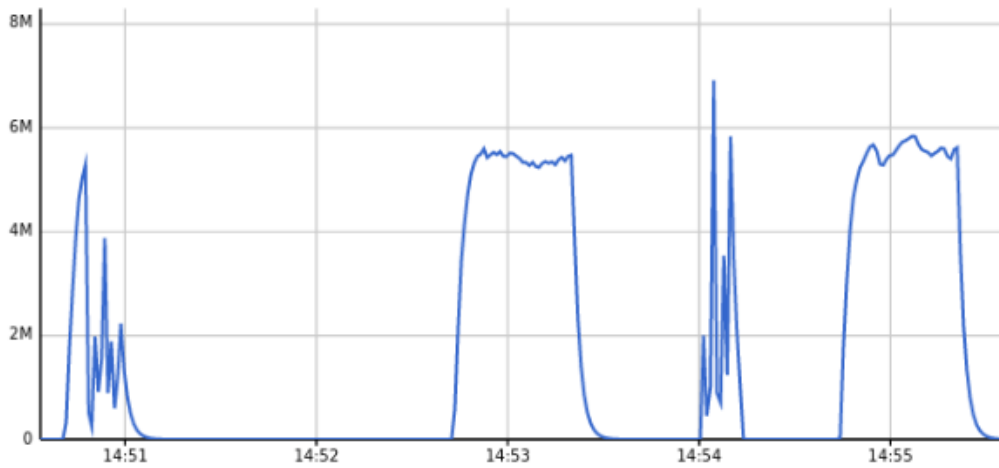
1. ICMP Flood



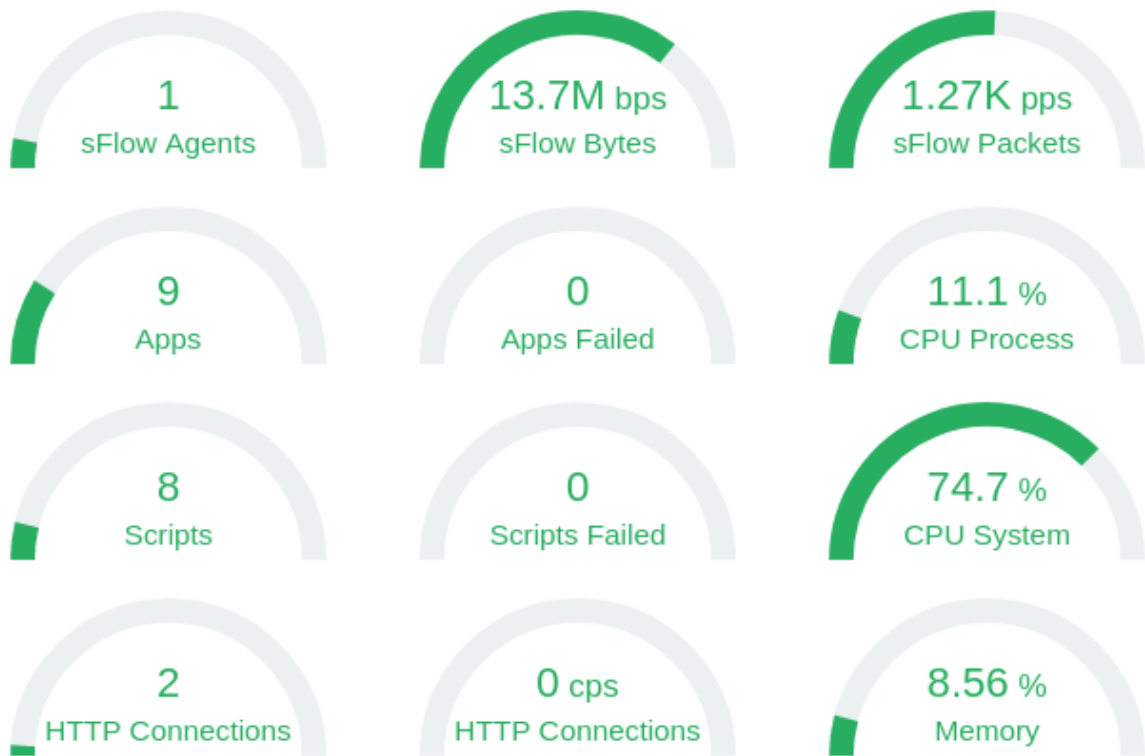
Gambar Lampiran 5.1 Dashboard Sflow RT Pada Lalu Lintas ICMP Flood Setelah Mitigasi



Gambar Lampiran 5.2 Flow dan Port Saat Lalu lintas ICMP Flood Setelah Mitigasi



Gambar Lampiran 5.3 Jumlah Flow saat Lalu Lintas ICMP Flood Setelah Mitigasi



Gambar Lampiran 5.4 Resource Saat Lalu Lintas ICMP Flood Setelah Mitigasi


```
"Node: h10"
root@ivan-virtual-machine:/home/ivan/Documents/DDoS Mitigation V8# iperf -c 10.0.0.2
-----
Client connecting to 10.0.0.2, TCP port 5001
TCP window size: 85.3 KByte (default)
-----
[ 1] local 10.0.0.10 port 51384 connected with 10.0.0.2 port 5001
[ ID] Interval      Transfer    Bandwidth
[ 1] 0.0000-10.0101 sec 3.75 GBytes 3.21 Gbits/sec
root@ivan-virtual-machine:/home/ivan/Documents/DDoS Mitigation V8#
```

