NETWORK SOLUTIONS

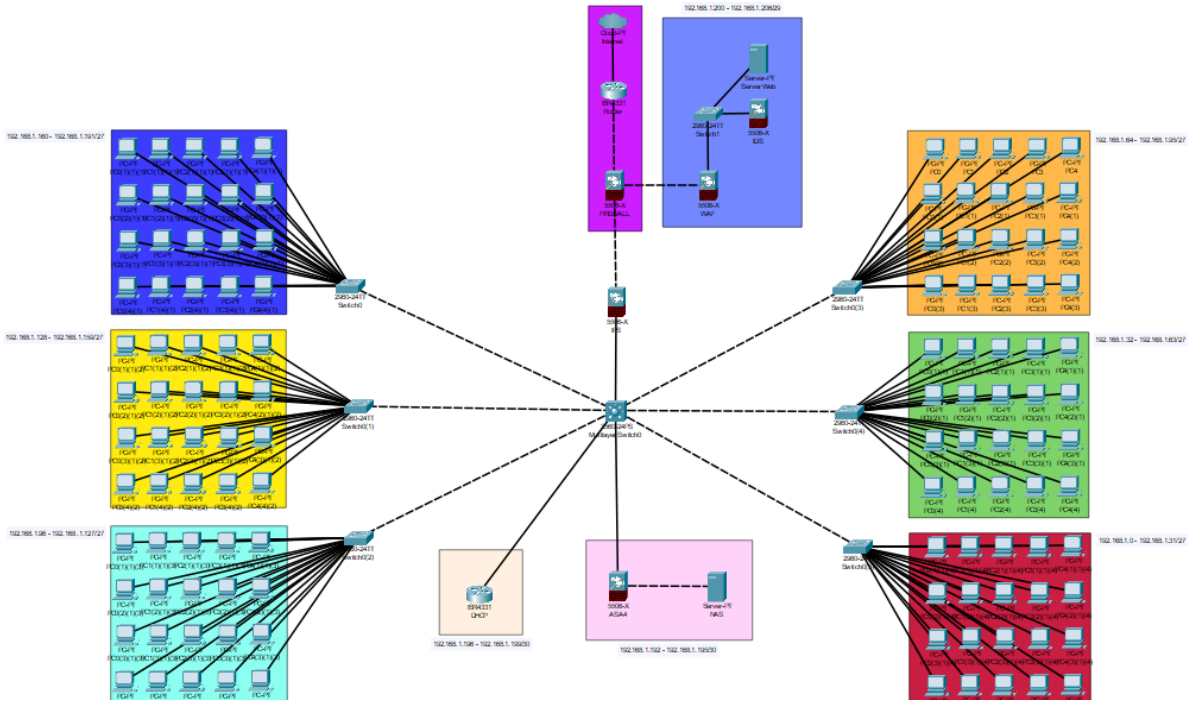
PROGETTO DI RETE PER LA COMPAGNIA THETA

La Compagnia Theta è un istituto scolastico tecnico con indirizzo informatico, e il progetto di rete descritto in questo documento è stato sviluppato per supportare le esigenze tecnologiche didattiche e amministrative di una realtà educativa moderna. La presente proposta progettuale viene redatta da **Network Solutions**, azienda specializzata nella progettazione e realizzazione di infrastrutture informatiche complesse. Il progetto ha come obiettivo la realizzazione di una **rete completa e funzionale** per la **Compagnia Theta**, un moderno edificio direzionale articolato su **6 piani**, con un totale di **120 postazioni informatiche**.

