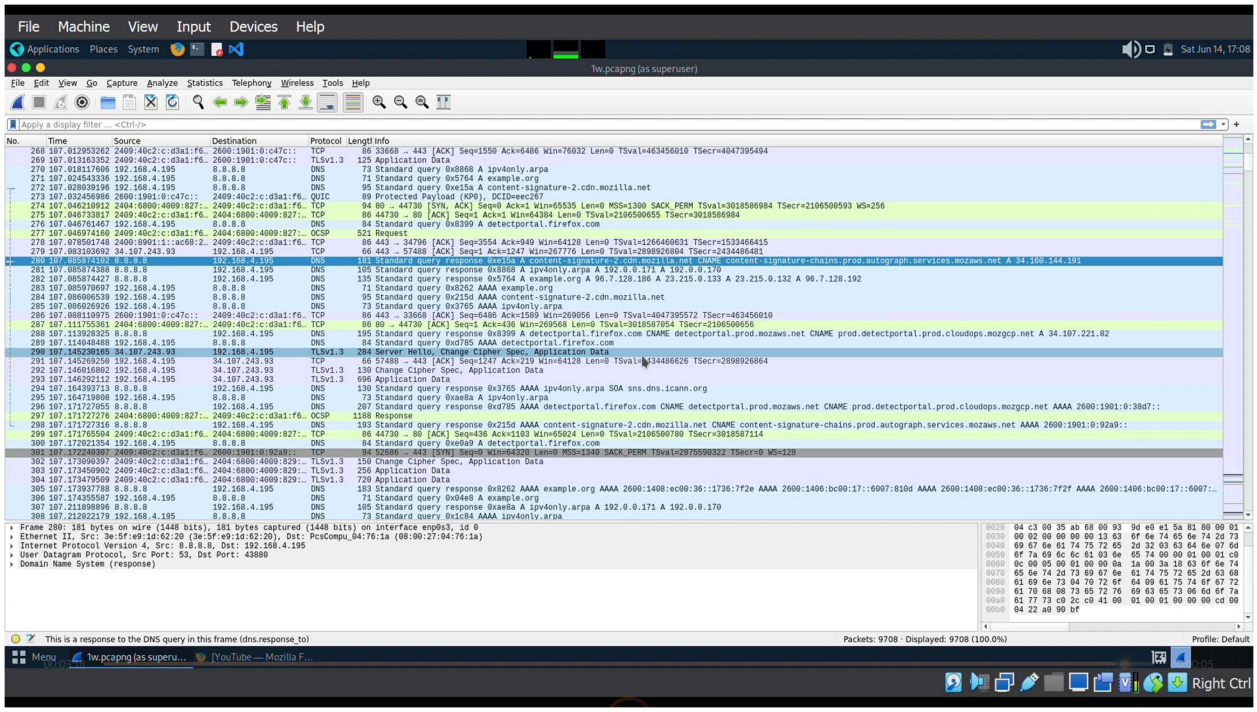
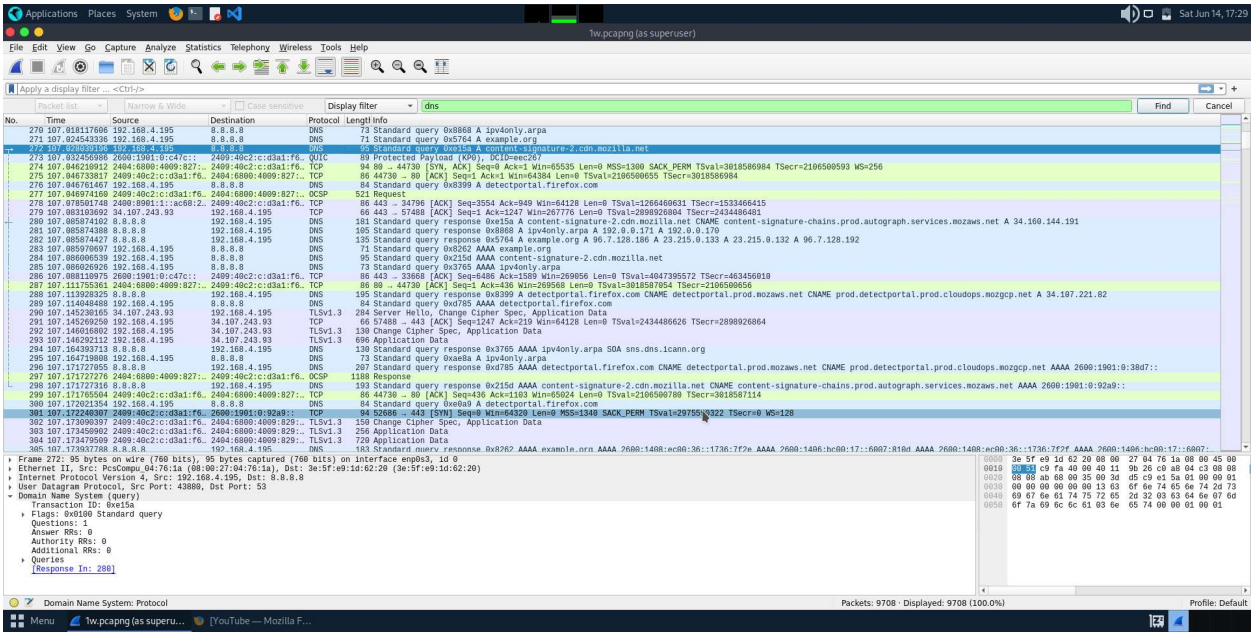
WIRESHARK REPORT



