Project Title: Network Path Analysis Using `traceroute` Utility

Objective

To utilize the `traceroute` tool for visualizing and analyzing the network path taken by packets to reach a destination. This includes understanding routing behavior, identifying latency at each hop, and detecting potential network/firewall filtering behaviors. The project simulates real-world diagnostic workflows used by IT support, network engineers, and SOC teams.

Tools Used

- **Operating System**: Kali Linux (or any Debian-based distro)
- **Primary Utility**: `traceroute`
- **Network**: Internet-connected system with multiple hops
- **Firewall/NAT Simulation**: Optional for advanced analysis

Skills Learned

- Understanding network hops and router traversal
- Identifying latency issues and bottlenecks across routes
- Using different protocols (UDP, TCP) for trace visibility
- Detecting firewall and NAT filtering behaviors
- Adjusting hop limits for controlled path exploration
- Developing intuition around network topology

Test Scenarios Covered

Level	Description	Command Example
🟢 Basic	Run a default traceroute to a public site like Google and observe the number of hops	`traceroute google.com`
🟡 Intermediate	Limit the max hops to observe truncated path output	`traceroute -m 5 google.com`
🔴 Advanced	Perform UDP-based traceroute (default) and analyze missing hops due to firewalls/NATs	`traceroute -U server.com`
🔴 Advanced	Perform TCP-based traceroute to bypass ICMP/UDP restrictions and get cleaner hop data	`traceroute -T google.com`

Real-World Impact & Use Case

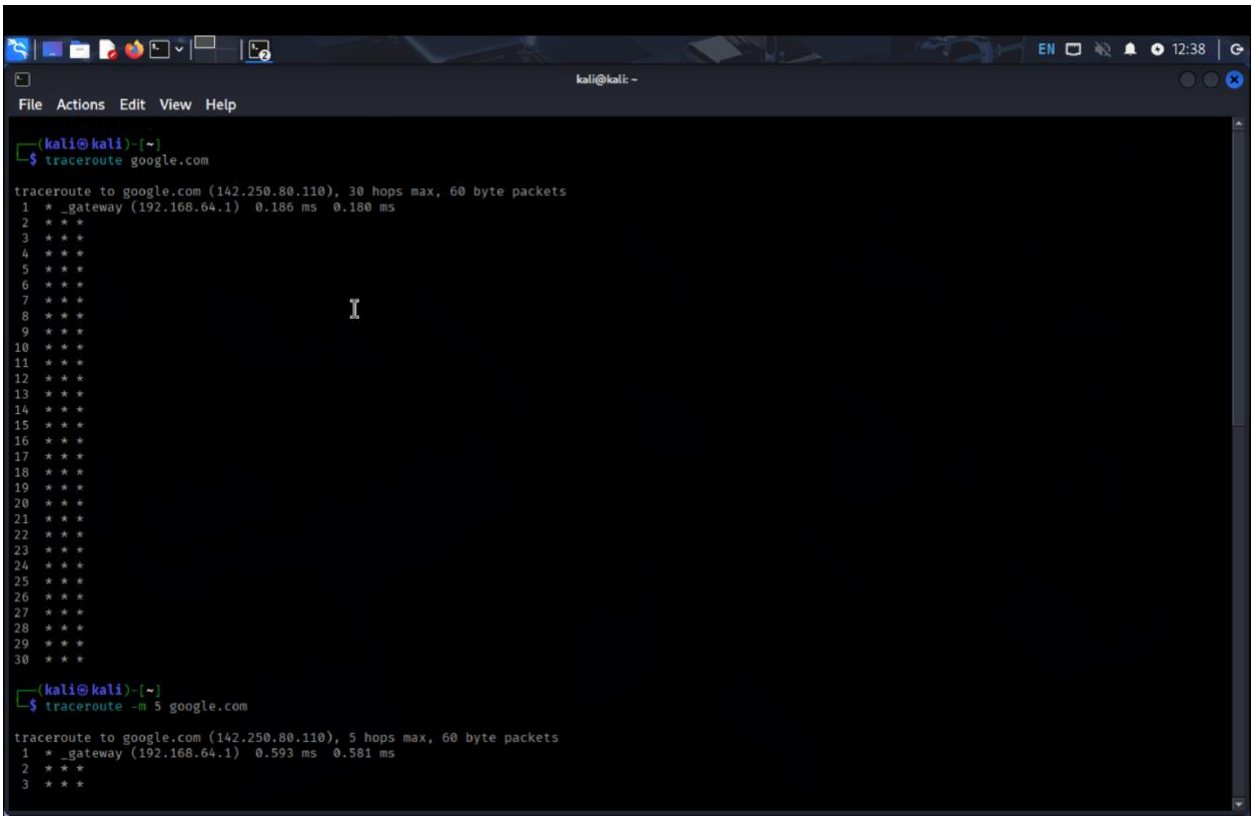
Mastering `traceroute` allows professionals to:

