

Wireshark Project Report — TCP 3-Way Handshake & Stealth Scan Analysis

Author: Temitope C Folowosele

Role: Cybersecurity Analyst

Date: October 2025

Lab Host: kali (privileged)

Executive Summary

This report documents a Wireshark analysis performed to observe and validate a TCP 3-way handshake between a host (10.0.2.4) and a destination server (66.29.153.49) on destination port 80. The capture was filtered by TCP port and IP address to isolate the session. Additionally, the analysis covers stealth scanning techniques (stealth/SYN scans, decoy scans, time fragmentation scans), how they can bypass detection, and recommended detection & mitigation strategies.

Objectives

- Capture and identify the complete TCP 3-way handshake (SYN, SYN-ACK, ACK) between 10.0.2.4 and 66.29.153.49 on port 80.
- Demonstrate packet filtering in Wireshark by IP and TCP port (e.g., `ip.addr==10.0.2.4` `ip.addr==66.29.153.49 && tcp.port==80`).
- Explain stealth scan variants (SYN/half-open, RST) and how they evade detection (use of nonstandard flag combos, fragmented/timed probes).
- Provide practical detection and mitigation recommendations.

Environment & Capture Details

Environment: Local lab (VM host: Kali).

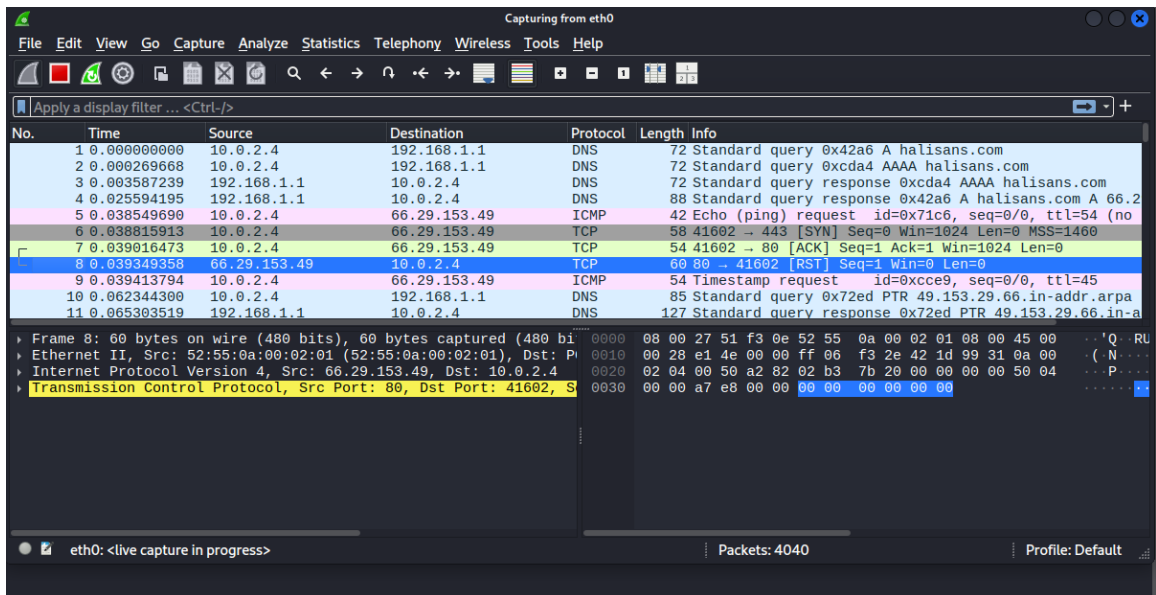
Wireshark version: (placeholder) capture saved as `capture_wireshark.pcapng`.

Host IP: 10.0.2.4

Destination IP: 66.29.153.49

Target service: HTTP (TCP port 80)