Obiettivi del progetto

- Fornire connettività stabile e veloce a **120 postazioni client**, 20 per piano
- Separare logicamente le reti tramite **VLAN** (amministrazione, uffici, server, laboratorio sicurezza)
- Implementare un'infrastruttura centralizzata con **server DHCP, DNS, Web, NAS**
- Garantire sicurezza e protezione dei dati tramite **firewall, IDS/IPS e VLAN ACL**
- Assicurare continuità operativa tramite **UPS locali e centrale**
- Abilitare test e simulazioni di vulnerabilità tramite **server DVWA (Metasploitable)**

- STRUTTURA DELLA RETE



In ciascuno dei 6 piani dell'edificio, i **20 PC** sono stati collegati a un **switch Layer 2 dedicato (Cisco SF350-24P-K9)**. Ogni switch di piano è poi connesso a un **core switch Layer 3 (Cisco WS-C2960XR-24TS-I)**, situato nella sala server centrale.

Abbiamo optato per l'utilizzo di uno switch Layer 3 centrale, in quanto consente di gestire il routing tra le diverse sottoreti direttamente a livello di switching. Questa soluzione permette, ad esempio, di assegnare al **NAS (QNAP TS-873XU-RP)** una **sottorete dedicata**, migliorando il livello di sicurezza complessivo. Il traffico verso il NAS può così essere segmentato e controllato attraverso regole di accesso basate sui privilegi degli utenti.

Il NAS è collegato direttamente allo switch Layer 3 centrale per garantire **alta velocità di accesso** e centralizzazione delle risorse condivise. Questo collegamento è protetto da un **sistema IPS (Cisco ASA5512-IPS-K8)**, configurato per monitorare in tempo reale il traffico verso il NAS, bloccando eventuali accessi non autorizzati o traffico malevolo.

Un **server DHCP dedicato (HPE ProLiant MicroServer Gen10)**, collegato allo switch Layer 3, ha il compito di assegnare dinamicamente indirizzi IP a tutti i dispositivi della rete. Questo server opera in modalità relay per supportare tutte le VLAN definite nella struttura.

Un secondo dispositivo **IPS** è stato posizionato tra lo switch Layer 3 e il router Cisco RV340, per monitorare il traffico in ingresso e in uscita verso l'esterno. Questo

rappresenta una **prima linea di difesa** per proteggere l'intera infrastruttura IT da minacce provenienti dall'esterno.

Per la protezione perimetrale, è stato installato un **firewall Fortinet FortiGate 100F**, posto tra la rete interna e la connessione a Internet. Il firewall blocca tutte le connessioni non autorizzate, consentendo soltanto il traffico definito come sicuro.

A valle del firewall, prima della **DMZ** (zona demilitarizzata), è stato inserito un **WAF (Web Application Firewall)** per proteggere il **server web DVWA** da attacchi a livello applicativo come SQL injection, XSS e altre vulnerabilità comuni.

All'interno della DMZ, un **switch dedicato** è stato configurato per instradare il traffico del server web verso un **IDS**. Questo terzo dispositivo **IDS (Cisco ASA5512-IPS-K8)** è incaricato di monitorare tutto il traffico in entrata e in uscita dal server web, allertando in tempo reale in caso di attività sospette.

Questa struttura di rete a strati consente di garantire **prestazioni elevate**, **isolamento delle componenti critiche**, e un **sistema di sicurezza multilivello** coerente con i requisiti di un'infrastruttura moderna per una realtà scolastica avanzata.

● SUBNETTING

Una consegna opzionale del progetto richiedeva di effettuare il **subnetting dell’intera infrastruttura**, con l’obiettivo di assegnare subnet appropriate ai diversi segmenti della rete. Considerando la struttura dell'edificio (6 piani, 20 PC ciascuno) e la necessità di reti dedicate per la **DMZ**, per il **NAS** e per il **server DHCP**, sono state previste **9 subnet** in totale.

Per soddisfare i requisiti, è stata adottata una strategia di **VLSM (Variable Length Subnet Masking)** che permette di utilizzare subnet con maschere differenti in base al numero di host richiesti:

- Assegnazione di **/27 (255.255.255.224)** per i 6 piani dell’edificio, con **30 host utilizzabili** ciascuno
- Assegnazione di **/30 (255.255.255.252)** per NAS e DHCP, con **2 host utilizzabili**
- Assegnazione di **/29 (255.255.255.248)** per la DMZ, con **6 host utilizzabili**

Questa soluzione consente un uso efficiente dello spazio di indirizzamento, migliorando la sicurezza e la segmentazione della rete.

