Team 7

Progetto di rete per la Compagnia Theta

Indice

Contatti d'ingaggio

Theta

Henri Kapidani	Amministratore Delegato

Team 7

Manuel Burgio	Consulente IT
Andrea Monopoli	Web Designer
Jeremiah Mendoza	Programmatore
Michele Girardi	Programmatore
Davide Talamo	Supervisore

Riepilogo approccio

Siamo stati ingaggiati dalla compagnia Theta per sviluppare un preventivo di spesa e un progetto di rete per la loro infrastruttura IT.
La compagnia Theta ci ha dato delle indicazioni sui requisiti e le componenti necessari:

- Struttura dell'edificio: 6 piani
- Dispositivi previsti: 20 computer per piano, per un totale di 120 computer
- Componenti aggiuntivi:
  - 1 Web server (rappresentato dalla macchina DVWA di Metasploitable)
  - 1 Firewall perimetrale
  - 1 NAS (Network Attached Storage)
  - 3 IDS/IPS (Intrusion Detection System / Intrusion Prevention System)

I requisiti della rete interna aziendale sono i seguenti:

- Switch per ogni piano: Collegare i 20 computer di ciascun piano a uno switch dedicato.
- Router: Collegare tutti gli switch dei vari piani a un router centrale.
- Firewall: Posizionare il firewall perimetrale tra il router interno e la connessione a Internet.
- NAS: Collegare il NAS allo switch al piano terra (vicino al router) per garantire l'accesso ai dati da parte di tutti i computer aziendali.
- IDS/IPS: Implementare 3 IDS/IPS nel perimetro interno per monitorare il traffico di rete e prevenire intrusioni.

I requisiti della rete esterna (Internet) sono i seguenti:

- Connessione a Internet: Collegare il firewall perimetrale a Internet.
- Web Server: Posizionare il web server (DVWA di Metasploitable) nella zona demilitarizzata (DMZ) tra il firewall e la connessione a Internet, garantendo così un accesso sicuro dall'esterno.
- Se avete bisogno di un altro firewall potete comprarlo e montarlo.

Architettura Fisica

L'azienda Theta è un complesso ospedaliero di 6 piani di cui 1 seminterrato.

Ogni piano sarà dotato di 20 PC suddivisi nelle rispettive stanze e implementeremo inoltre il seguente hardware:

- Un patch panel a 24 porte (Networking - IT, P/24-PS-C6GN) per la terminazione ordinata dei cavi di rete.
- Uno switch Gigabit Ethernet a 24 porte (Networking - IT, I-SW/HUB TEG1024G) per la connettività dei 20 PC all-in-one locali.
- Un sistema di continuità dell'alimentazione (UPS) Line Interactive da 600VA (Networking - IT, I-CUXL6MP600) per garantire l'operatività degli apparati di rete in caso di brevi interruzioni elettriche.
- Multiprese per l'alimentazione dei dispositivi.
- Cablaggio: Tutti i PC all-in-one (Client, 907L6EA, HP ProOne 240 G10) saranno connessi alla rete tramite cavi di rete Cat 6 Slim da 2 metri (Client, ICCO6U6-SLIM-020BK), garantendo una connessione Gigabit affidabile.

Nel seminterrato prevediamo una sala CED dove implementeremo:

- Il NAS (NAS, RS1221+, Synology RS1221+) con una capacità iniziale di 4TB, espandibile con 8 dischi rigidi interni da 12TB Western Digital Red Plus (NAS, WC120EFBX) per garantire una scalabilità dello storage nel tempo. Sarà alloggiato in un armadio rack con patch panel a 24 porte (Networking - IT, P/24-PS-C6GN) e uno switch Gigabit Ethernet a 48 porte (Networking - IT, ISW/HUB-48GS) per la connettività dei server e l'aggregazione del traffico.
- IDS/IPS: Il Netgate 6100 MAX (Firewall/IDS/IPS, USGFLEX6100-EU0102F) con pfSense Plus sarà configurato come firewall perimetrale e integrerà funzionalità IDS/IPS per l'analisi del traffico in ingresso e in uscita da Internet. Ulteriori funzionalità IDS/IPS (implementate tramite pfSense Plus e potenzialmente pacchetti come Suricata) saranno configurate per monitorare il traffico interno tra le subnet dei piani e per proteggere specificamente i server nel CED:

NB: La nostra azienda si occuperà dell'installazione e della configurazione degli apparati, supervisionando al contempo l'azienda di elettricisti incaricata dell'infilaggio e della canalizzazione dei cavi.