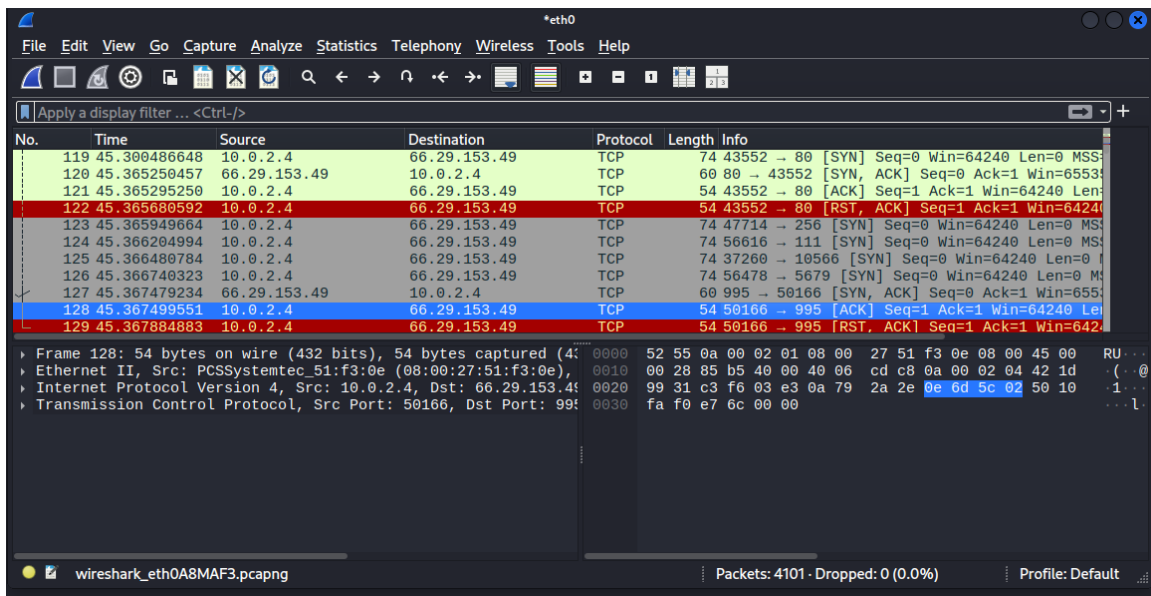
Capture method: Promiscuous mode on the host interface; capture start/end timestamps:
(placeholders).



Identifying the 3-Way Handshake

The TCP 3-way handshake can be identified by locating three packets in sequence:

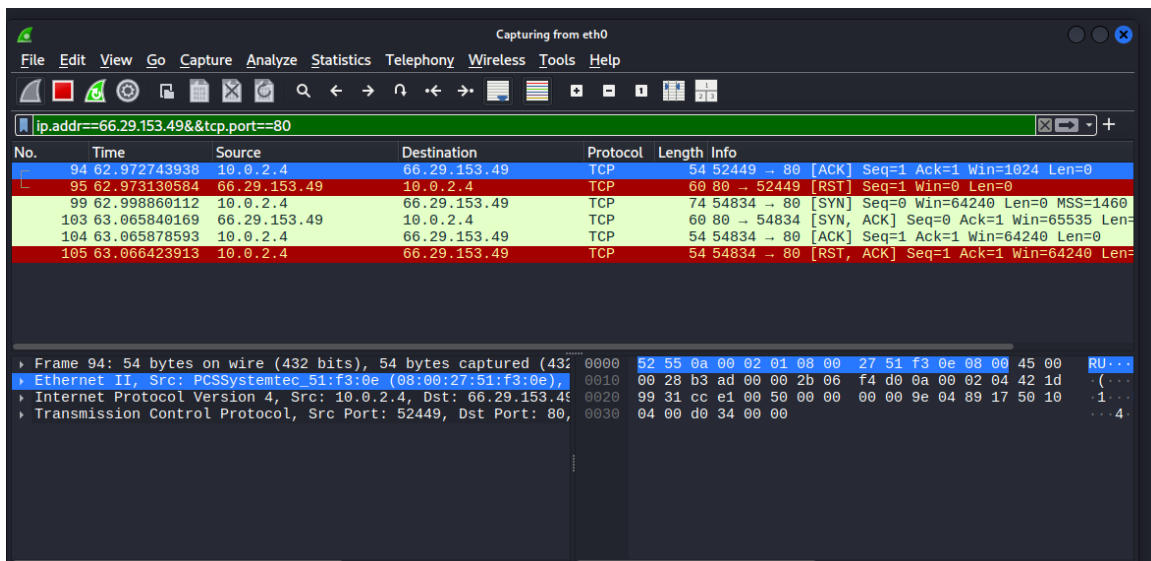
1. SYN — Client (10.0.2.4) sends TCP segment with SYN bit set (tcp.flags.syn==1, tcp.flags.ack==0).
2. SYN-ACK — Server (66.29.153.49) replies with SYN+ACK (tcp.flags.syn==1, tcp.flags.ack==1).
3. ACK — Client (10.0.2.3) sends ACK (tcp.flags.ack==1, tcp.flags.syn==0) to complete the handshake.