- Assegnare **/30 (255.255.255.252)** a NAS e server DHCP, ognuna con **2 host utilizzabili**
- Assegnare **/29 (255.255.255.248)** alla subnet DMZ, con **6 host utilizzabili**

192.168.1.0/27	Piano Terra
192.168.1.32/27	Piano 1
192.168.1.64/27	Piano 2
192.168.1.96/27	Piano 3
192.168.1.128/27	Piano 4
192.168.1.160/27	Piano 5
192.168.1.192/30	NAS, IPS
192.168.1.196/30	DHCP
192.168.1.200/29	DMZ

PREVENTIVO

Questo documento rappresenta il preventivo completo per la realizzazione dell'infrastruttura IT della **Compagnia Theta**, su **6 piani** con un totale di **120 postazioni di lavoro**. Il preventivo include **hardware, manodopera, cablaggio, installazione, configurazione** e **stima dettagliata dei materiali**.

COMPUTO METRICO E COMPONENTI HARDWARE

Voce	**Quantità**	**Costo Unitario (€/$)**	**Totale (€/$)**

PC client (desktop)	120	€500	€60.000
Switch Layer 2 (Cisco SF350-24P-K9)	7	€259	€1.813
Switch Layer 3 (Cisco WS-C2960XR-24TS-I)	1	€1.850	€1.850
Server DHCP/DNS (HPE ProLiant Gen10)	2	€1.236	€2.472
Router Cisco RV340-K9-G5	1	€1.995	€1.995
Firewall FortiGate 100F	1	€1.868	€1.868
NAS QNAP TS-873XU-RP	1	€1.573	€1.573
Hard Disk NAS (Seagate IronWolf 12TB)	3	€267	€800
Server Web (Lenovo SR530)	1	€1.979	€1.979
IDS/IPS (Cisco ASA5512-IPS-K8)	3	€1.275	€3.825
Patch panel 24 porte	6	€90	€540
Armadio rack 12U	6	€400	€2.400
Cablaggio strutturato (Cat 6, 2000m)	2000 m	€0,50/m	€1.000
Patch cord e connettori	150	€5	€750
UPS 1000VA (per piano)	6	€250	€1.500
UPS centrale (3000VA)	1	€1.000	€1.000

● MANODOPERA E SERVIZI

Servizio	Quantità	Costo Unitario (€)	Totale (€)
Installazione e configurazione hardware	1	€2.500	€2.500
Stesura e certificazione cablaggio strutturato	1	€3.000	€3.000
Configurazione rete, VLAN, routing e sicurezza	1	€1.500	€1.500
Formazione del personale IT	1	€800	€800
Progetto tecnico documentato e consegna finale	1	€600	€600

● TOTALE COMPLESSIVO DEL PROGETTO

- 93.265,00

✅ **Include:**

- 120 postazioni PC
- 7 switch di piano + 1 switch core L3
- Servizi core: DHCP, DNS, Web, IDS/IPS, Firewall, NAS
- Protezione con UPS locale e centrale
- Installazione, configurazione, formazione

● TESTING DELLA RETE

Per completare correttamente i **collegamenti di rete** e simulare diversi scenari reali, è stato necessario integrare **porzioni di codice Python**. Questi script sono stati fondamentali per:

- Effettuare configurazioni di rete mirate
- Inviare richieste personalizzate tra client e server
- Inserire, analizzare e ricevere dati all'interno della rete simulata

In particolare, il sistema è stato strutturato con **Kali Linux come client** e **Metasploitable come server**, creando un contesto realistico utile sia per il testing che per l'apprendimento pratico delle comunicazioni client-server.

