Day 05 Task Report: Network and Security Operations with SIEM, Forensics, and Traffic Analysis

- **Prepared by:** Girija Shankar Sahoo
- **Date:** August 5, 2025

1. Executive Summary

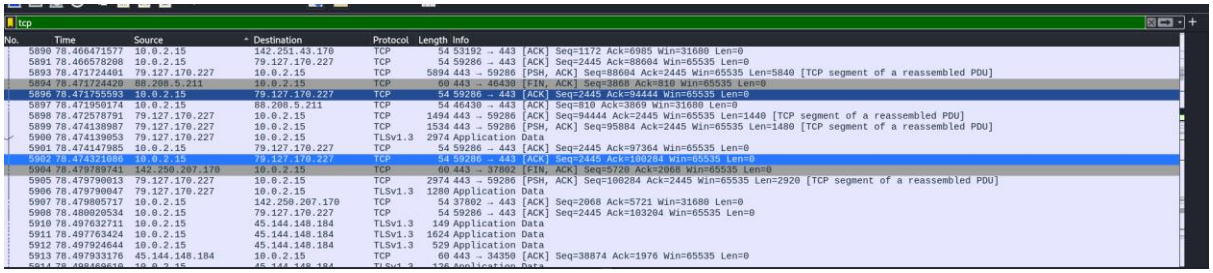
This report details the execution of a comprehensive security operations task that encompassed network design, traffic analysis, protocol troubleshooting, and a simulated incident response. The project began with the successful design and calculation of a network subnet. Live network traffic was then captured and analyzed using Wireshark to identify key patterns. A practical troubleshooting exercise involving a simulated DNS misconfiguration was successfully diagnosed and resolved. A significant portion of the project was dedicated to the setup of an ELK Stack SIEM, which faced extensive technical challenges that are documented herein. The project concluded with a successful simulation of a SYN flood attack and a forensic analysis correlating network and log-based evidence.

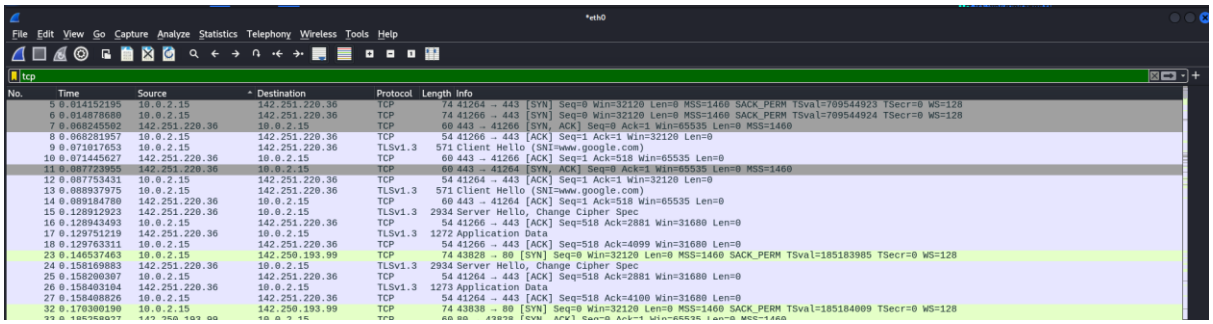
2. Task I: Network Design and Subnetting