Architettura Logica

La segmentazione della rete sarà ottenuta attraverso l'utilizzo di subnet IP fisicamente separate per ciascun piano dell'edificio. Ogni piano sarà configurato con una propria subnet basata sulla classe 10.10.x.0/24, dove 'x' rappresenta il numero del piano. Questa separazione a livello di indirizzamento IP richiederà che ogni piano sia connesso a una rete fisica separata gestita dagli switch locali.

- Seminterrato: Rete 10.10.100.0/24, Gateway: 10.10.100.1
- Piano Terra: Rete 10.10.55.0/24, Gateway: 10.10.55.1
- Piano 1: Rete 10.10.10.0/24, Gateway: 10.10.10.1
- Piano 2: Rete 10.10.20.0/24, Gateway: 10.10.20.1
- Piano 3: Rete 10.10.30.0/24, Gateway: 10.10.30.1
- Piano 4: Rete 10.10.40.0/24, Gateway: 10.10.40.1

Ogni dispositivo all'interno di un piano riceverà un indirizzo IP appartenente alla subnet del rispettivo piano. Il primo indirizzo IP di ogni subnet (.1) sarà riservato come gateway per quel piano, rappresentato dall'interfaccia dello switch di livello 3 che connette le diverse subnet. Lo switch di livello 3, installato anche per motivi di fattibilità di calata cavi e per salvaguardare che l'eventuale caduta di uno switch causi il malfunzionamento dell'intera rete, viene configurato per eseguire il servizio di DHCP necessario vista la quantità di computer che fanno parte della rete Theta e di DNS.