Il processo è stato suddiviso in cinque fasi operative:

1. Configurazione del sistema
2. Scansione delle porte
3. Considerazioni di sicurezza sulle porte
4. Test dei verbi
5. Socket di rete (Bonus)

1. Configurazione del sistema

Abbiamo avviato la configurazione dell'ambiente di rete utilizzando **pfSense** come **firewall/router centrale**, con l'obiettivo di permettere la comunicazione tra le macchine **Kali Linux** (client) e **Metasploitable** (server).

- Le configurazioni di rete per le macchine sono:
  - **psfSense**
    - WAN 192.168.1.1
    - LAN 192.168.2.1
    - OPT1 192.168.3.1
  - **Kali**
    - LAN 192.168.2.10
  - **Metasploitable**
    - LAN 192.168.3.10

Il **primo ostacolo** che abbiamo incontrato è stato l'**accesso alla piattaforma phpMyAdmin tramite browser**, che richiedeva **user e password** validi.

Per risolvere il problema, abbiamo effettuato l'**accesso diretto al database MySQL** dalla macchina server (Metasploitable), al fine di **modificare la password di accesso dell'utente**. Durante questa operazione, abbiamo anche **concesso**

all'utente `root` tutti i privilegi, in modo da poter eseguire qualsiasi operazione sul database senza restrizioni.

Questa configurazione è stata effettuata per **esigenze didattiche e di laboratorio**, e va **evitata in ambienti di produzione**, dove è essenziale applicare il principio del **minimo privilegio** per motivi di sicurezza.

```
mysql> UPDATE user SET password=PASSWORD('root') WHERE user='root' AND host='%';
Query OK, 0 rows affected (0.00 sec)
Rows matched: 1  Changed: 0  Warnings: 0

mysql> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.00 sec)

mysql> SELECT user, host, password FROM user WHERE user='root';
+------+------+-------------------------------------------+
| user | host | password                                  |
+------+------+-------------------------------------------+
| root | %    | *81F5E21E35407D884A6CD4A731AEBFB6AF209E1B |
+------+------+-------------------------------------------+
1 row in set (0.00 sec)

mysql>
```

```
msfadmin@metasploitable:~$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 13
Server version: 5.0.51a-3ubuntu5 (Ubuntu)

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> SELECT user, host, password FROM mysql.user WHERE user = 'root';
+------+------+----------+
| user | host | password |
+------+------+----------+
| root | %    |          |
+------+------+----------+
1 row in set (0.00 sec)
```

2. Scansione delle porte

Dopo la configurazione iniziale, siamo passati alla fase di **scansione delle porte** con l'obiettivo di **identificare i servizi attivi su Metasploitable2**. Questa attività ci ha permesso di analizzare quali porte risultassero **aperte, chiuse o filtrate**, per poi valutarne le implicazioni in termini di sicurezza.

La scansione è stata eseguita su un **range prestabilito di porte**, focalizzandoci principalmente su quelle **più comunemente utilizzate nei servizi di rete**.

Per svolgere questa attività, abbiamo sviluppato e utilizzato uno **script Python personalizzato**, capace di iterare sulle porte desiderate ed effettuare connessioni TCP al fine di verificarne lo stato.

Tra le porte principali abbiamo avuto i seguenti riscontri:

Numero Porta	Tipo Porta	Stato
20	FTP	Chiusa
21	FTP	Aperta
22	SSH	Aperta
23	TELNET	Aperta
25	SMTP	Aperta
53	DNS	Aperta
67 - 68	DHCP	Chiuse
80	HTTP	Aperta
110	POP	Chiusa
111	PORT MAPPER	Aperta
137-138	NETBIOS	Chiusa
139	NETBIOS	Aperta
143	IMAP	Chiusa
161-162	SNMP	Chiuse
443	HTTPS	Chiusa
445	SAMBA	Aperta
989 - 990	FTPS	Chiuse

3. Considerazioni di sicurezza delle porte

Porta 21 (FTP, File Transfer Protocol): È una porta che ospita un server FTP, non è sicura dato che spesso questi server usano l'autenticazione anonima e non possiedono crittografia.