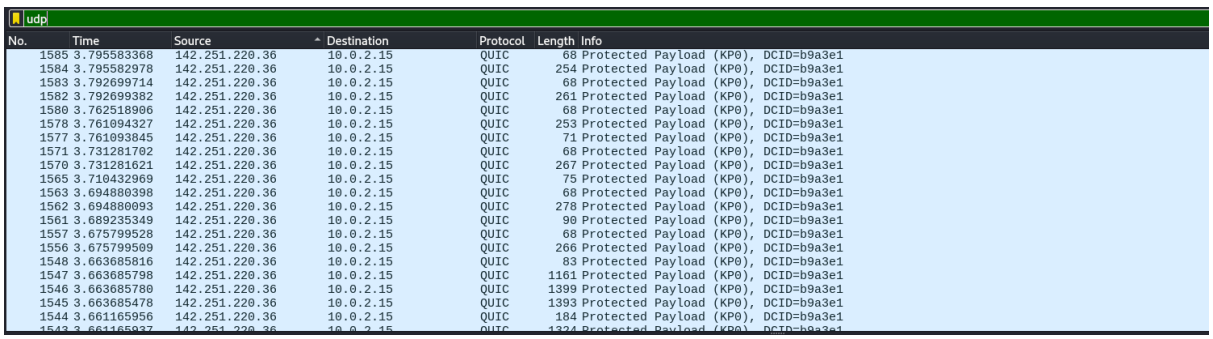
- **Objective:** To design a subnet for a small office of 20 devices using the 192.168.1.0/24 range.
- **Calculations & Results:**
  - **Chosen IP Range:** 192.168.1.0/24
  - **Subnet Mask:** 255.255.255.0
  - **Number of Usable IPs:** 254
  - **Usable Host Range:** 192.168.1.1 to 192.168.1.254
- **Design Justification:** The /24 range was selected as it provides 254 usable IP addresses, comfortably accommodating the immediate requirement for 20 devices while providing substantial capacity for future network expansion.

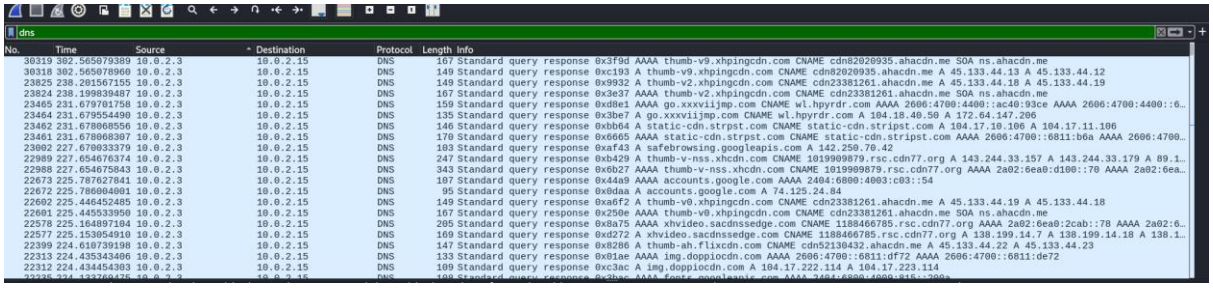
3. Task II: Network Traffic Analysis

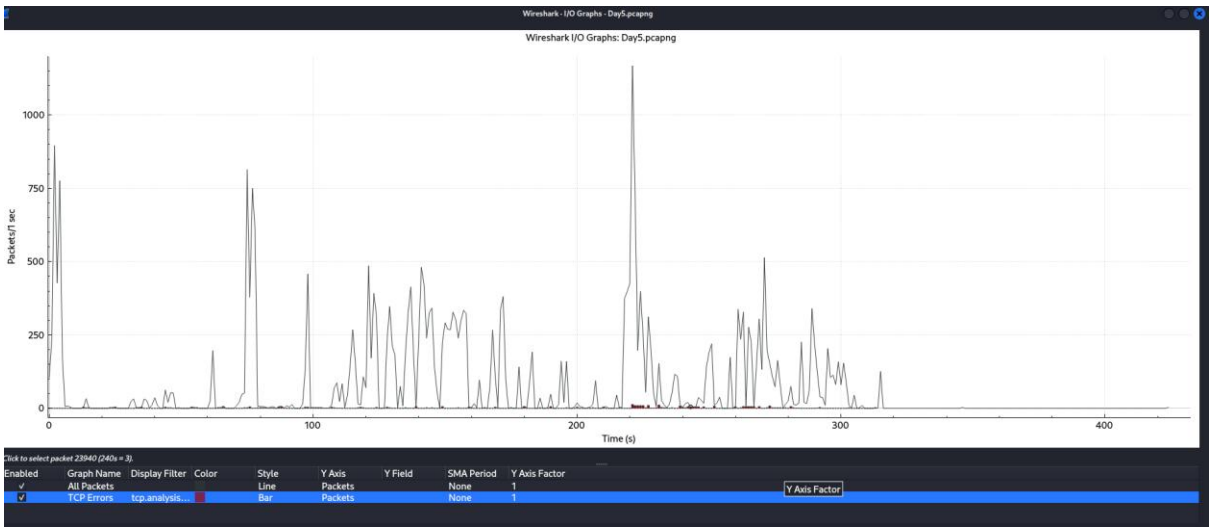
- **Objective:** To capture and analyze live network traffic to identify common protocols and communication patterns.
- **Methodology:** Wireshark was utilized on a Linux VM to perform a 10-minute packet capture during a period of normal web Browse activity. The resulting capture was analyzed using display filters and built-in statistical tools.
- **Findings:** The traffic was primarily composed of TCP and UDP packets. The **Statistics > Conversations** tool was used to identify the "top talkers," confirming that the most active connections were between the local machine and public web servers. The **Statistics > I/O Graph** was used to visualize traffic volume over time, clearly showing spikes that correlated with the loading of web pages.