Gambar Lampiran 5.5 Throughput Lalu Lintas ICMP Flood Setelah Mitigasi

```
"Node: h10"
64 bytes from 10.0.0.2: icmp_seq=82 ttl=64 time=0.063 ms
64 bytes from 10.0.0.2: icmp_seq=83 ttl=64 time=0.053 ms
64 bytes from 10.0.0.2: icmp_seq=84 ttl=64 time=0.090 ms
64 bytes from 10.0.0.2: icmp_seq=85 ttl=64 time=0.067 ms
64 bytes from 10.0.0.2: icmp_seq=86 ttl=64 time=0.041 ms
64 bytes from 10.0.0.2: icmp_seq=87 ttl=64 time=0.035 ms
64 bytes from 10.0.0.2: icmp_seq=88 ttl=64 time=0.051 ms
64 bytes from 10.0.0.2: icmp_seq=89 ttl=64 time=0.051 ms
64 bytes from 10.0.0.2: icmp_seq=90 ttl=64 time=0.074 ms
64 bytes from 10.0.0.2: icmp_seq=91 ttl=64 time=0.080 ms
64 bytes from 10.0.0.2: icmp_seq=92 ttl=64 time=0.056 ms
64 bytes from 10.0.0.2: icmp_seq=93 ttl=64 time=0.063 ms
64 bytes from 10.0.0.2: icmp_seq=94 ttl=64 time=0.047 ms
64 bytes from 10.0.0.2: icmp_seq=95 ttl=64 time=0.094 ms
64 bytes from 10.0.0.2: icmp_seq=96 ttl=64 time=0.103 ms
64 bytes from 10.0.0.2: icmp_seq=97 ttl=64 time=0.132 ms
64 bytes from 10.0.0.2: icmp_seq=98 ttl=64 time=0.066 ms
64 bytes from 10.0.0.2: icmp_seq=99 ttl=64 time=7.97 ms
64 bytes from 10.0.0.2: icmp_seq=100 ttl=64 time=0.066 ms

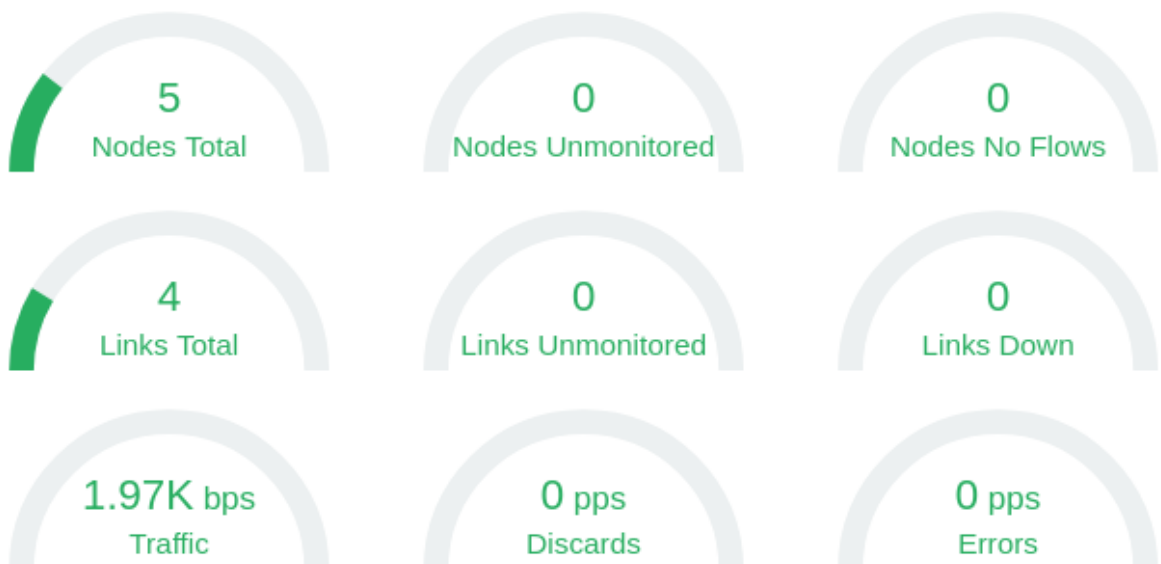
--- 10.0.0.2 ping statistics ---
100 packets transmitted, 100 received, 0% packet loss, time 101295ms
rtt min/avg/max/mdev = 0.028/0.242/8.717/1.162 ms
root@ivan-virtual-machine:/home/ivan/Documents/DDoS Mitigation V8#
```

Gambar Lampiran 5.6 Packet Loss dan Latensi Saat Lalu Lintas ICMP Flood Setelah Mitigasi

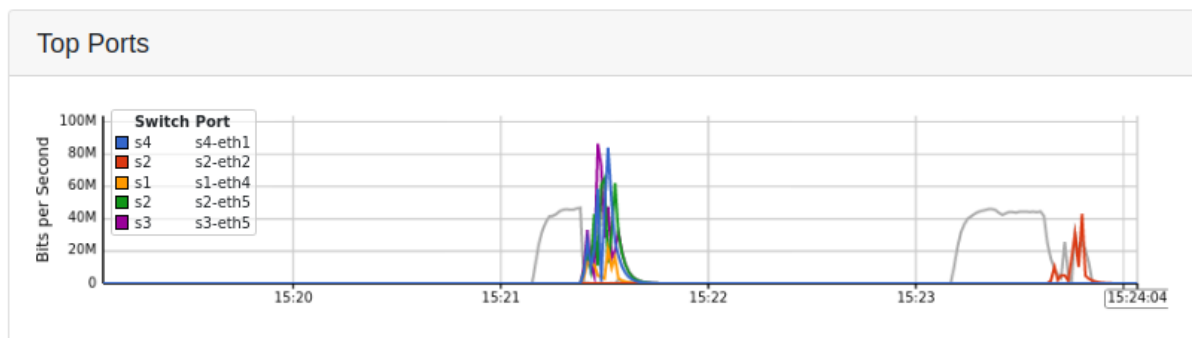
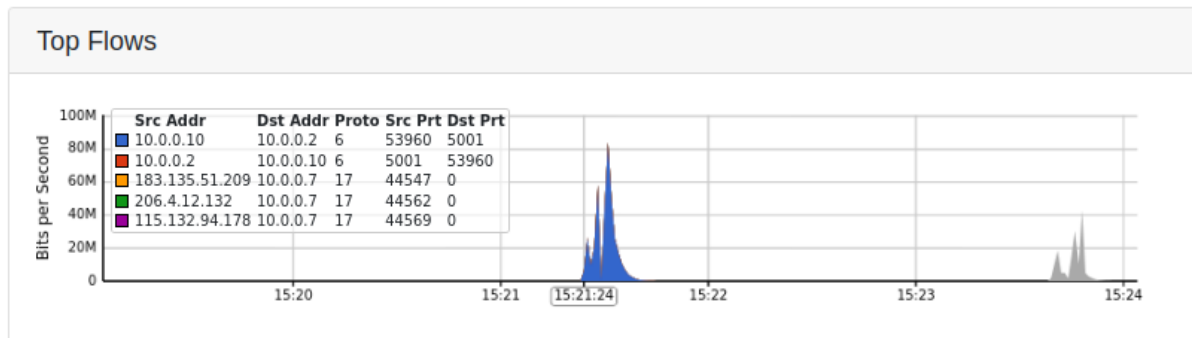
```
Terminal
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
-----
Traffic Analysis:
Legitimate Traffic = 0
DoS Traffic = 470
Total Traffic analyzed: 470
DDoS traffic detected.
Victim Host: h7
Mitigation process in progress!
-----
```