Porta 22 (SSH, Secure Shell): È una porta spesso associata ad un account con password poco sicure, per questo sono soggette a molte vulnerabilità ed è necessario distribuire patch di sicurezza costantemente.

Porta 23 (Telnet): È una porta collegata ad un servizio di comunicazione ormai arretrato che invia dati senza crittografia, per questo è esposto a molte minacce.

Porta 25 (SMTP, Simple Mail Transfer Protocol): Serve al proprio client per il trasferimento di messaggi su Internet, i contro sono la mancanza di crittografia e di contrasto allo spam, proprio per questo spesso viene bloccata dagli ISP.

Porta 53 (DNS, Domain Name Service): È la porta assegnata al DNS per la conversione di nomi in indirizzi IP leggibili dalla macchina. Il problema è che viene raramente monitorata, infatti viene spesso attaccata per l'esfiltrazione di dati. Un modo per renderla più sicura è il monitoraggio e il filtraggio del traffico DNS.

Porta 80 (HTTP): Una delle porte più utilizzate perché serve per stabilire connessioni HTTP in modo da poter scambiare dati tra un browser e un server. Il problema principale è che, non essendo sicura, viene spesso utilizzata per effettuare attacchi DDoS. Il consiglio è bloccarla ed utilizzare solo la porta 443 (HTTPS), che invece utilizza crittografia.

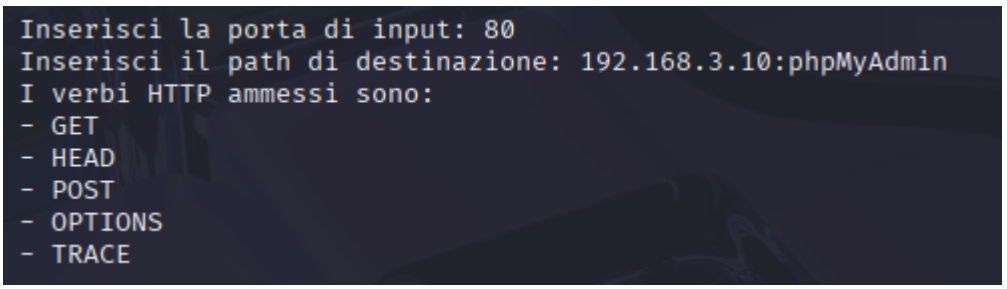
Porta 111 (RPC, Remote Procedure Call): Serve per collegare due computer non facenti parte della stessa rete, ad esempio usa il NFS (Network File System) che permette ad un computer di accedere ai dati di un altro come se fossero entrambi nella stessa rete locale. Presenta diverse vulnerabilità e un modo per contrastarle è la creazione di una regola nel firewall che permetta l'utilizzo di Portmapper solo all'interno della rete locale o tramite specifici indirizzi IP.

Porta 139 (SMB, Server Message Block): Permette alle applicazioni su diversi computer di comunicare all'interno di una rete locale; è stata sviluppata negli anni '80, periodo in cui la sicurezza non era di certo la priorità. Per proteggersi, bisognerebbe creare una regola nel firewall che blocchi qualsiasi connessione in entrata da fonti non autorizzate.

Porta 445 (SMB, Server Message Block): Nuove versioni di SMB che non utilizzano più la porta 139 ma la 445. Per proteggerla si seguono le stesse istruzioni descritte per la porta 139.

4. Richiesta dei verbi ammessi

- Per questa fase abbiamo creato un programma in Python ed effettuato una richiesta OPTIONS al server phpMyAdmin tramite porta 80 (HTTP).
- Abbiamo ricevuto come risultati i seguenti verbi ammessi:



GET:
Viene utilizzato per richiedere una risorsa (come una pagina web, un'immagine, un file, ecc.) dal server. Non modifica nulla all'interno del server, ma si occupa solo della visualizzazione o del recupero della risorsa richiesta.
Esempio: Se visiti una pagina web, il tuo browser invia una richiesta GET per ottenere la pagina.