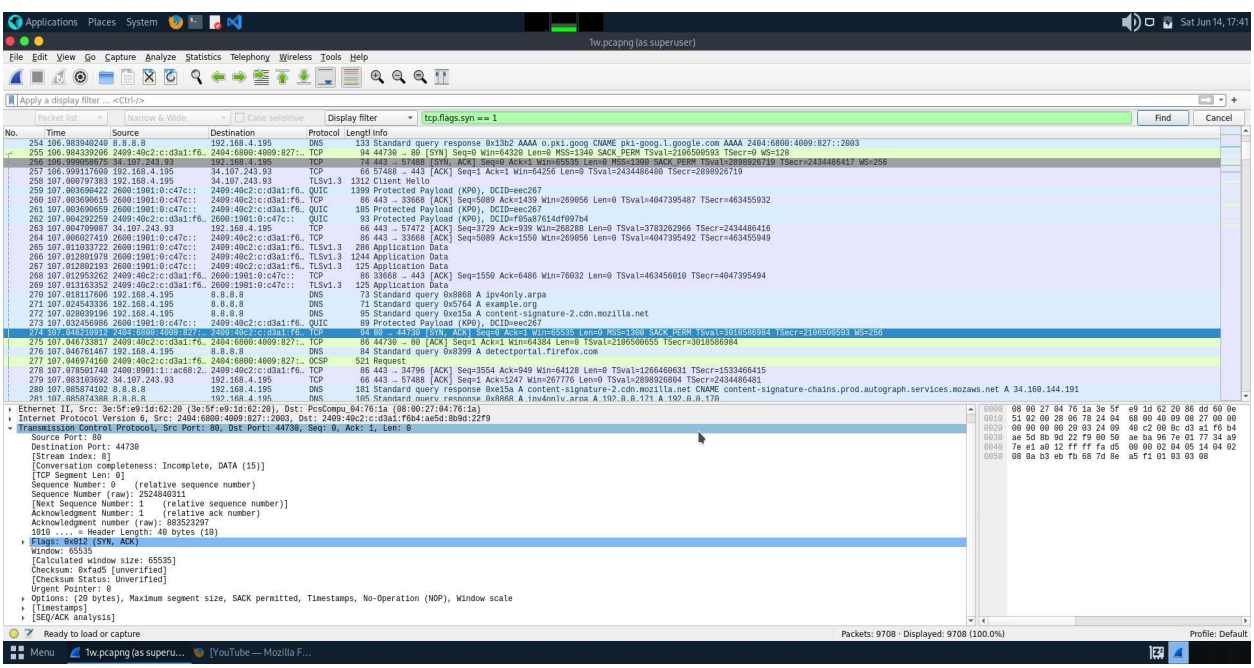
I filtered for packets sent to my own machine.