Gambar Lampiran 5.7 Hasil Deteksi Lalu Lintas ICMP Flood Setelah Mitigasi

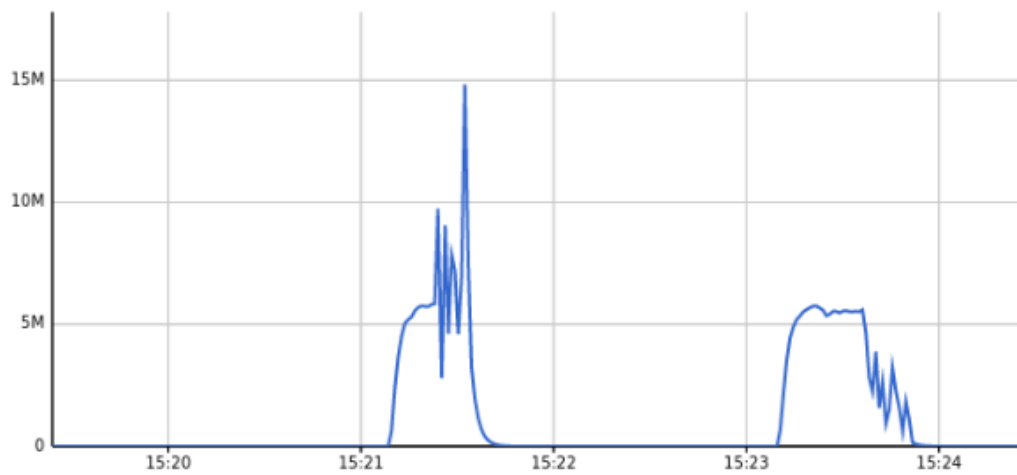
2. UDP Flood



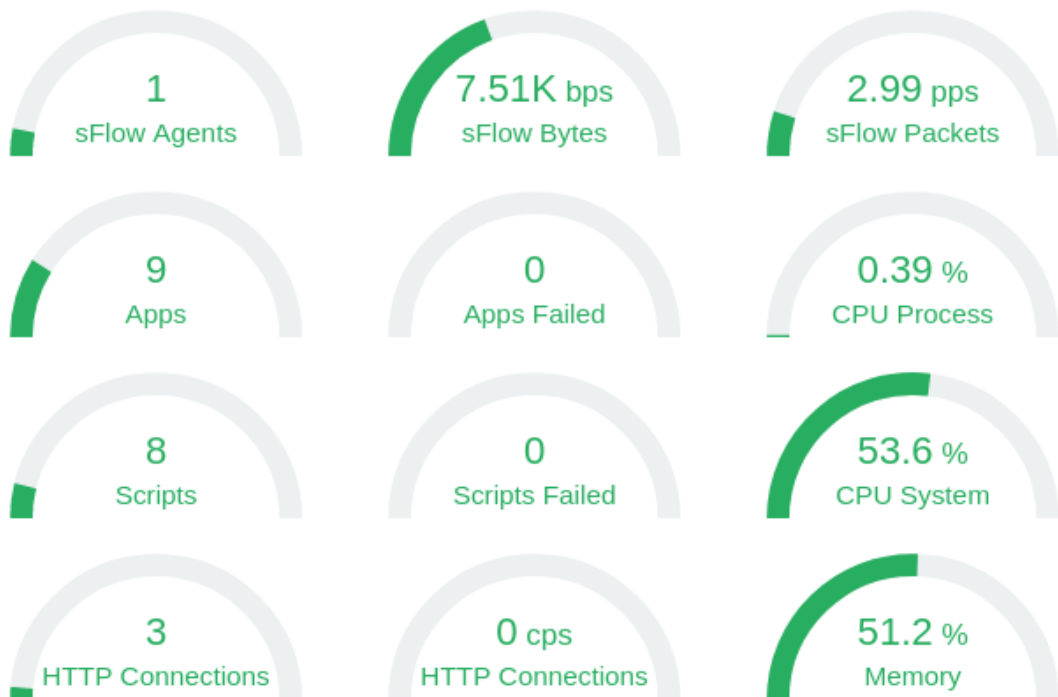
Gambar Lampiran 2.1 Dashboard Sflow RT Pada Lalu Lintas UDP Flood Setelah Mitigasi



Gambar Lampiran 2.2 Flow dan Port Saat Lalu lintas UDP Flood Setelah Mitigasi



Gambar Lampiran 2.3 Jumlah Flow saat Lalu Lintas UDP Flood Setelah Mitigasi



Gambar Lampiran 2.4 Resource Saat Lalu Lintas UDP Flood Setelah Mitigasi

```

"Node: h10"
root@ivan-virtual-machine:/home/ivan/Documents/DDoS Mitigation V8# iperf -c 10.0.0.2
-----
Client connecting to 10.0.0.2, TCP port 5001
TCP window size: 85.3 kByte (default)
-----
[ 1] local 10.0.0.10 port 53960 connected with 10.0.0.2 port 5001
[ ID] Interval      Transfer    Bandwidth
[ 1] 0.0000-10.0249 sec 3.61 GBytes 3.09 Gbits/sec
root@ivan-virtual-machine:/home/ivan/Documents/DDoS Mitigation V8#