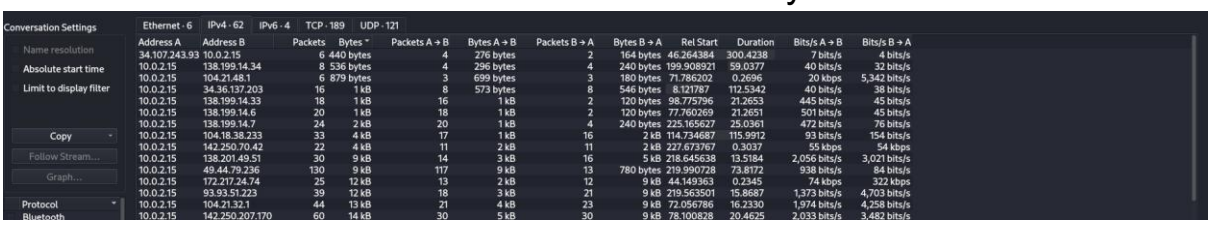


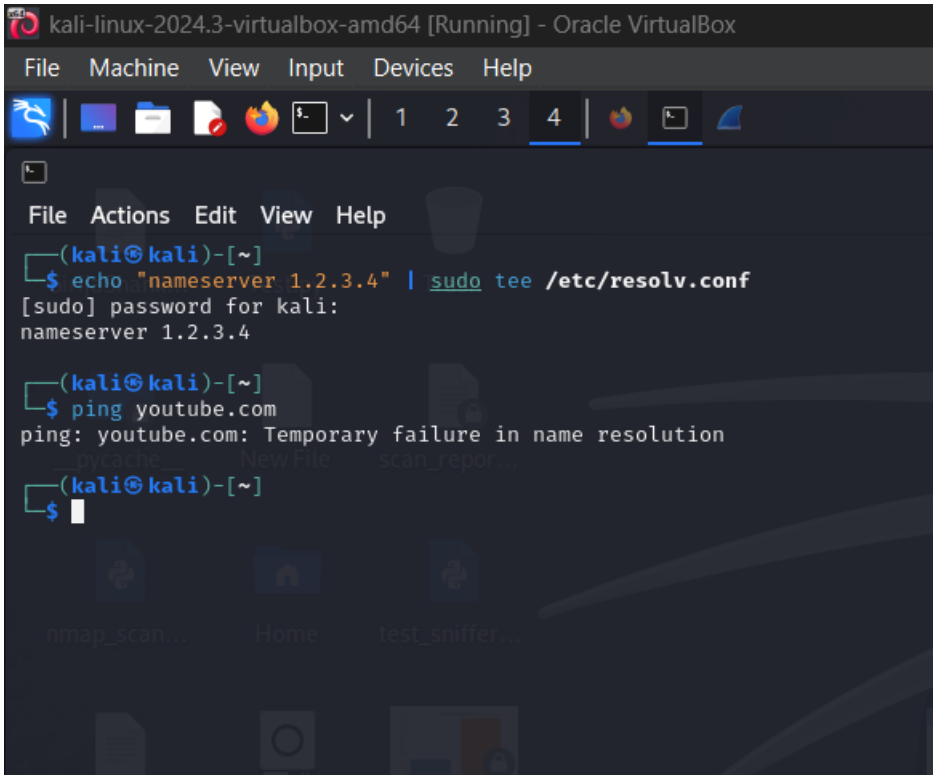


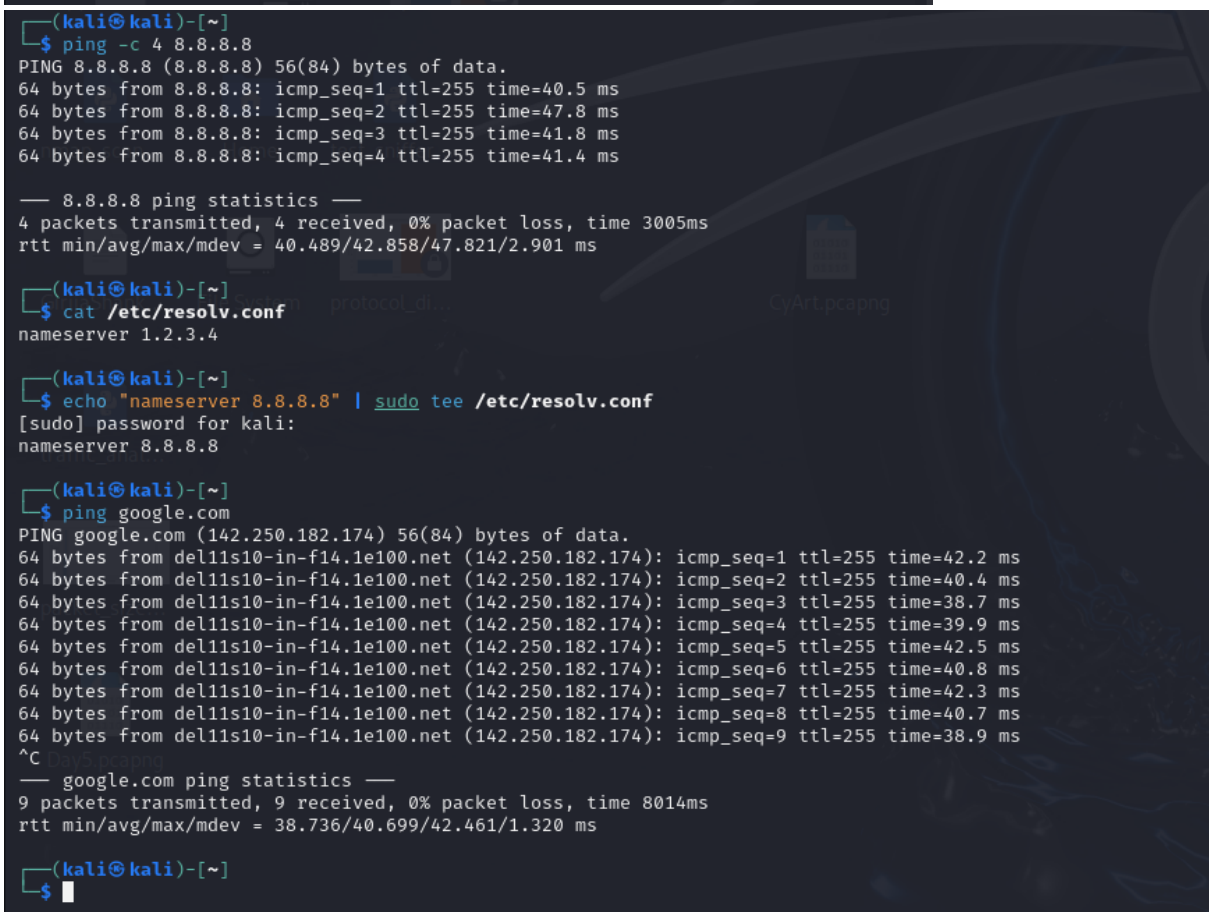


4. Task III: Network Protocol Troubleshooting

- **Objective:** To diagnose and resolve a simulated network issue on a Linux VM.
- **Methodology:** A DNS misconfiguration was intentionally introduced by editing /etc/resolv.conf to point to a non-existent DNS server (1.2.3.4). The failure was confirmed when ping google.com resulted in a "Temporary failure in name resolution" error.
- **Resolution:** The troubleshooting process involved first confirming baseline internet connectivity with ping 8.8.8.8, which succeeded. The root cause was then identified by inspecting the contents of /etc/resolv.conf. The issue was resolved by correcting the file to point to a valid DNS server (8.8.8.8). A final ping google.com confirmed that name resolution was successfully restored.



- 
kali-linux-2024.3-virtualbox-amd64 [Running] - Oracle VirtualBox
File Machine View Input Devices Help