Port Scanning & Filters Used

Port scanning was performed targeting port 80 on 66.29.153.49. To isolate scan traffic, the following Wireshark filters were used:

```
# Filter by source host and port
ip.addr == 66.29.153.49&&tcp.port==80
```



No.	Time	Source	Destination	Protocol	Length	Info
94	62.972743938	10.0.2.4	66.29.153.49	TCP	54	52449 → 80 [ACK] Seq=1 Ack=1 Win=1024 Len=0
95	62.973130584	66.29.153.49	10.0.2.4	TCP	60	80 → 52449 [RST] Seq=1 Win=0 Len=0
99	62.998860112	10.0.2.4	66.29.153.49	TCP	74	54834 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460
103	63.065840169	66.29.153.49	10.0.2.4	TCP	60	80 → 54834 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0
104	63.065878593	10.0.2.4	66.29.153.49	TCP	54	54834 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
105	63.066423913	10.0.2.4	66.29.153.49	TCP	54	54834 → 80 [RST, ACK] Seq=1 Ack=1 Win=64240 Len=0

Frame 94: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface 0	0000	52 55 0a 00 02 01 08 00 27 51 f3 0e 08 00 45 00	RU...
Ethernet II, Src: PCSSystemtec_51:f3:0e (08:00:27:51:f3:0e),	0010	00 28 b3 ad 00 00 2b 06 f4 d0 0a 00 02 04 42 1d	(...
Internet Protocol Version 4, Src: 10.0.2.4, Dst: 66.29.153.49	0020	99 31 cc e1 00 50 00 00 00 00 9e 04 89 17 50 10	-1...
Transmission Control Protocol, Src Port: 52449, Dst Port: 80,	0030	04 00 d0 34 00 00	...4

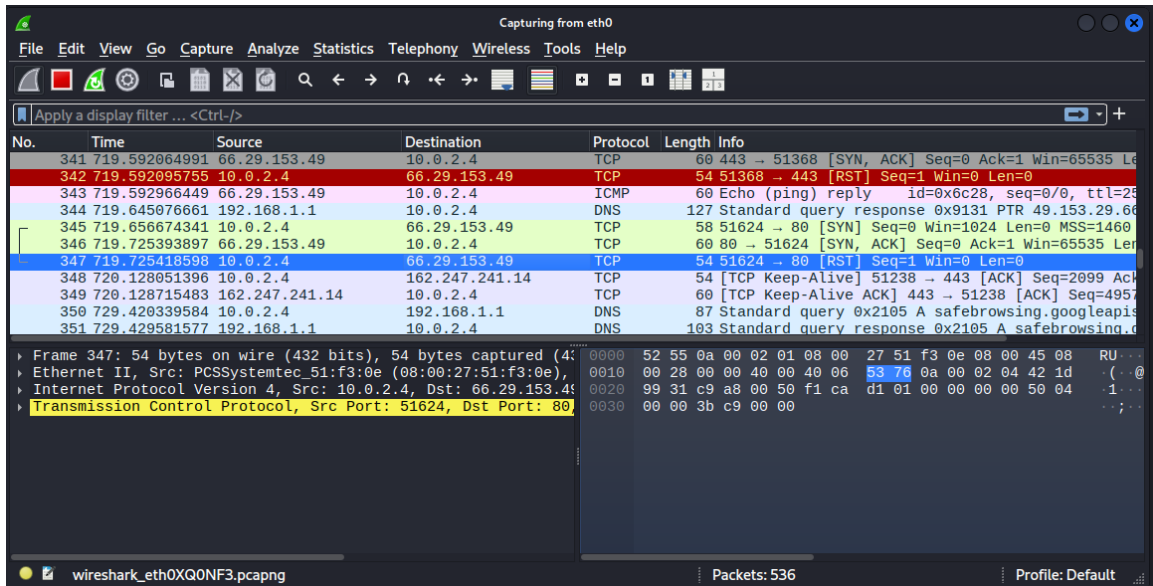
Evasion Techniques Observed / Discussed

The following stealth scan techniques were discussed. These techniques can make detection harder for simple signature-based IDS/IPS or manual review.

1) Stealth (SYN) Scans — Half-Open Scans

Behavior: The attacker sends a SYN and analyzes the response. If SYN-ACK is received the port is open; attacker sends RST instead of completing the handshake (no final ACK), avoiding full connection establishment.

Why it can bypass detection: Some naive detection rules look for completed handshakes or payloads; dropping the handshake completion can avoid certain logging. However, modern IDS/IPS and connection tracking usually detect large volumes of SYNs and incomplete handshakes (SYN floods or unusual SYN/SYN-ACK ratios).