```

Gambar Lampiran 2.5 Throughput Lalu Lintas UDP Flood Setelah Mitigasi

```
"Node: h10"
64 bytes from 10.0.0.2: icmp_seq=82 ttl=64 time=0,065 ms
64 bytes from 10.0.0.2: icmp_seq=83 ttl=64 time=0,049 ms
64 bytes from 10.0.0.2: icmp_seq=84 ttl=64 time=0,051 ms
64 bytes from 10.0.0.2: icmp_seq=85 ttl=64 time=0,068 ms
64 bytes from 10.0.0.2: icmp_seq=86 ttl=64 time=0,052 ms
64 bytes from 10.0.0.2: icmp_seq=87 ttl=64 time=0,052 ms
64 bytes from 10.0.0.2: icmp_seq=88 ttl=64 time=0,064 ms
64 bytes from 10.0.0.2: icmp_seq=89 ttl=64 time=0,062 ms
64 bytes from 10.0.0.2: icmp_seq=90 ttl=64 time=0,066 ms
64 bytes from 10.0.0.2: icmp_seq=91 ttl=64 time=0,089 ms
64 bytes from 10.0.0.2: icmp_seq=92 ttl=64 time=0,072 ms
64 bytes from 10.0.0.2: icmp_seq=93 ttl=64 time=0,094 ms
64 bytes from 10.0.0.2: icmp_seq=94 ttl=64 time=0,086 ms
64 bytes from 10.0.0.2: icmp_seq=95 ttl=64 time=0,117 ms
64 bytes from 10.0.0.2: icmp_seq=96 ttl=64 time=0,078 ms
64 bytes from 10.0.0.2: icmp_seq=97 ttl=64 time=0,116 ms
64 bytes from 10.0.0.2: icmp_seq=98 ttl=64 time=0,084 ms
64 bytes from 10.0.0.2: icmp_seq=99 ttl=64 time=0,088 ms
64 bytes from 10.0.0.2: icmp_seq=100 ttl=64 time=20,4 ms

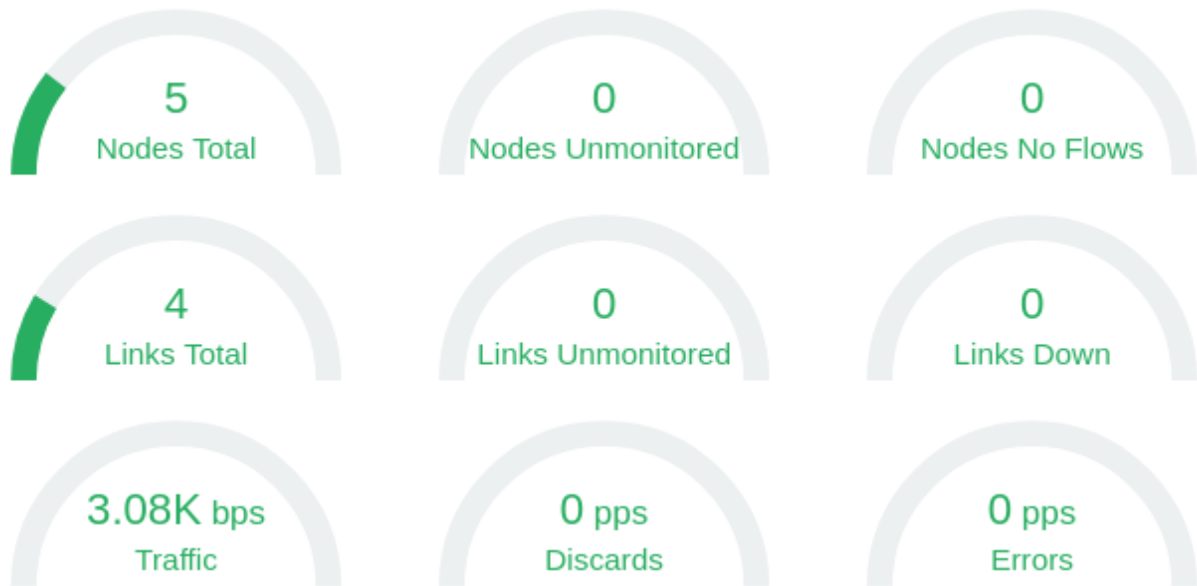
--- 10.0.0.2 ping statistics ---
100 packets transmitted, 100 received, 0% packet loss, time 101327ms
rtt min/avg/max/mdev = 0,037/0,400/20,409/2,322 ms
root@ivan-virtual-machine:/home/ivan/Documents/DDoS Mitigation V8#
```

Gambar Lampiran 2.6 Packet Loss dan Latensi Saat Lalu Lintas UDP Flood Setelah Mitigasi

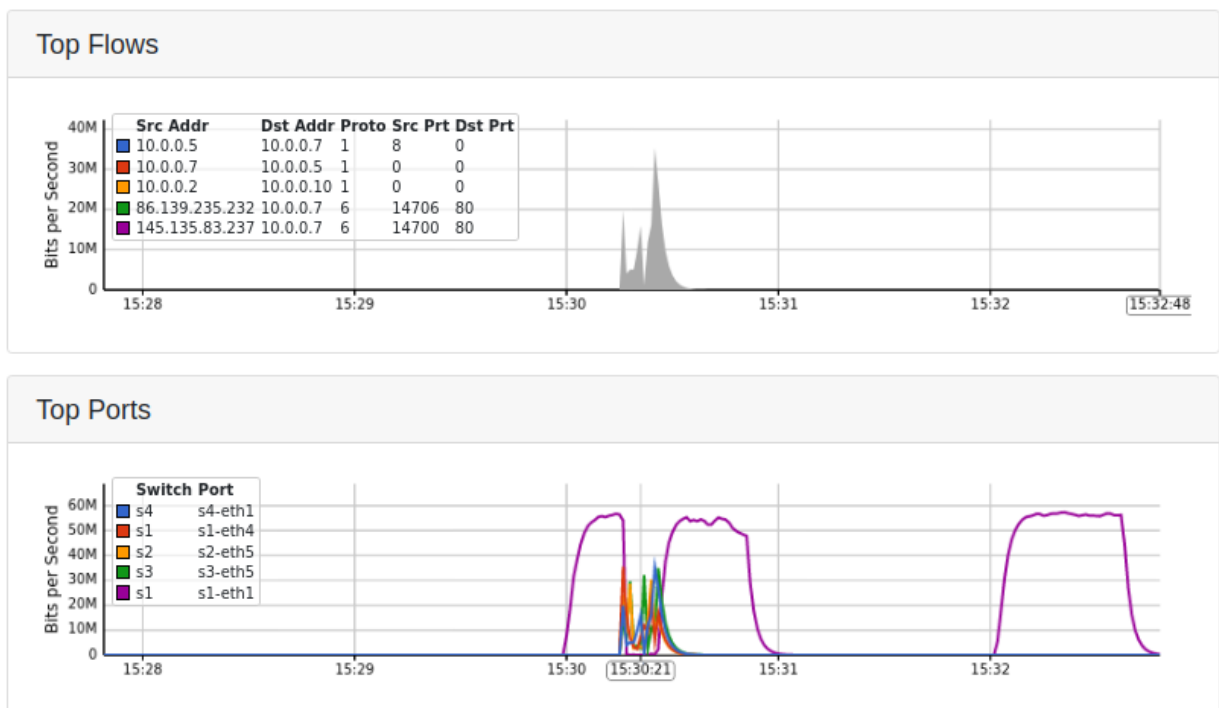
```
Terminal
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
-----
Traffic Analysis:
Legitimate Traffic = 0
DoS Traffic = 526
Total Traffic analyzed: 526
DDoS traffic detected.
Victim Host: h7
Mitigation process in progress!
-----
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
```