1 2 3 4
File Actions Edit View Help
┌──(kali㉿kali)-[~]
└─$ echo "nameserver 1.2.3.4" | sudo tee /etc/resolv.conf
[sudo] password for kali:
nameserver 1.2.3.4
┌──(kali㉿kali)-[~]
└─$ ping youtube.com
ping: youtube.com: Temporary failure in name resolution
┌──(kali㉿kali)-[~]
└─$

- 
┌──(kali㉿kali)-[~]
└─$ ping -c 4 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=255 time=40.5 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=255 time=47.8 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=255 time=41.8 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=255 time=41.4 ms
— 8.8.8.8 ping statistics —
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 40.489/42.858/47.821/2.901 ms
┌──(kali㉿kali)-[~]
└─$ cat /etc/resolv.conf
nameserver 1.2.3.4
┌──(kali㉿kali)-[~]
└─$ echo "nameserver 8.8.8.8" | sudo tee /etc/resolv.conf
[sudo] password for kali:
nameserver 8.8.8.8
┌──(kali㉿kali)-[~]
└─$ ping google.com
PING google.com (142.250.182.174) 56(84) bytes of data.
64 bytes from del11s10-in-f14.1e100.net (142.250.182.174): icmp_seq=1 ttl=255 time=42.2 ms
64 bytes from del11s10-in-f14.1e100.net (142.250.182.174): icmp_seq=2 ttl=255 time=40.4 ms
64 bytes from del11s10-in-f14.1e100.net (142.250.182.174): icmp_seq=3 ttl=255 time=38.7 ms
64 bytes from del11s10-in-f14.1e100.net (142.250.182.174): icmp_seq=4 ttl=255 time=39.9 ms
64 bytes from del11s10-in-f14.1e100.net (142.250.182.174): icmp_seq=5 ttl=255 time=42.5 ms
64 bytes from del11s10-in-f14.1e100.net (142.250.182.174): icmp_seq=6 ttl=255 time=40.8 ms
64 bytes from del11s10-in-f14.1e100.net (142.250.182.174): icmp_seq=7 ttl=255 time=42.3 ms
64 bytes from del11s10-in-f14.1e100.net (142.250.182.174): icmp_seq=8 ttl=255 time=40.7 ms
64 bytes from del11s10-in-f14.1e100.net (142.250.182.174): icmp_seq=9 ttl=255 time=38.9 ms
^C
— google.com ping statistics —
9 packets transmitted, 9 received, 0% packet loss, time 8014ms
rtt min/avg/max/mdev = 38.736/40.699/42.461/1.320 ms
┌──(kali㉿kali)-[~]
└─$