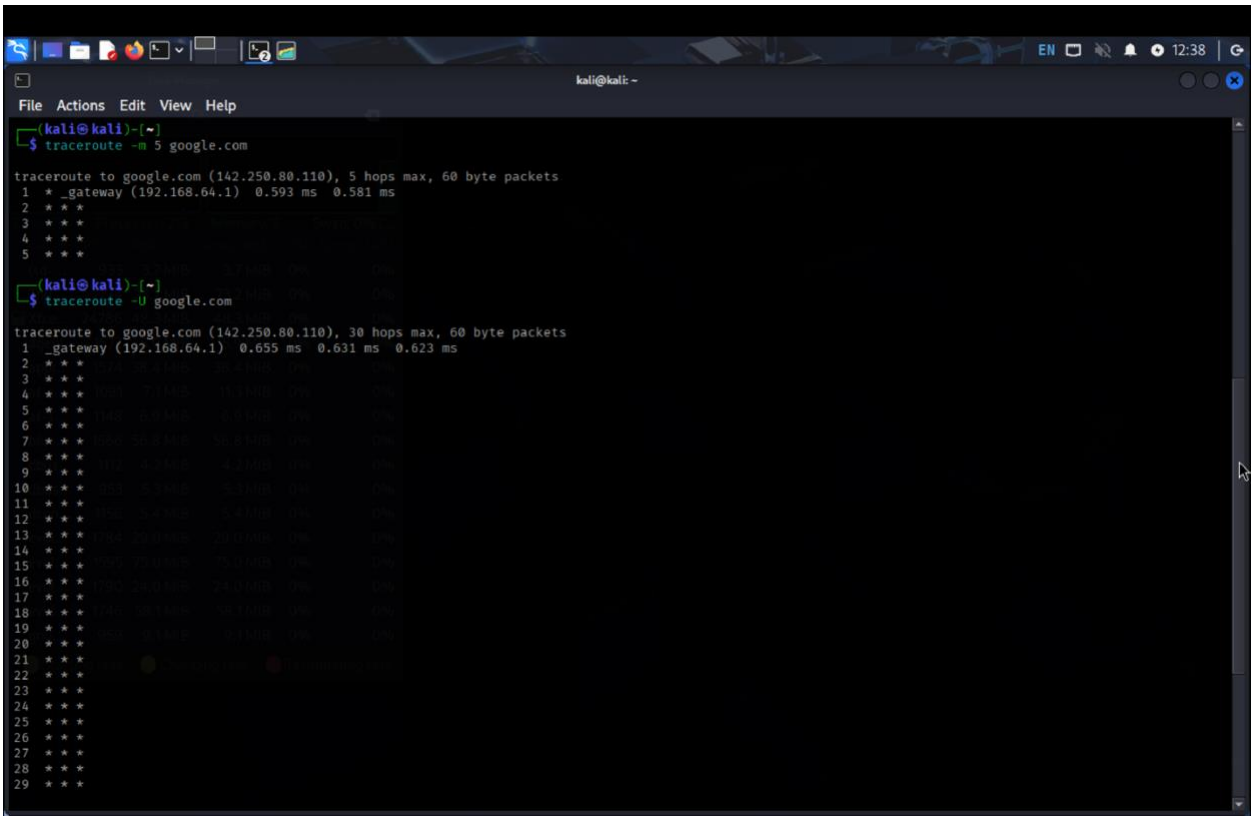
- Diagnose path delays and degraded performance across networks
- Identify where packets are being filtered or dropped (common in SOC analysis)
- Troubleshoot VPN or remote access latency issues
- Audit ISP or cloud routing behavior

- Detect man-in-the-middle or anomalous routing paths (threat intel use case)

Tip

- In SOC and cybersecurity settings, `traceroute` is often used in combination with tools like `ping`, `mtr`, and packet sniffers (`tcpdump`) to **triangulate network anomalies**, detect **malicious rerouting**, or confirm **network segmentation policies**.




EN
12:38
kali@kali: ~
File Actions Edit View Help
┌──(kali㉿kali)-[~]
└─$ traceroute -m 5 google.com
traceroute to google.com (142.250.80.110), 5 hops max, 60 byte packets
1 * _gateway (192.168.64.1) 0.593 ms 0.581 ms
2 * * *
3 * * *
4 * * *
5 * * *
┌──(kali㉿kali)-[~]
└─$ traceroute -U google.com
traceroute to google.com (142.250.80.110), 30 hops max, 60 byte packets
1 _gateway (192.168.64.1) 0.655 ms 0.631 ms 0.623 ms
2 * * *
3 * * *
4 * * *
5 * * *
6 * * *
7 * * *
8 * * *
9 * * *
10 * * *
11 * * *
12 * * *
13 * * *
14 * * *
15 * * *
16 * * *
17 * * *
18 * * *
19 * * *
20 * * *
21 * * *
22 * * *
23 * * *
24 * * *
25 * * *
26 * * *
27 * * *
28 * * *
29 * * *

```
kali@kali: ~
File  Actions  Edit  View  Help
 5  * * *
 6  * * *
 7  * * *
 8  * * *
 9  * * *
10  * * *
11  * * *
12  * * *
13  * * *
14  * * *
15  * * *
16  * * *
17  * * *
18  * * *
19  * * *
20  * * *
21  * * *
22  * * *
23  * * *
24  * * *
25  * * *
26  * * *
27  * * *
28  * * *
29  * * *
30  * * *

┌──(kali㉿kali)-[~]
└─$ traceroute -T google.com

You do not have enough privileges to use this traceroute method.
socket: Operation not permitted

┌──(kali㉿kali)-[~]
└─$ sudo traceroute -T google.com

[sudo] password for kali:
traceroute to google.com (142.250.80.110), 30 hops max, 60 byte packets
 1  * * *
 2  lga34s36-in-f14.1e100.net (142.250.80.110)  1.875 ms  1.871 ms  1.858 ms

┌──(kali㉿kali)-[~]
└─$
```