It means I was observing responses or traffic where **my IP was the destination** — which is normal during:

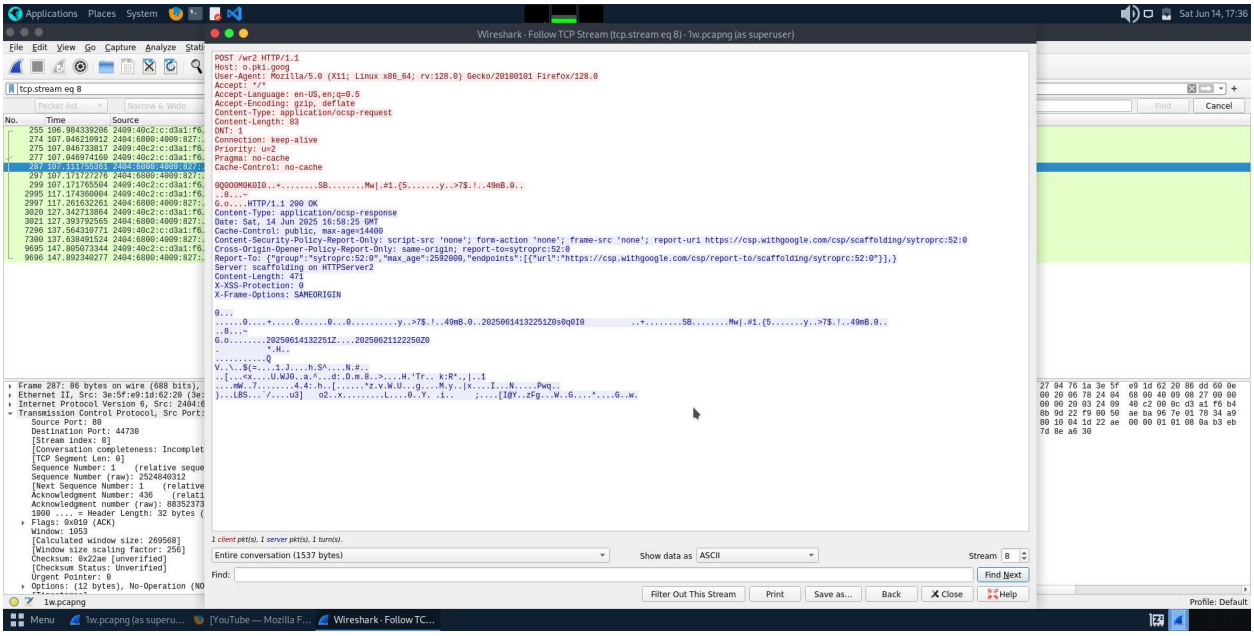
- **Browsing (HTTP/HTTPS)** — your machine receives data
- **DNS replies** — your system gets the resolved IP
- **TCP/UDP handshakes** — server responds to you
- **ICMP (ping)** — if someone pings you