HEAD:
Funziona in modo simile a GET, ma non restituisce il corpo della risposta. Viene utilizzato per ottenere solo le intestazioni (Headers).
Esempio: Un utilizzo comune è verificare se una pagina esiste prima di scaricarla completamente.

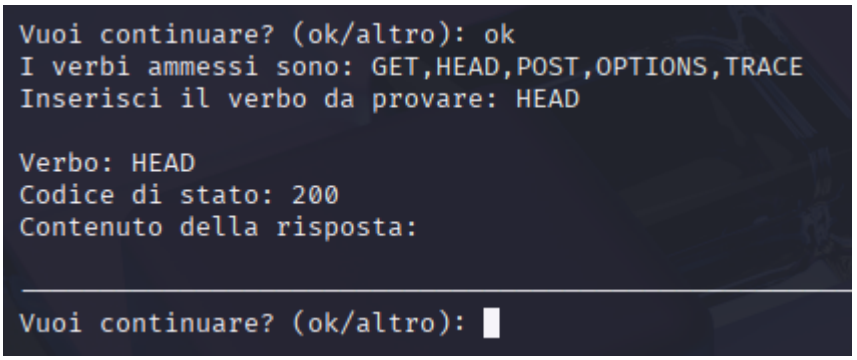
POST:
Viene utilizzato per inviare dati al server, tipicamente per creare o aggiornare una risorsa. I dati vengono inviati nel corpo della richiesta.
Esempio: Quando compili un modulo su una pagina web (come il login), i dati vengono inviati al server tramite una richiesta POST.

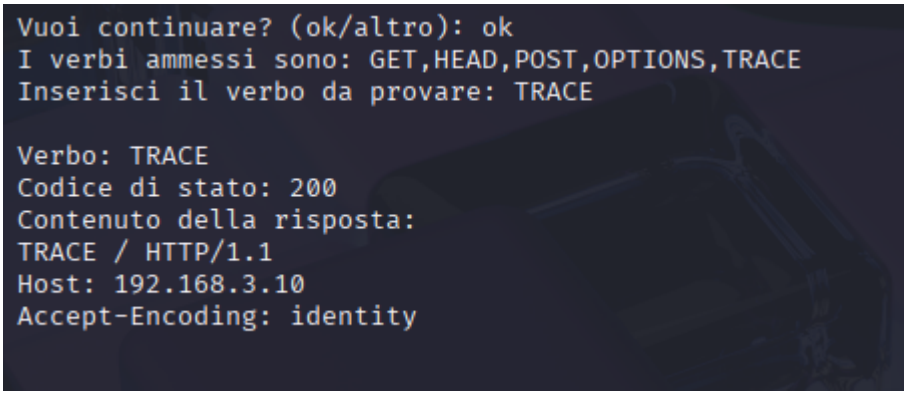
OPTIONS:
Utilizzato per ottenere i verbi HTTP disponibili per una risorsa o un server; riceve in risposta l'elenco dei verbi supportati (GET, POST, PUT, DELETE, ecc.).
Esempio: Un'applicazione web può usare OPTIONS per verificare se è possibile comunicare con il server e con quali verbi HTTP.

TRACE:
Serve per tracciare il percorso che una richiesta HTTP compie attraverso i vari server.
Esempio: In situazioni di debugging, TRACE può essere utile per vedere come una richiesta arriva al server.

5. Test dei verbi

- Per eseguire il test abbiamo creato un programma in Python che permette di scegliere tra test singolo, inserendo il verbo specifico a cui vogliamo effettuare la richiesta, oppure il test di tutti i verbi disponibili in sequenza. Ecco alcuni esempi:





- Nella richiesta ci soffermiamo su un primo codice di risposta, il codice di stato = 200, che ci assicura la corretta elaborazione della richiesta, la quale verrà esposta nelle righe successive.

6. Socket di rete (Bonus)

- E’ stato sviluppato un altro programma su Metasploitable che si occupa di fare il socket di rete sulla porta 80.

- Per poter attivare il socket abbiamo avuto la necessità di spegnere il servizio apache2 per liberare la porta 80.