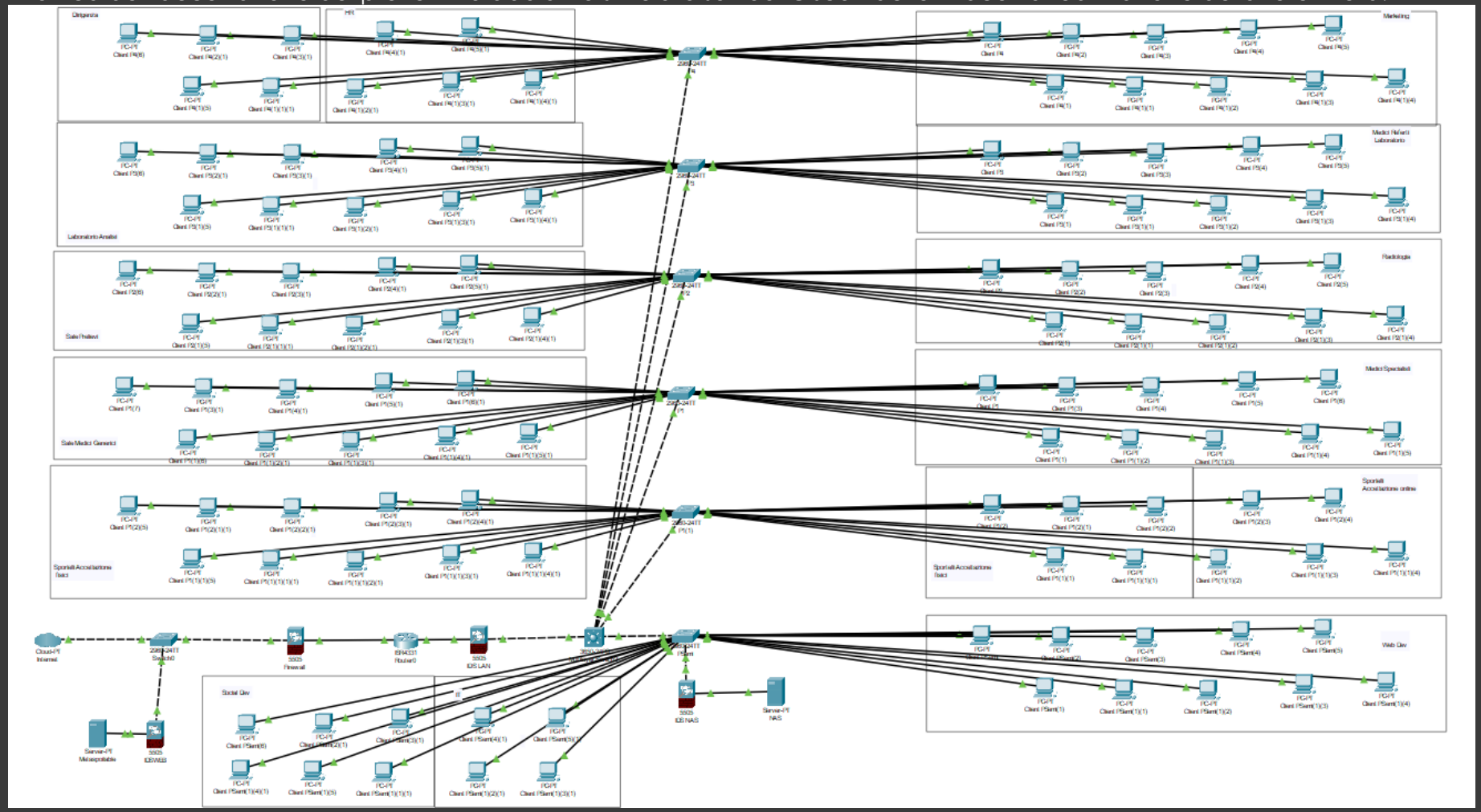
Sicurezza

La strategia di sicurezza prevede un approccio a più livelli:

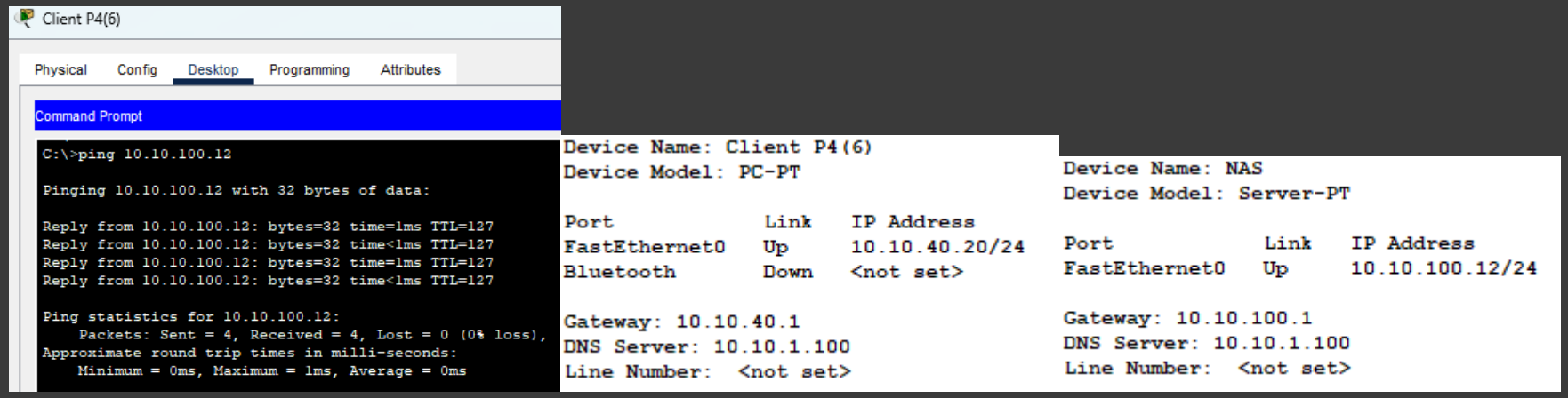
- Firewall Perimetrale e IDS/IPS (Netgate 6100 MAX con pfSense Plus): Proteggerà la rete da minacce esterne, controllerà il traffico in base a regole definite e analizzerà il traffico per rilevare e prevenire intrusioni.
- Segmentazione di Rete (Subnetting Fisico): La separazione fisica delle reti per piano, gestita dal routing del firewall, limiterà la propagazione di eventuali incidenti di sicurezza.
- IDS/IPS Interni (pfSense Plus): Monitoreranno il traffico tra le diverse subnet, rilevando attività sospette o malevole che potrebbero verificarsi all'interno della rete.
    - Un IDS implementato come da richiesta tra il firewall e la connessione internet
    - Un IDS implementato per l'ispezione dei dati di rete tra i vari piani
    - Un IDS implementato per il monitoraggio di traffico rete per il NAS
- Protezione dei NAS: Il server NAS sarà protetto da policy di accesso specifiche e monitorato dal sistema IDS/IPS