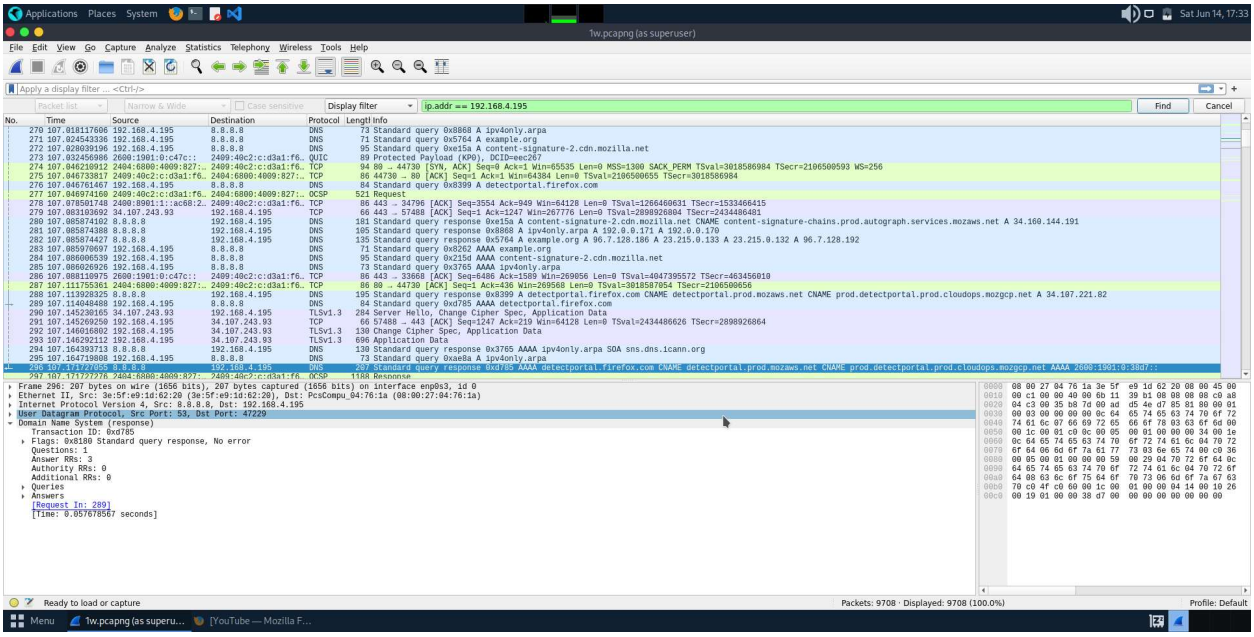
- **Protocol:** DNS
- **Layer:** Application
- **Details:** Multiple DNS queries like `content-signature-2.cdn.mozilla.net` and `example.org`
- **Insight:** User accessed web content which triggered domain resolution



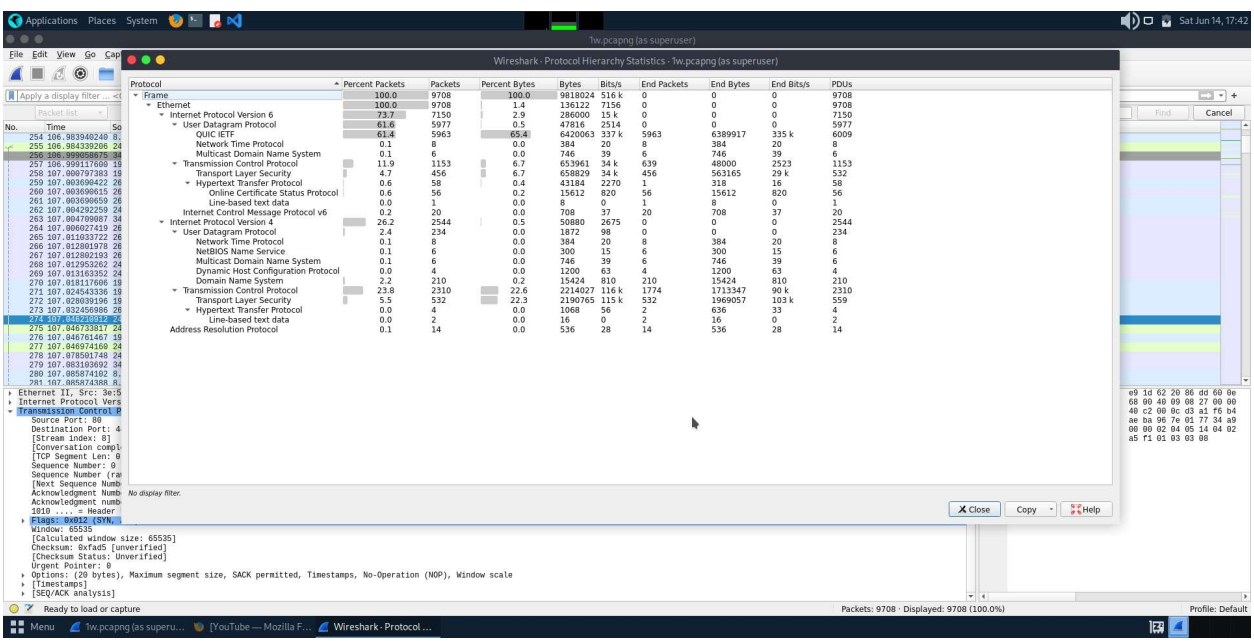
- **Protocol:** TCP
- **Layer:** Transport
- **Details:** SYN + ACK indicates a successful TCP handshake step
- **Insight:** Establishment of encrypted session possibly over HTTPS