Gambar Lampiran 2.7 Hasil Deteksi Lalu Lintas UDP Flood Setelah Mitigasi

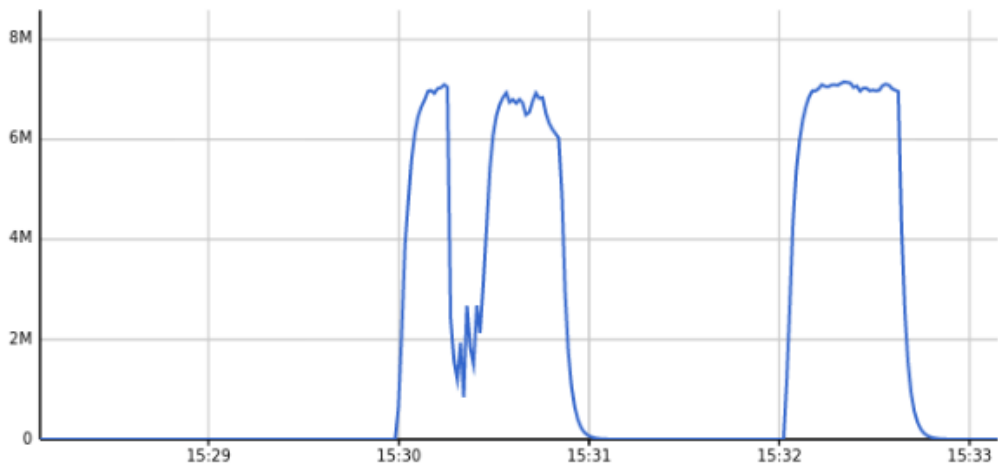
3. TCP SYN Flood



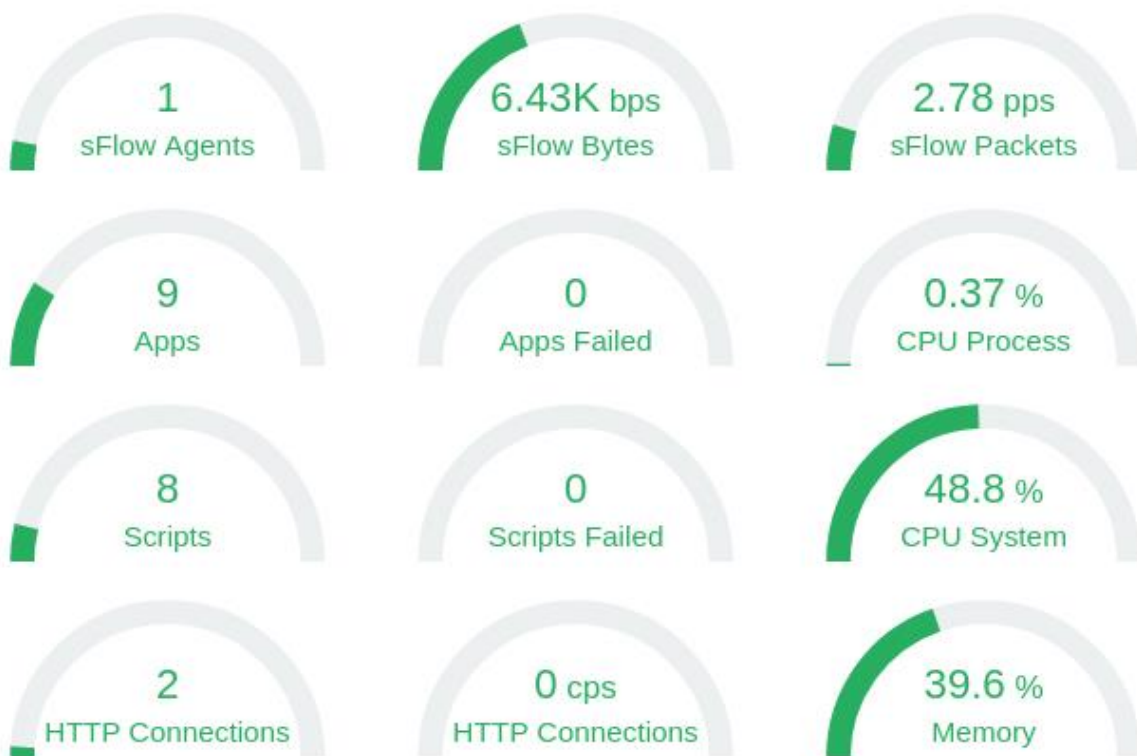
Gambar Lampiran 2.1 Dashboard Sflow RT Pada Lalu Lintas TCP SYN Setelah Mitigasi



Gambar Lampiran 2.2 Flow dan Port Saat Lalu lintas TCP SYN Setelah Mitigasi



Gambar Lampiran 2.3 Jumlah Flow saat Lalu Lintas TCP SYN Setelah Mitigasi



Gambar Lampiran 2.4 Resource Saat Lalu Lintas TCP SYN Setelah Mitigasi

```
"Node: h10"
root@ivan-virtual-machine:/home/ivan/Documents/DDoS Mitigation V8# iperf -c 10.0.0.2
-----
Client connecting to 10.0.0.2, TCP port 5001
TCP window size: 85.3 KByte (default)
-----
[ 1] local 10.0.0.10 port 52104 connected with 10.0.0.2 port 5001
[ ID] Interval      Transfer    Bandwidth
[ 1] 0.0000-10.0054 sec 4.06 GBytes 3.49 Gbits/sec
root@ivan-virtual-machine:/home/ivan/Documents/DDoS Mitigation V8#
```

Gambar Lampiran 2.5 Throughput Lalu Lintas TCP SYN Setelah Mitigasi

```
"Node: h10"
64 bytes from 10.0.0.2: icmp_seq=82 ttl=64 time=0.099 ms
64 bytes from 10.0.0.2: icmp_seq=83 ttl=64 time=0.065 ms
64 bytes from 10.0.0.2: icmp_seq=84 ttl=64 time=0.074 ms
64 bytes from 10.0.0.2: icmp_seq=85 ttl=64 time=0.055 ms
64 bytes from 10.0.0.2: icmp_seq=86 ttl=64 time=0.100 ms
64 bytes from 10.0.0.2: icmp_seq=87 ttl=64 time=0.074 ms
64 bytes from 10.0.0.2: icmp_seq=88 ttl=64 time=0.051 ms
64 bytes from 10.0.0.2: icmp_seq=89 ttl=64 time=0.081 ms
64 bytes from 10.0.0.2: icmp_seq=90 ttl=64 time=0.074 ms
64 bytes from 10.0.0.2: icmp_seq=91 ttl=64 time=0.098 ms
64 bytes from 10.0.0.2: icmp_seq=92 ttl=64 time=0.068 ms
64 bytes from 10.0.0.2: icmp_seq=93 ttl=64 time=0.055 ms
64 bytes from 10.0.0.2: icmp_seq=94 ttl=64 time=0.054 ms
64 bytes from 10.0.0.2: icmp_seq=95 ttl=64 time=0.054 ms
64 bytes from 10.0.0.2: icmp_seq=96 ttl=64 time=0.051 ms
64 bytes from 10.0.0.2: icmp_seq=97 ttl=64 time=0.084 ms
64 bytes from 10.0.0.2: icmp_seq=98 ttl=64 time=0.070 ms
64 bytes from 10.0.0.2: icmp_seq=99 ttl=64 time=11.0 ms
64 bytes from 10.0.0.2: icmp_seq=100 ttl=64 time=0.057 ms

--- 10.0.0.2 ping statistics ---
100 packets transmitted, 100 received, 0% packet loss, time 101298ms
rtt min/avg/max/mdev = 0.040/0.375/17.544/2.045 ms
root@ivan-virtual-machine:/home/ivan/Documents/DDoS Mitigation V8#
```