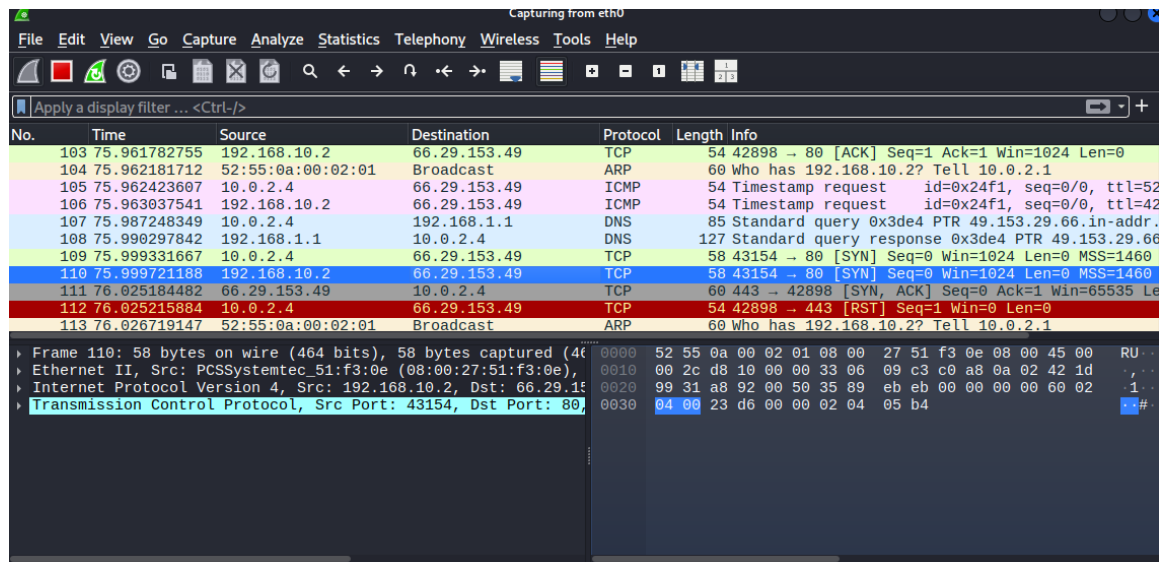


2) Decoy Scans

Behavior: The attacker uses multiple spoofed source IPs (decoys) along with the true source to mix legitimate-looking traffic with malicious probes. This blends probe traffic with benign-looking flows, making attribution and detection harder.

Why it can bypass detection: Alerts generated per-source may be diluted; threshold-based systems may not detect low-rate probes spread across many decoy IPs.

Detection strategies: Correlate destination-side logs, look for identical probe patterns (same TTL, window size, TCP options), and use anomaly detection across multiple sources.



The image shows a Wireshark packet capture window titled "Capturing from eth0". The interface includes a menu bar (File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, Help) and a toolbar with various icons. A display filter bar shows "Apply a display filter ... <Ctrl-/>". The packet list pane displays a table of captured packets with columns for No., Time, Source, Destination, Protocol, and Length. The packet details pane shows the selected packet (No. 110) with its structure: Ethernet II, Internet Protocol Version 4, and Transmission Control Protocol. The packet bytes pane shows the raw data in hexadecimal and ASCII.

No.	Time	Source	Destination	Protocol	Length	Info
103	75.961782755	192.168.10.2	66.29.153.49	TCP	54	42898 → 80 [ACK] Seq=1 Ack=1 Win=1024 Len=0
104	75.962181712	52:55:0a:00:02:01	Broadcast	ARP	60	Who has 192.168.10.2? Tell 10.0.2.1
105	75.962423607	10.0.2.4	66.29.153.49	ICMP	54	Timestamp request id=0x24f1, seq=0/0, ttl=52
106	75.963937541	192.168.10.2	66.29.153.49	ICMP	54	Timestamp request id=0x24f1, seq=0/0, ttl=42
107	75.987248349	10.0.2.4	192.168.1.1	DNS	85	Standard query 0x3de4 PTR 49.153.29.66.in-addr.
108	75.990297842	192.168.1.1	10.0.2.4	DNS	127	Standard query response 0x3de4 PTR 49.153.29.66
109	75.999331667	10.0.2.4	66.29.153.49	TCP	58	43154 → 80 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
110	75.999721188	192.168.10.2	66.29.153.49	TCP	58	43154 → 80 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
111	76.025184482	66.29.153.49	10.0.2.4	TCP	60	443 → 42898 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0
112	76.025215884	10.0.2.4	66.29.153.49	TCP	54	42898 → 443 [RST] Seq=1 Win=0 Len=0
113	76.026719147	52:55:0a:00:02:01	Broadcast	ARP	60	Who has 192.168.10.2? Tell 10.0.2.1