Il progetto su Cisco Packet Tracer

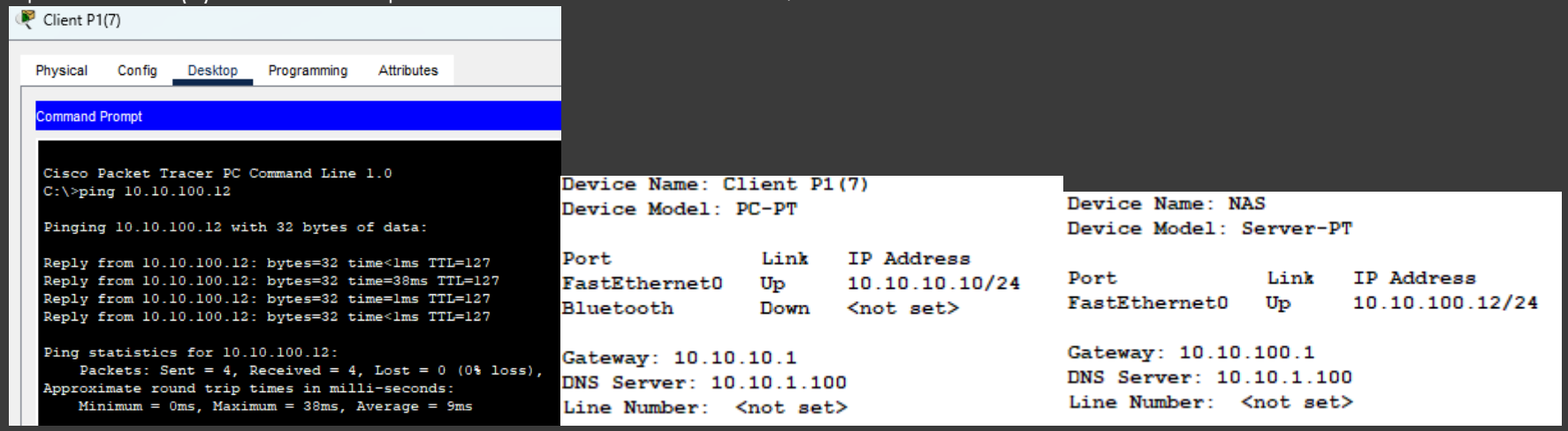
In ottica dell'accettazione del preventivo abbiamo simulato sul tool Cisco Packet Tracer la realizzazione della rete Theta.