Gambar Lampiran 2.6 Packet Loss dan Latensi Saat Lalu Lintas TCP SYN Setelah Mitigasi


```
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
Blocking port 1 in switch 1
-----
Traffic Analysis:
Legitimate Traffic = 0
DoS Traffic = 100
Total Traffic analyzed: 100
DDoS traffic detected.
Victim Host: h7
Mitigation process in progress!
```

Gambar Lampiran 2.7 Hasil Deteksi Lalu Lintas TCP SYN Setelah Mitigasi

LAMPIRAN B KODE PROGRAM

1. Pembuatan Topologi

```
from mininet.topo import Topo
from mininet.net import Mininet
from mininet.link import TCLink
from mininet.log import setLogLevel
from mininet.cli import CLI
from mininet.node import OVSKernelSwitch, RemoteController
# Impor fungsi wrapper dari sflow.py
from sflow import wrapper

class MyTopo(Topo):
    def build(self):
        switches = []
        hosts = []

        for s in range(1, 6):
            switch_name = 's{}'.format(s)
            switch = self.addSwitch(switch_name,
cls=OVSKernelSwitch, protocols='OpenFlow13')
            switches.append(switch)

        for h in range(1, 4):
            host_name = 'h{}'.format(s * 3 + h - 3)
            host_ip = '10.0.0.{}'.format(s * 3 + h - 3)
            host = self.addHost(host_name, cpu=1.0/20,
mac="00:00:00:00:00:{}".format(s * 3 + h - 3), ip=host_ip)
            hosts.append(host)
            self.addLink(host, switch)

        for i in range(len(switches) - 1):
            self.addLink(switches[i], switches[i + 1])

    def startNetwork():
        topo = MyTopo()
        c0 = RemoteController('c0', ip='127.0.0.1', port=6653)
        net = Mininet(topo=topo, link=TCLink, controller=c0)

        # Menggunakan fungsi wrapper dari sflow.py
        wrapper(net)

        net.start()
        CLI(net)
        net.stop()

if __name__ == '__main__':
    setLogLevel('info')
    startNetwork()
```

2. Pembuatan dataset

a. Traffic generator

```
from mininet.topo import Topo
from mininet.net import Mininet
from mininet.link import TCLink
from mininet.log import setLogLevel
from mininet.node import OVSKernelSwitch, RemoteController
from time import sleep
from datetime import datetime
from random import randrange, choice

class CustomTopo(Topo):
    def build(self):
        switches = []
        hosts = []

        for s in range(1, 6):
            switch = self.addSwitch(f's{s}',
cls=OVSKernelSwitch, protocols='OpenFlow13')
            switches.append(switch)

            for h in range(1, 4):
                host_id = s * 3 + h - 3
                host_ip = f'10.0.0.{host_id}/24'
                host_mac = f"00:00:00:00:00:{host_id:02d}"
                host = self.addHost(f'h{host_id}',
cpu=1.0/20, mac=host_mac, ip=host_ip)
                hosts.append(host)
                self.addLink(host, switch)

            for i in range(len(switches) - 1):
                self.addLink(switches[i], switches[i + 1])

    def generate_ip():
        return f"10.0.0.{randrange(1, 16)}"

    def launch_network():
        topology = CustomTopo()
        controller = RemoteController('c0', ip='127.0.0.1')
```

```

network = Mininet(topo=topology, link=TCLink,
controller=controller)
network.start()

server_host = network.get('h1')
server_host.cmd('cd /home/mininet/webserver')
server_host.cmd('python -m SimpleHTTPServer 80 &')

hosts = [network.get(f'h{i}') for i in range(1, 16)]

for attack, cmd in [("ICMP Flood", "-1"), ("UDP Flood",
"-2"), ("TCP SYN Flood", "-S")]:
    attacker = choice(hosts)
    target_ip = generate_ip()
    print(f"-----")
    print(f"Performing {attack}")
    print(f"-----")
    attacker.cmd(f"timeout 20s hping3 {cmd} -V -d 120
-w 64 -p 80 --rand-source --flood {target_ip}")
    sleep(100)

network.stop()

if __name__ == '__main__':
    start_time = datetime.now()

    setLogLevel('info')
    launch_network()

    end_time = datetime.now()
    print(end_time - start_time)

```

b. Ekstraksi dataset

```

Cimport topologi_kontroler
from ryu.controller import ofp_event
from ryu.controller.handler import MAIN_DISPATCHER,
DEAD_DISPATCHER
from ryu.controller.handler import set ev cls

```

```

from ryu.lib import hub

from datetime import datetime

class
StatsCollectionApp(topologi_kontroler.SimpleSwitch13):
    def __init__(self, *args, **kwargs):
        super(StatsCollectionApp, self).__init__(*args,
**kwargs)
        self.datapaths = {}
        self.monitor_thread = hub.spawn(self._monitor)

        @set_ev_cls(ofp_event.EventOFPPStateChange,
[MAIN_DISPATCHER, DEAD_DISPATCHER])
        def _on_state_change(self, ev):
            datapath = ev.datapath
            if ev.state == MAIN_DISPATCHER:
                self._register_datapath(datapath)
            elif ev.state == DEAD_DISPATCHER:
                self._unregister_datapath(datapath)

        def _register_datapath(self, datapath):
            self.logger.debug('Registering datapath: %016x',
datapath.id)
            self.datapaths[datapath.id] = datapath

        def _unregister_datapath(self, datapath):
            self.logger.debug('Unregistering datapath: %016x',
datapath.id)
            if datapath.id in self.datapaths:
                del self.datapaths[datapath.id]

        def _monitor(self):
            while True:
                for dp in self.datapaths.values():
                    self._send_stats_request(dp)
                    hub.sleep(10)

        def _send_stats_request(self, datapath):
            self.logger.debug('Sending stats request: %016x',
datapath.id)
            parser = datapath.ofproto_parser
            req = parser.OFPFlowStatsRequest(datapath)
            datapath.send_msg(req)

        @set_ev_cls(ofp_event.EventOFPFlowStatsReply,
MAIN_DISPATCHER)
        def _handle_flow_stats_reply(self, ev):
            timestamp = datetime.now().timestamp()
            with open("FlowStatsfile.csv", "a") as file:
                for stat in sorted(ev.msg.body, key=lambda f:
(f.match.get('eth_type'), f.match.get('ipv4_src'),
f.match.get('ipv4_dst'), f.match.get('ip_proto'))):
                    if stat.priority == 1:
                        ip_src = stat.match.get('ipv4_src',
'0.0.0.0')
                        ip_dst = stat.match.get('ipv4_dst',
'0.0.0.0')