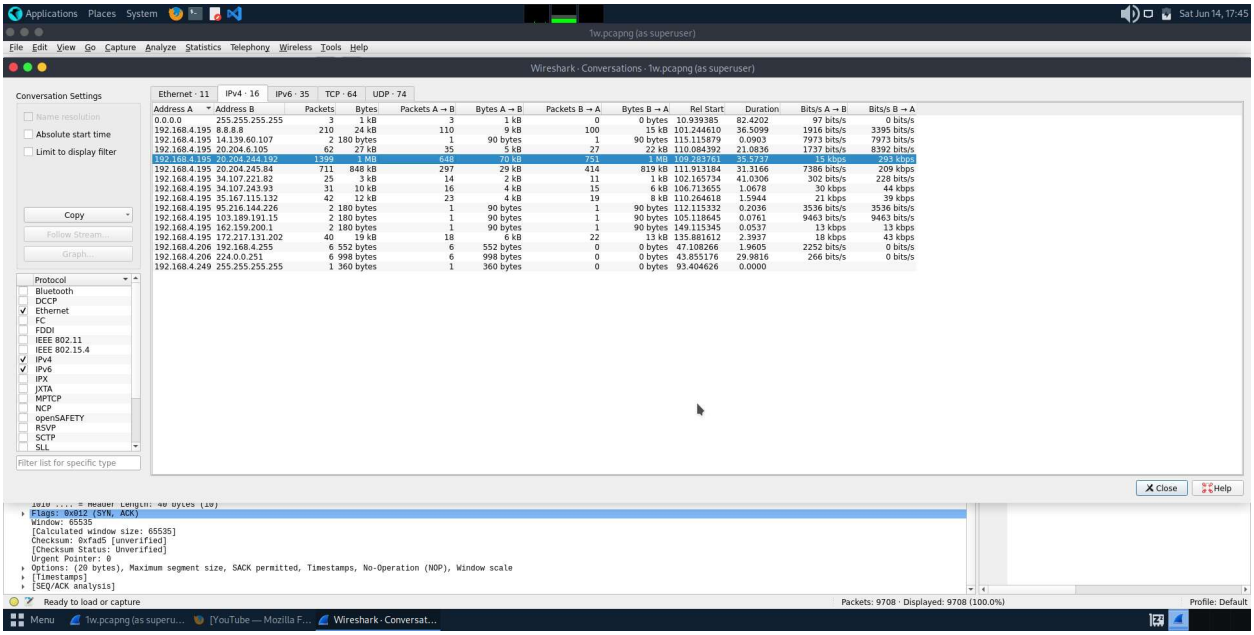
- **Protocol:** HTTP
- **Layer:** Application
- **Details:** POST request to Google's OCSP server
- **Insight:** Shows certificate validation over HTTP



- **Protocol:** DNS
- **Layer:** Application
- **Details:** Response to DNS queries shows resolved IPv6 addresses
- **Insight:** Connection attempted with IPv6-capable services like Firefox/Cloud services



- **Protocol:** Summary (All captured protocols)
- **Layer:** All layers
- **Details:** Shows percentage of each protocol observed (e.g., 65% TCP, 22% DNS)
- **Insight:** Helps understand traffic behavior and protocol usage distribution



Conversations View

Go to: `Statistics → Conversations → IPv4`

🎯 We'll see:

- All IPs your machine talked to
- How much data was sent/received
- Useful for identifying **which site or server** had most activity

OSI MODEL

Layer	Name	Function	Protocols/Examples	Data Unit	Key Devices
7	Application	User interfaces, network services	HTTP, FTP, SMTP, DNS, SSH	Data	Gateways, Firewalls