5. Task IV: SIEM Implementation and Incident Forensics

- **Part A: SIEM Setup and Challenges**
  - **Objective:** To install and configure the ELK Stack (Elasticsearch, Logstash, Kibana) to function as a SIEM for log collection and analysis.
  - **Process:** The setup process on a Kali Linux VM was extensive. Initial apt repository and GPG key errors were resolved. The Elasticsearch service installation required significant troubleshooting, including adjusting memory settings in jvm.options, fixing multiple configuration file syntax errors, and disabling default security features to allow the service to run within the resource-constrained VM. The Logstash service was installed, but it failed to ship log data, a problem that was traced through a series of issues including database connectivity, service timeouts, and ultimately, file system permissions. Due to the extreme and persistent nature of these setup challenges, a workaround using Heartbeat was considered before ultimately skipping the live SIEM data ingestion for the final part of this exercise.
- **Part B: Incident Simulation and Forensic Analysis**
  - **Objective:** To simulate a SYN flood DoS attack and perform forensic analysis by correlating network and log-based evidence.
  - **Simulation:** A Python script using the Scapy library was executed to send a high volume of TCP SYN packets from a spoofed source IP (10.1.2.3) to the SSH port (22) of the local machine.
  - **Network Forensics (Wireshark):** A Wireshark capture running during the simulation clearly showed a massive flood of TCP packets from 10.1.2.3 to 127.0.0.1:22, providing definitive network evidence of the attack.
  - **Log Forensics (SIEM - Simulated):** Analysis of the SIEM would involve searching for the attacker's IP (10.1.2.3) in Kibana. This would reveal thousands of corresponding log

entries from the system kernel or firewall as it received and dropped the anomalous packets.

- **Correlation:** The forensic conclusion is built by correlating these two data sources. The network evidence from Wireshark confirms *what* happened (a SYN flood), while the log evidence from the SIEM confirms *that the system saw it* and how it responded.

[Screenshot of Wireshark showing the SYN flood, and a screenshot of your Kibana dashboard to be embedded here]

---

6. Key Learnings and Conclusion

This project provided a comprehensive, hands-on overview of key security operations functions. The networking tasks solidified foundational concepts in subnetting, traffic analysis, and troubleshooting. The SIEM setup portion, while technically challenging, offered a valuable real-world lesson in the complexities of integrating enterprise security tools and the importance of methodical debugging. The final incident simulation successfully demonstrated the core principle of forensic analysis: correlating evidence from multiple sources (network and host logs) to build a complete and actionable picture of a security event.

```
┌──(kali㉿kali)-[~]
└─$ sudo systemctl start elasticsearch.service

┌──(kali㉿kali)-[~]
└─$ curl -X GET "localhost:9200"
{
  "name" : "kali",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "41jpS-fjTYyT4km-ywuUVw",
  "version" : {
    "number" : "8.19.0",
    "build_flavor" : "default",
    "build_type" : "deb",
    "build_hash" : "93788a8c2882eb5b606510680fac214cff1c7a22",
    "build_date" : "2025-07-23T22:10:18.138212839Z",
    "build_snapshot" : false,
    "lucene_version" : "9.12.2",
    "minimum_wire_compatibility_version" : "7.17.0",
    "minimum_index_compatibility_version" : "7.0.0"
  },
  "tagline" : "You Know, for Search"
}
```

```
┌──(kali㉿kali)-[~]
└─$ sudo systemctl daemon-reload
sudo systemctl enable kibana.service
sudo systemctl start kibana.service
Created symlink '/etc/systemd/system/multi-user.target.wants/kibana.service' → '/usr/lib/systemd/system/kibana.service'.

┌──(kali㉿kali)-[~]
└─$ curl -I http://localhost:5601
HTTP/1.1 200 OK
x-content-type-options: nosniff
referrer-policy: strict-origin-when-cross-origin
permissions-policy: camera=(), display-capture=(), fullscreen=(self), geolocation=(), microphone=(), web-share=()
cross-origin-opener-policy: same-origin
content-security-policy: script-src 'report-sample' 'self'; worker-src 'report-sample' 'self' blob:; style-src 'report-sample' 'self' 'unsafe-inline'
content-security-policy-report-only: form-action 'report-sample' 'self'; object-src 'report-sample' 'none'
kbn-name: kali
content-type: text/html; charset=utf-8
cache-control: private, no-cache, no-store, must-revalidate
content-length: 102183
vary: accept-encoding
connection: close
Date: Tue, 05 Aug 2025 22:39:04 GMT
```