```

```

0)         ip_proto = stat.match.get('ip_proto',
icmp_code, icmp_type, tp_src, tp_dst =
-1, -1, 0, 0
         if ip_proto == 1:
             icmp_code =
stat.match.get('icmpv4_code', -1)
             icmp_type =
stat.match.get('icmpv4_type', -1)
         elif ip_proto == 6:
             tp_src = stat.match.get('tcp_src',
0)
             tp_dst = stat.match.get('tcp_dst',
0)
         elif ip_proto == 17:
             tp_src = stat.match.get('udp_src',
0)
             tp_dst = stat.match.get('udp_dst',
0)
         flow_id = f"{ip_src}-{tp_src}-
{ip_dst}-{tp_dst}-{ip_proto}"
         pps = stat.packet_count /
stat.duration_sec if stat.duration_sec else 0
         pns = stat.packet_count /
stat.duration_nsec if stat.duration_nsec else 0
         bps = stat.byte_count /
stat.duration_sec if stat.duration_sec else 0
         bns = stat.byte_count /
stat.duration_nsec if stat.duration_nsec else 0

file.write(f"{timestamp},{ev.msg.datapath.id},{flow_id},{i
p_src},{tp_src},{ip_dst},{tp_dst},{ip_proto},{icmp_code},{
icmp_type},{stat.duration_sec},{stat.duration_nsec},{stat.
idle_timeout},{stat.hard_timeout},{stat.flags},{stat.packe
t_count},{stat.byte_count},{pps},{pns},{bps},{bns},1\n")

```

3. Pemilihan Hyperparameter Model

```

from sklearn.neural_network import MLPClassifier
from sklearn.model_selection import RandomizedSearchCV
from scipy.stats import uniform

# Tentukan parameter yang ingin Anda cari
param_distributions = {
    'hidden_layer_sizes': [(50, 50), (100, 50), (64, 128), (256,
128)],
    'activation': ['tanh', 'relu'],
    'solver': ['sgd', 'adam'],
    'alpha': uniform(0.0001, 0.001), # Regularization term
    'learning_rate_init': uniform(0.001, 0.01) # Initial
learning rate
}

# Buat MLP Classifier
mlp = MLPClassifier(max_iter=100, early_stopping=True,
random_state=42)

# Buat instance RandomizedSearchCV

```

```

random_search = RandomizedSearchCV(mlp, param_distributions,
n_iter=100, cv=3, verbose=2, random_state=42, n_jobs=-1)

# Lakukan pencarian pada data pelatihan
random_search.fit(X_flow_train, y_flow_train)

# Hasil terbaik
print("Best parameters found: ", random_search.best_params_)
print("Best          cross-validation          score:
{: .2f}".format(random_search.best_score_))

```

4. Pelatihan Model

```

import pandas as pd
import os
import joblib
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelEncoder
import tensorflow as tf
from sklearn.neural_network import MLPClassifier
from keras.models import load_model
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score

flow_dataset = pd.read_csv('dataset.csv')
flow_dataset.head()

flow_dataset.iloc[:, 2] = flow_dataset.iloc[:,
2].str.replace('.', '')
flow_dataset.iloc[:, 3] = flow_dataset.iloc[:,
3].str.replace('.', '')
flow_dataset.iloc[:, 5] = flow_dataset.iloc[:,
5].str.replace('.', '')

X_flow = flow_dataset.iloc[:, :-1].values
X_flow = X_flow.astype('float64')

y_flow = flow_dataset.iloc[:, -1].values

X_flow_train, X_flow_test, y_flow_train, y_flow_test =
train_test_split(X_flow, y_flow, test_size=0.25, random_state=0)

# Membuat dan melatih model MLP yang ditingkatkan
classifier = MLPClassifier(
    hidden_layer_sizes=(256, 128), # Ubah ukuran lapisan
    tersembunyi
    activation='relu',
    alpha=0.001, # Menambahkan regularisasi L2
    random_state=42,
    solver='adam',
    learning_rate_init=0.001,
    verbose=1,
    early_stopping=True, # Menggunakan early stopping
    validation_fraction=0.1 # Porsi data validasi
)

```

```

#Training Model
flow_model = classifier.fit(X_flow_train, y_flow_train)

y_flow_pred = flow_model.predict(X_flow_test)

# Save the scaler
joblib.dump(classifier, 'mlp_model.pkl')

```

5. Uji Coba

```

from ryu.controller import ofp_event
from ryu.controller.handler import MAIN_DISPATCHER,
DEAD_DISPATCHER
from ryu.controller.handler import set_ev_cls
from ryu.lib import hub

import switch
from datetime import datetime

import os
import joblib
from sklearn.neural_network import MLPClassifier

import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, accuracy_score

class MonitoringApp(switch.SimpleSwitch13):

    def __init__(self, *args, **kwargs):
        super(MonitoringApp, self).__init__(*args, **kwargs)
        self.datapaths = {}
        self.monitor_thread = hub.spawn(self._monitor)

        start_time = datetime.now()
        self.train_flow_model()
        end_time = datetime.now()

        print("Training duration: ", (end_time - start_time))

        @set_ev_cls(ofp_event.EventOFPPStateChange, [MAIN_DISPATCHER,
DEAD_DISPATCHER])
        def handle_state_change(self, ev):
            datapath = ev.datapath
            if ev.state == MAIN_DISPATCHER:
                if datapath.id not in self.datapaths:
                    self.logger.debug('Registering datapath: %016x',
datapath.id)
                    self.datapaths[datapath.id] = datapath
            elif ev.state == DEAD_DISPATCHER:
                if datapath.id in self.datapaths:
                    self.logger.debug('Unregistering datapath:
%016x', datapath.id)
                    del self.datapaths[datapath.id]

    def _monitor(self):
        while True:
            for dp in self.datapaths.values():