Layer	Name	Function	Protocols/Examples	Data Unit	Key Devices
		(HTTP, email, file transfers).			
6	Presentation	Data translation, encryption, compression (e.g., JPEG, SSL/TLS).	SSL/TLS, JPEG, MPEG, ASCII	Data	-
5	Session	Manages connections (setup, maintenance, termination).	NetBIOS, RPC, SIP	Data	-
4	Transport	End-to-end communication (reliability, flow control, error correction).	TCP, UDP, SCTP	Segment (TCP)	Firewalls, Load Balancers
3	Network	Logical addressing and routing (IP, routers).	IP, ICMP, OSPF, BGP, IPv4/IPv6	Packet	Routers, L3 Switches
2	Data Link	Physical addressing (MAC), error detection (switches, bridges).	Ethernet, PPP, VLANs, MAC	Frame	Switches, NICs
1	Physical	Transmits raw bitstream over physical media (cables, wireless).	USB, DSL, Fiber, IEEE 802.11 (Wi-Fi)	Bit	Hubs, Repeaters

Phishing Incident Response Playbook

1. Preparation

- Conduct awareness training for all employees.
- Deploy anti-phishing email filters.
- Ensure MFA (Multi-Factor Authentication) is enabled.
- Maintain updated contact info for SOC/IT teams.

---

2. Identification

- **Source:** Report from user or automatic detection tool (e.g., email flagged).
- **Signs of phishing:**
  - Suspicious sender email
  - Urgent request for credentials
  - Malicious link or attachment
- **Tools to use:**
  - Email header analyzer
  - VirusTotal (for attachment or URL)
  - SIEM alert dashboard

---

3. Containment

- Instruct user **not to interact** with the email.
- Block sender domain at email gateway.
- Quarantine or delete the email.
- Revoke access tokens or reset password if credentials were entered.

---

4. Eradication

- Delete phishing email from all affected inboxes.
- Clean infected endpoints (if link/attachment executed malware).
- Block phishing domain/IP at firewall or DNS level.

---

5. Recovery

- Restore clean email backups (if needed).
- Re-enable user access (if locked).
- Monitor the system and user activity closely for 24–72 hours.

6. Lessons Learned

- Document the timeline of the phishing incident.
- Review what detection/prevention worked and what failed.
- Update playbooks, detection rules, and training material.
- Share anonymized case with team for future awareness.

LOG PARSING SCRIPTS

1. Linux SSH Failed Login Parser (journal or auth.log)

Python:-

```
import re

with open("ssh_failures.log", "r") as file:
    for line in file:
        match = re.search(r"Failed password for( invalid user)? (\w+) from
([\d.]+) port (\d+)", line)
        if match:
            user = match.group(2)
            ip = match.group(3)
            port = match.group(4)
            print(f"User: {user} | IP: {ip} | Port: {port}")
```

Use case: After running

Bash:-

```
journalctl _COMM=sshd | grep "Failed password" > ssh_failures.log
```

2. Apache Access Log Parser

For `/var/log/apache2/access.log`:

Python:-

```
import re

log_file = "access.log"

with open(log_file, "r") as file:
    for line in file:
```

```
        match = re.search(r'(\d+\.\d+\.\d+\.\d+) - - \[(.*?)\] "GET (.*?)
HTTP/.*?" (\d+)', line)
        if match:
            ip = match.group(1)
            datetime = match.group(2)
            page = match.group(3)
            status = match.group(4)
            print(f"[{datetime}] {ip} requested {page} → Status: {status}")
```

Use case: Tracks who visited which page and what HTTP status they received.

---

3. Generic Keyword Alert System (e.g., for detecting 'error')

Python:-

```
keywords = ["error", "unauthorized", "denied", "failed"]
log_file = "syslog.log"  # Use any log file

with open(log_file, "r") as file:
    for line in file:
        if any(keyword.lower() in line.lower() for keyword in keywords):
            print(f"[ALERT] {line.strip()}")
```

Use case: Can be pointed at any log and used to find critical alerts.