Dopo aver eseguito a livello fisico le dovute connessioni e aver configurato lo switch LV3 per abilitare il servizio di DHCP, abbiamo effettuato un test per verificare che un qualsiasi computer possa raggiungere il NAS al quale, secondo le indicazioni di Theta, tutti devono avere accesso. Per fare questo abbiamo pingato con 2 computer situati su piani differenti il NAS:

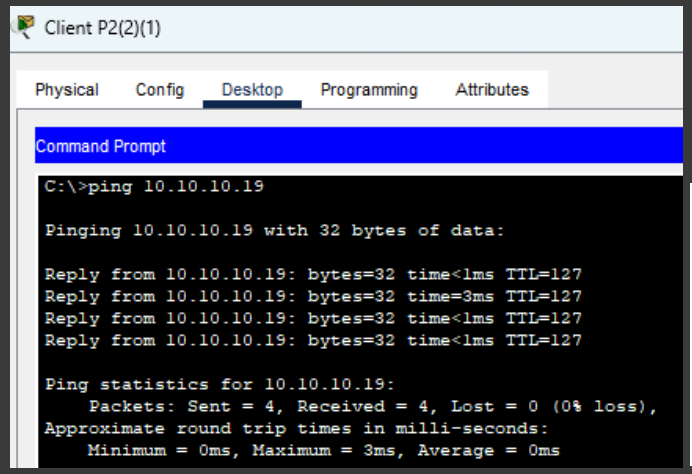


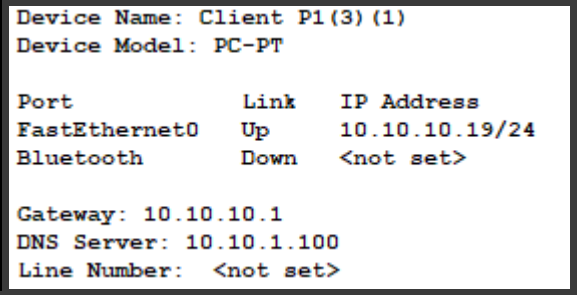
Il pc Client P4(6) ubicato al 4* piano riesce a contattare il NAS;



Il pc Client P1(7) ubicato al 1* piano riesce a contattare il NAS

Procediamo con un test per verificare se è possibile la comunicazione tra 2 computer ubicati su piani diversi.
Per il test abbiamo effettuato un ping dal pc Client P2(2)(1), ubicato al 2* piano al pc Client P1(3)(1) ubicato al 1* piano