```



```

        self.request_stats(dp)
        hub.sleep(10)
        self.predict_traffic()

    def request_stats(self, datapath):
        self.logger.debug('Sending stats request: %016x',
datapath.id)
        parser = datapath.ofproto_parser
        req = parser.OFPFlowStatsRequest(datapath)
        datapath.send_msg(req)

    @set_ev_cls(ofp_event.EventOFPFlowStatsReply,
MAIN_DISPATCHER)
    def handle_flow_stats_reply(self, ev):
        timestamp = datetime.now().timestamp()

        with open("PredictFlowStatsfile.csv", "w") as file:

file.write('timestamp,datapath_id,flow_id,ip_src,tp_src,ip_dst,t
p_dst,ip_proto,icmp_code,icmp_type,flow_duration_sec,flow_durati
on_nsec,idle_timeout,hard_timeout,flags,packet_count,byte_count,
packet_count_per_second,packet_count_per_nsecond,byte_count_per
second,byte_count_per_nsecond\n')

            body = ev.msg.body
            for stat in sorted([flow for flow in body if
flow.priority == 1], key=lambda flow: (flow.match['eth_type'],
flow.match['ipv4_src'], flow.match['ipv4_dst'],
flow.match['ip_proto'])):
                ip_src = stat.match['ipv4_src']
                ip_dst = stat.match['ipv4_dst']
                ip_proto = stat.match['ip_proto']
                icmp_code = stat.match.get('icmpv4_code', -1)
                icmp_type = stat.match.get('icmpv4_type', -1)
                tp_src = stat.match.get('tcp_src', 0) if ip_proto
== 6 else stat.match.get('udp_src', 0)
                tp_dst = stat.match.get('tcp_dst', 0) if ip_proto
== 6 else stat.match.get('udp_dst', 0)

                flow_id =
f"{ip_src}{tp_src}{ip_dst}{tp_dst}{ip_proto}"
                packet_count_per_second = stat.packet_count /
stat.duration_sec if stat.duration_sec else 0
                packet_count_per_nsecond = stat.packet_count /
stat.duration_nsec if stat.duration_nsec else 0
                byte_count_per_second = stat.byte_count /
stat.duration_sec if stat.duration_sec else 0
                byte_count_per_nsecond = stat.byte_count /
stat.duration_nsec if stat.duration_nsec else 0

file.write(f"{timestamp},{ev.msg.datapath.id},{flow_id},{ip_src}
,{tp_src},{ip_dst},{tp_dst},{ip_proto},{icmp_code},{icmp_type},{
stat.duration_sec},{stat.duration_nsec},{stat.idle_timeout},{sta
t.hard_timeout},{stat.flags},{stat.packet_count},{stat.byte_coun
t},{packet_count_per_second},{packet_count_per_nsecond},{byte_co
unt_per_second},{byte_count_per_nsecond}\n")

    def train_flow_model(self):

```

```

self.logger.info("Starting flow model training...")

if os.path.isfile('mlp_model.pkl'):
    self.flow_model = joblib.load('mlp_model.pkl')
else:
    dataset = pd.read_csv('dataset.csv')
    dataset.iloc[:, 2] = dataset.iloc[:,
2].str.replace('.', '')
    dataset.iloc[:, 3] = dataset.iloc[:,
3].str.replace('.', '')
    dataset.iloc[:, 5] = dataset.iloc[:,
5].str.replace('.', '')

    X = dataset.iloc[:, :-1].values.astype('float64')
    y = dataset.iloc[:, -1].values

    X_train, X_test, y_train, y_test =
train_test_split(X, y, test_size=0.25, random_state=0)

    classifier = MLPClassifier(
        hidden_layer_sizes=(256, 128),
        activation='relu',
        alpha=0.001,
        random_state=42,
        solver='adam',
        learning_rate_init=0.001,
        verbose=1,
        early_stopping=True,
        validation_fraction=0.1
    )

    self.flow_model = classifier.fit(X_train, y_train)
    joblib.dump(self.flow_model, 'mlp_model.pkl')

    y_pred = self.flow_model.predict(X_test)

    self.logger.info("Confusion Matrix:")
    self.logger.info(confusion_matrix(y_test, y_pred))

    accuracy = accuracy_score(y_test, y_pred)
    self.logger.info(f"Success accuracy: {accuracy *
100:.2f}%")
    self.logger.info(f"Fail accuracy: {(1 - accuracy) *
100:.2f}%")

    def predict_traffic(self):
        try:
            predict_flow_dataset =
pd.read_csv('PredictFlowStatsfile.csv')

            predict_flow_dataset.iloc[:, 2] =
predict_flow_dataset.iloc[:, 2].str.replace('.', '')
            predict_flow_dataset.iloc[:, 3] =
predict_flow_dataset.iloc[:, 3].str.replace('.', '')
            predict_flow_dataset.iloc[:, 5] =
predict_flow_dataset.iloc[:, 5].str.replace('.', '')

            X_predict_flow =
predict_flow_dataset.values.astype('float64')

```

```

        y_pred = self.flow_model.predict(X_predict_flow)

        legitimate_traffic = sum(y_pred == 0)
        ddos_traffic = sum(y_pred != 0)

        self.logger.info("Traffic Analysis:")
        self.logger.info(f"Legitimate           Traffic:
{legitimate_traffic}")
        self.logger.info(f"DDoS Traffic: {ddos_traffic}")
        self.logger.info(f"Total           Traffic           analyzed:
{len(y_pred)}")

        if (legitimate_traffic / len(y_pred)) * 100 > 80:
            self.logger.info("Traffic is mostly normal.")
        else:
            victim = int(predict_flow_dataset.iloc[y_pred !=
0, 5].iloc[0]) % 20
            self.logger.info("DDoS traffic detected.")
            self.logger.info(f"Victim Host: h{victim}")
            print("Mitigation process in progress!")
            self.mitigation = 1

        with open("PredictFlowStatsfile.csv", "w") as file:
            file.write('timestamp,datapath_id,flow_id,ip_src,tp_src,ip_dst,t
p_dst,ip_proto,icmp_code,icmp_type,flow_duration_sec,flow_durati
on_nsec,idle_timeout,hard_timeout,flags,packet_count,byte_count,
packet_count_per_second,packet_count_per_nsecond,byte_count_per_
second,byte_count_per_nsecond\n')

        except Exception as e:
            self.logger.error(f"Error in predicting traffic:
{e}")

```