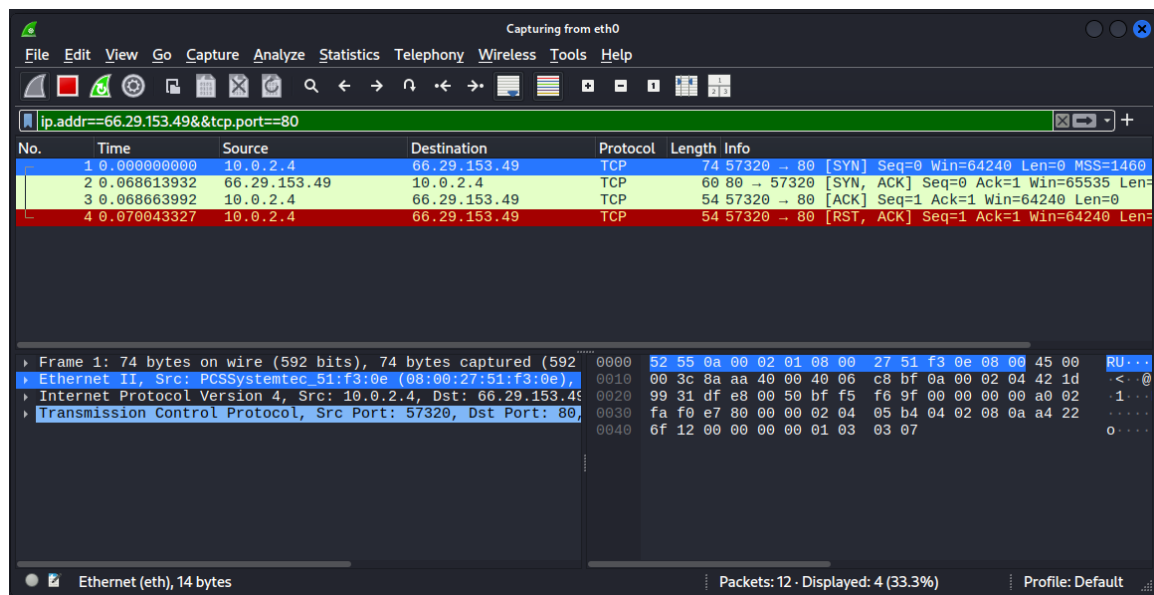
Frame 110: 58 bytes on wire (464 bits), 58 bytes captured (464 bits) on interface eth0
Ethernet II, Src: PCSSystemtec_51:f3:0e (08:00:27:51:f3:0e), Dst: 66.29.153.49 (08:00:27:51:f3:0e)
Internet Protocol Version 4, Src: 192.168.10.2, Dst: 66.29.153.49
Transmission Control Protocol, Src Port: 43154, Dst Port: 80, Seq: 43154, Win: 0, Len: 0

3) Time Fragmentation / Fragmented Scans

Behavior: Attackers split packets into small fragments or send them slowly over time to hide probe payloads. This can evade signature-based detection that inspects single packets.

Why it can bypass detection: If the IDS lacks full IP fragment reassembly or has limited buffer/timeouts, signatures won't match. Time-spaced fragments avoid triggering rate or threshold-based alerts.

Detection strategies: Enable full IP reassembly in IDS/Wireshark, adjust timeouts and buffers, and monitor unusual fragmentation or timing patterns with flow/session correlation.



Detection & Mitigation Recommendations

1. Use stateful network devices and enable connection tracking—this helps detect incomplete handshakes.
2. Enable IP fragment reassembly in IDS/IPS (and ensure adequate buffers/timeouts).
3. Correlate network flow telemetry (NetFlow/sFlow) with packet captures to detect distributed scans at low-rate
4. Implement rate limiting and SYN cookies to mitigate SYN-based evasions and floods.
5. Use behavioral detection (anomaly-based IDS) to spot patterns across decoys or time-sliced probes.
6. Log and centralize alerts; enrich with context (TCP options, TTL, packet sizes) for better triage.

7. Deploy honeypots to attract scans and analyze attacker tools and techniques safely.

Conclusion

The Wireshark analysis verified the TCP 3-way handshake between 10.0.2.4 and 66.29.153.49 on port 80 and demonstrated how attackers can use stealthy scanning techniques to avoid naive detection. Combining packet-level inspection with flow telemetry and behavioral analytics increases detection resilience against decoy and fragmentation-based evasions.