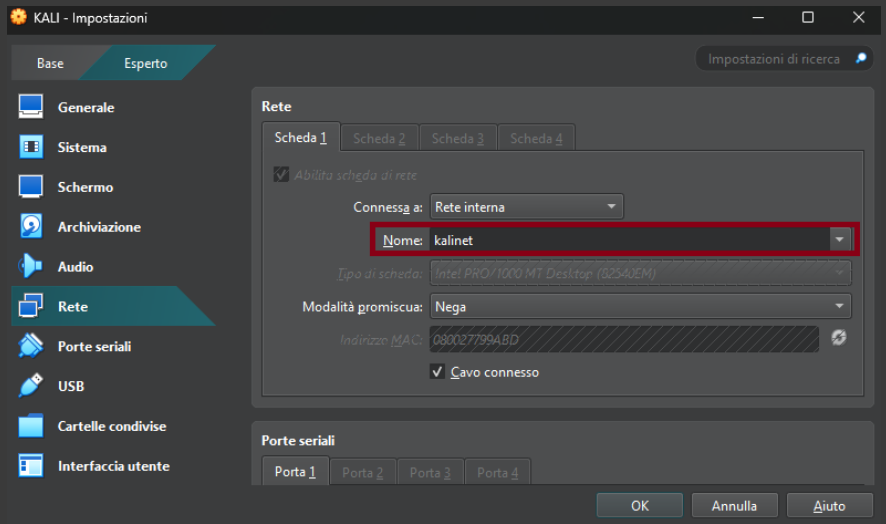


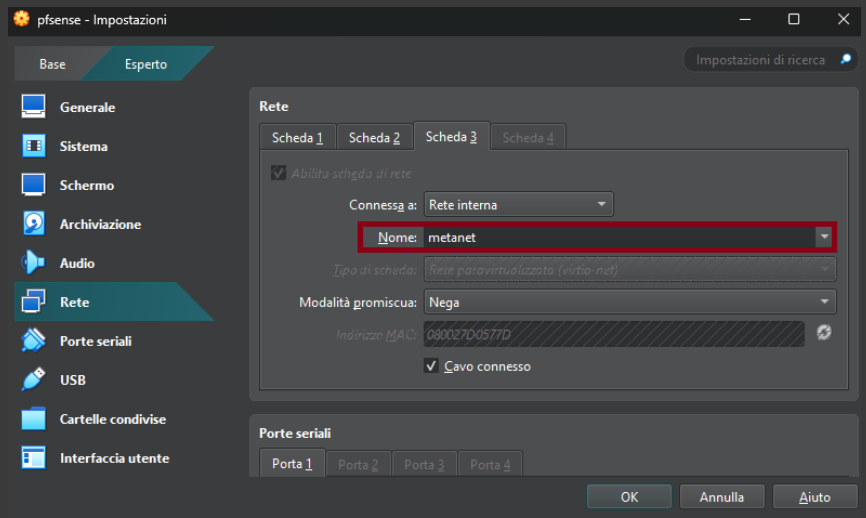
Abbiamo quindi testato che i vari pc possono comunicare tra loro con l'implementazione di una segmentazione di rete fatta tramite il subnetting fisico ai fini della sicurezza.

Test di rete

Per configurare la rete avremo bisogno di utilizzare un ambiente virtuale nel quale faremo comunicare il nostro client (nel nostro caso Kali) con il Web Server di Tetha (nel nostro caso simulato da META) e aggiungeremo anche un Firewall (una macchina virtuale pfSense).

Una volta configurate le schede di rete di Kali e pfSense affinché siano collegate tra di loro (come nella figura seguente) procediamo configurando anche le loro reti.





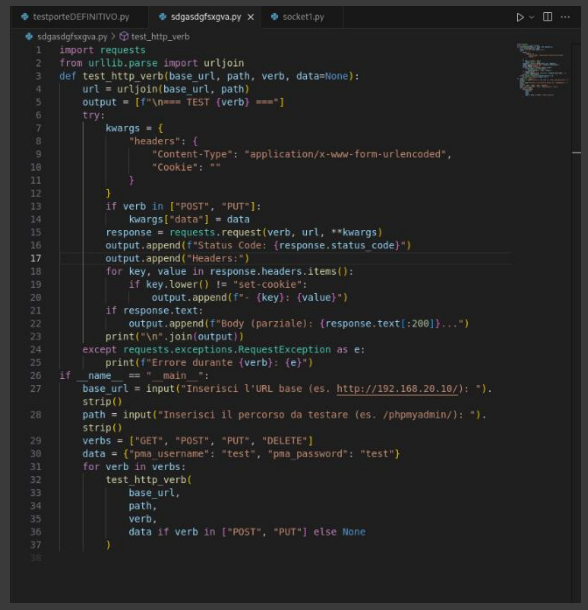
Test di rete Theta: programmi Python

Test delle richieste HTTP

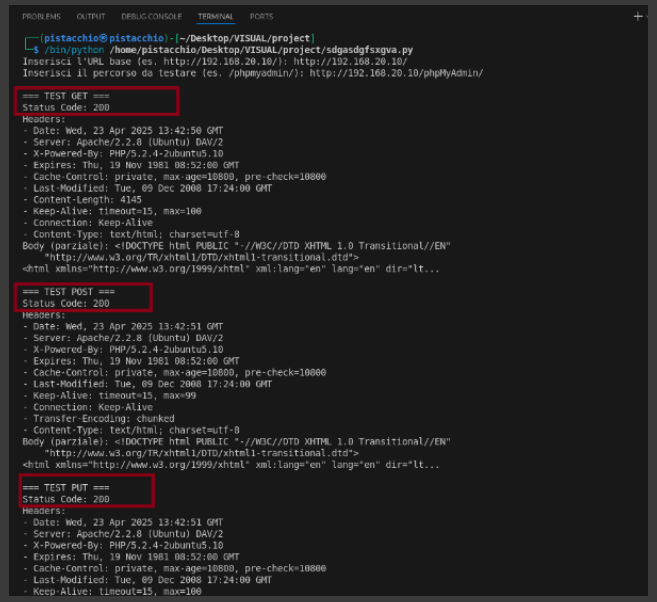
L'obiettivo Tetha da rispettare è di consegnare un progetto di rete sicura. Per poter verificare gli stati della rete andremo ad inviare delle specifiche richieste HTTP al web server utilizzando un tool appositamente creato dai nostri migliori corsisti di EPICODE guidati dal miglior insegnante che la scuola possa aver mai assunto, ovvero Henri Kapidani ;)