- Quindi abbiamo fatto partire i 3 programmi in contemporanea nelle due macchine Client e Server, in Kali lo scan e la richiesta dei verbi, in metasploitable il socket.

- Da Kali visualizziamo questo risultato:

```
┌──(kali㉿kali)-[~/Desktop/ProgrammiPY/BuildWeek_1]
└─$ python Programma.py
Inserisci l'indirizzo IP da testare: 192.168.2.7
Inserisci la porta iniziale: 75
Inserisci la porta finale: 85
La porta 75 è chiusa
La porta 76 è chiusa
La porta 77 è chiusa
La porta 78 è chiusa
La porta 79 è chiusa
La porta 80 è chiusa
La porta 81 è chiusa
La porta 82 è chiusa
La porta 83 è chiusa
La porta 84 è chiusa
La porta 85 è chiusa

Inserisci la porta di input: 80
Inserisci il path di destinazione: phpMyAdmin
```

```
msfadmin@metasploitable:~$ sudo /etc/init.d/apache2 stop
 * Stopping web server apache2
msfadmin@metasploitable:~$ sudo python Socket_rete.py
Server started, waiting for connections
('Client connected with address ', ('192.168.50.5', 44774))
msfadmin@metasploitable:~$ sudo python Socket_rete.py
Server started, waiting for connections
('Client connected with address ', ('192.168.50.5', 45372))
OPTIONS /phpMyAdmin HTTP/1.1
Host: 192.168.2.7:80
Accept-Encoding: identity
```

- Questo invece l'output di metasploitable

- Possiamo notare la prima riga di comando dove spegniamo apache2, successivamente abbiamo avviato il programma Python socket di rete e gli abbiamo assegnato la porta 80

- Quando si avvia il socket visualizziamo served started, waiting for connections, a questo punto da kali iniziamo ad avviare il programma che effettua lo scan delle porte e otteniamo su metasploitable la riga client connected con l'IP di kali 192.168.50.5 con porta 44774. Effettuato lo scan correttamente si disconnette e riavviamo il socket di rete, poi continuiamo con il programma in Kali con la richiesta OPTIONS per ricevere i verbi ammessi.

- Visualizziamo sempre client connection di Kali, Kali invia OPTIONS che è proprio il verbo che vogliamo inviare a Metasploitable.

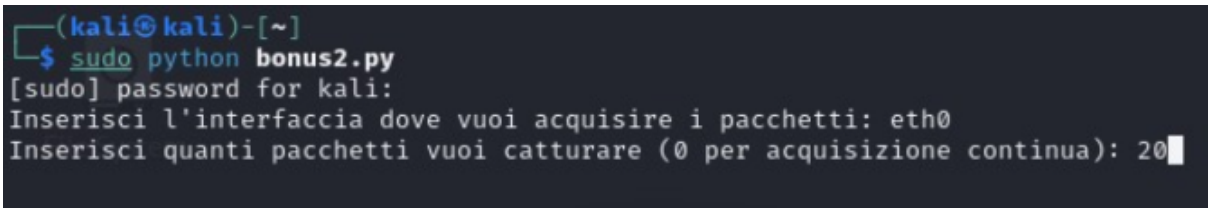
- Dall’output di Metasploitable vediamo la voce OPTIONS.

- A questo punto dato che abbiamo dovuto spegnere apache2, Kali non otterrà risposta.