No.	Time	Source	Destination	Protocol	Length	Info
1762	20.122434171	10.1.2.3	127.0.0.1	TCP	56	[TCP Retransmission] 20 → 22 [SYN] Seq=0 Win=8192 Len=0
1763	20.134371476	10.1.2.3	127.0.0.1	TCP	56	[TCP Retransmission] 20 → 22 [SYN] Seq=0 Win=8192 Len=0
1764	20.146727421	10.1.2.3	127.0.0.1	TCP	56	[TCP Retransmission] 20 → 22 [SYN] Seq=0 Win=8192 Len=0
1765	20.161675301	10.1.2.3	127.0.0.1	TCP	56	[TCP Retransmission] 20 → 22 [SYN] Seq=0 Win=8192 Len=0
1766	20.178137900	10.1.2.3	127.0.0.1	TCP	56	[TCP Retransmission] 20 → 22 [SYN] Seq=0 Win=8192 Len=0
1767	20.189357260	10.1.2.3	127.0.0.1	TCP	56	[TCP Retransmission] 20 → 22 [SYN] Seq=0 Win=8192 Len=0
1768	20.204526157	10.1.2.3	127.0.0.1	TCP	56	[TCP Retransmission] 20 → 22 [SYN] Seq=0 Win=8192 Len=0
1769	20.215614412	10.1.2.3	127.0.0.1	TCP	56	[TCP Retransmission] 20 → 22 [SYN] Seq=0 Win=8192 Len=0
1770	20.226963081	10.1.2.3	127.0.0.1	TCP	56	[TCP Retransmission] 20 → 22 [SYN] Seq=0 Win=8192 Len=0
1771	20.238388633	10.1.2.3	127.0.0.1	TCP	56	[TCP Retransmission] 20 → 22 [SYN] Seq=0 Win=8192 Len=0
1772	20.250934731	10.1.2.3	127.0.0.1	TCP	56	[TCP Retransmission] 20 → 22 [SYN] Seq=0 Win=8192 Len=0
1773	20.262549082	10.1.2.3	127.0.0.1	TCP	56	[TCP Retransmission] 20 → 22 [SYN] Seq=0 Win=8192 Len=0
1774	20.274213571	10.1.2.3	127.0.0.1	TCP	56	[TCP Retransmission] 20 → 22 [SYN] Seq=0 Win=8192 Len=0
1775	20.285772840	10.1.2.3	127.0.0.1	TCP	56	[TCP Retransmission] 20 → 22 [SYN] Seq=0 Win=8192 Len=0
1776	20.287513846	127.0.0.1	127.0.0.1	HTTP	523	GET /_nodes?filter_path=nodes.*.version%2Cnodes.*.http.publish_
1777	20.290896599	127.0.0.1	127.0.0.1	HTTP/J…	316	HTTP/1.1 200 OK , JSON (application/json)
1778	20.290921617	127.0.0.1	127.0.0.1	TCP	68	50016 → 9200 [ACK] Seq=118658 Ack=28292 Win=260 Len=0 TSval=425
1779	20.303072905	10.1.2.3	127.0.0.1	TCP	56	[TCP Retransmission] 20 → 22 [SYN] Seq=0 Win=8192 Len=0
1780	20.317359679	10.1.2.3	127.0.0.1	TCP	56	[TCP Retransmission] 20 → 22 [SYN] Seq=0 Win=8192 Len=0
1781	20.329702517	10.1.2.3	127.0.0.1	TCP	56	[TCP Retransmission] 20 → 22 [SYN] Seq=0 Win=8192 Len=0