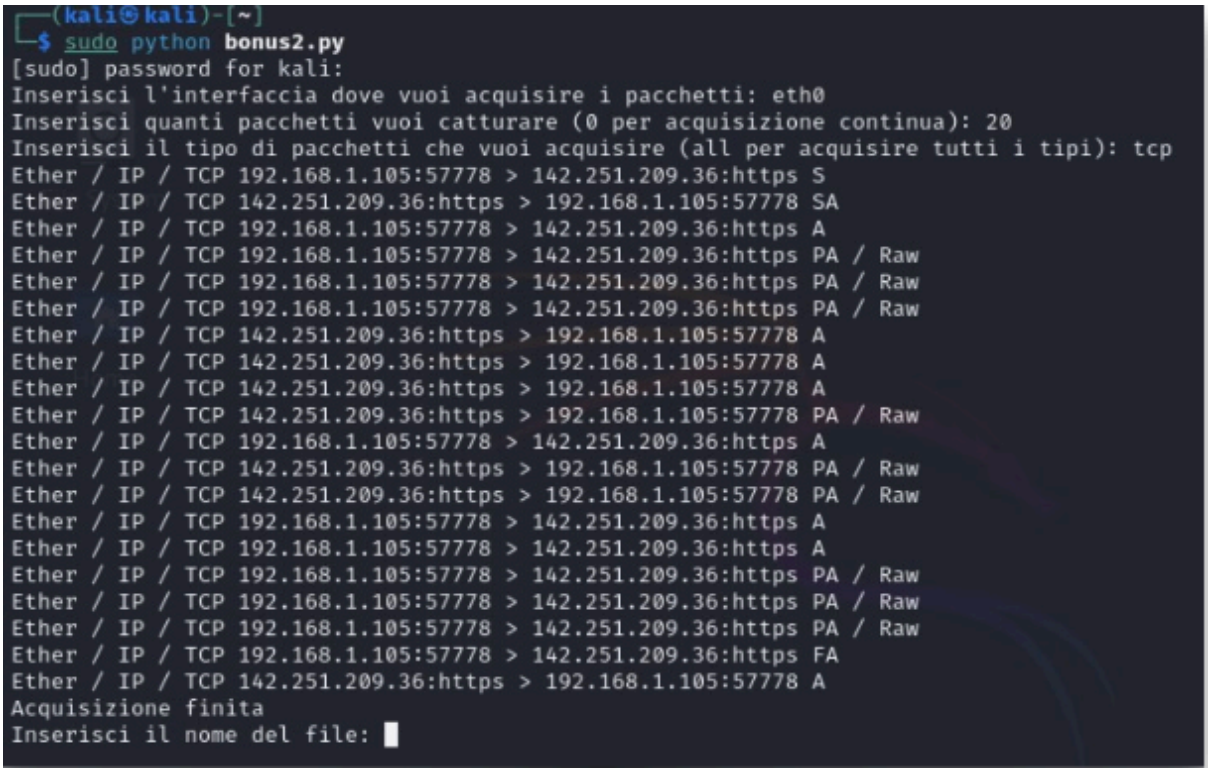
ESERCIZIO BONUS 2

Questo esercizio richiedeva la realizzazione di un programma in **Python** che sfruttasse la libreria **Scapy** per effettuare la cattura e l’analisi del traffico di rete. L’esecuzione del programma avviene con privilegi elevati, tramite il comando `sudo`, in quanto Scapy necessita dei permessi di amministratore per accedere alle interfacce di rete..

All’avvio, il programma richiede all’utente di specificare l’interfaccia di rete da utilizzare (es. eth0) e il numero di pacchetti da acquisire Nell’esempio riportato, sono stati scelti 20 pacchetti da catturare.



Successivamente, viene richiesto il tipo di pacchetti da analizzare; è possibile specificare un protocollo come **TCP**, permettendo così di concentrarsi esclusivamente su questo tipo di traffico. Al termine dell'acquisizione, il programma restituisce un elenco di pacchetti intercettati, contenenti informazioni dettagliate come **IP sorgente e destinazione**, **porte utilizzate** e **protocolli coinvolti**.



I dati acquisiti vengono poi salvati in un file di output con estensione `.pcap`, utile per un'analisi successiva. In questo caso, il file è stato nominato `prova.pcap`.

Per procedere con l'analisi offline, si può aprire **Scapy** dal terminale con `sudo scapy` e caricare direttamente il file `.pcap` generato. In questo modo è possibile ispezionare ogni pacchetto, applicare filtri, e comprendere meglio il comportamento del traffico in rete.