Per poterlo creare, questi famigerati migliori corsisti hanno deciso di usare la libreria "request" presente nella distro di Python per poter appunto inviare headers predefiniti. Ogni tipo di richiesta riporterà a schermo lo Status Code (codice di stato mostrato numericamente), headers HTTP (esclusi i cookie) e infine un Body parziale limitato a 200 caratteri per rendere la stampa del tool più breve.

A questo punto non resta che inserire l'URL del Web Server e il suo percorso per verificare le risposte.



richiesteHTTP.py



risposte con valore "200" ci segnalano l'esito positivo

Test delle porte

Procediamo con la creazione di un altro tool che ci permette di scansionare le porte in modo da capire quali sono aperte, quali chiuse e a quali servizi fanno riferimento fornendo dettagli sulla loro sicurezza.

All'interno del codice abbiamo fornito l'ip del server da scansionare e gli abbiamo dato un range di porte da controllare (nello specifico dalla 1 alla 1024) con un dizionario di porte note ai servizi conosciuti. Altra impostazione effettuata è un timeout da 0.5 nel caso in cui ci ritroviamo reti lente o connessioni bloccate.

Di seguito il tool e il risultato della scansione:

```
✅ Porta 21 è APERTA - Servizio: FTP - Sicurezza: Non Sicura
✅ Porta 22 è APERTA - Servizio: SSH - Sicurezza: Sicura
✅ Porta 23 è APERTA - Servizio: TELNET - Sicurezza: Non Sicura
❌ Porta 24 è CHIUSA
⚠️Porta 25 è APERTA - Servizio: SMTP - Sicurezza: Potenzialmente Rischiosa
❌ Porta 26 è CHIUSA
```

```
❌ Porta 511 è CHIUSA
🔍 Porta 512 è APERTA - Servizio sconosciuto
🔍 Porta 513 è APERTA - Servizio sconosciuto
🔍 Porta 514 è APERTA - Servizio sconosciuto
```

```
📋 RIEPILOGO SICUREZZA:

✅ Nessuna porta potenzialmente pericolosa rilevata.

🔐 Porte aperte considerate sicure:
1. Porta 21 - Servizio: FTP - Sicurezza: Non Sicura
2. Porta 22 - Servizio: SSH - Sicurezza: Sicura
3. Porta 23 - Servizio: TELNET - Sicurezza: Non Sicura
4. Porta 53 - Servizio: DNS - Sicurezza: Sicura se ben configurata
5. Porta 80 - Servizio: HTTP - Sicurezza: Non Sicura

⚠️Porte conosciute aperte e non sicure:
1. Porta 25 - Servizio: SMTP - Sicurezza: Potenzialmente Rischiosa
2. Porta 111 - Servizio: RPCBind - Sicurezza: Potenzialmente Rischiosa
3. Porta 139 - Servizio: NetBios - Sicurezza: Potenzialmente Rischiosa
4. Porta 445 - Servizio: SMB - Sicurezza: Potenzialmente Rischiosa

🔍 Servizi sconosciuti aperti:
1. Porta 512 - Servizio sconosciuto
2. Porta 513 - Servizio sconosciuto
3. Porta 514 - Servizio sconosciuto
```

Da questa schermata si evince il riepilogo finale che indica quali porte sono aperte con il grado di sicurezza.

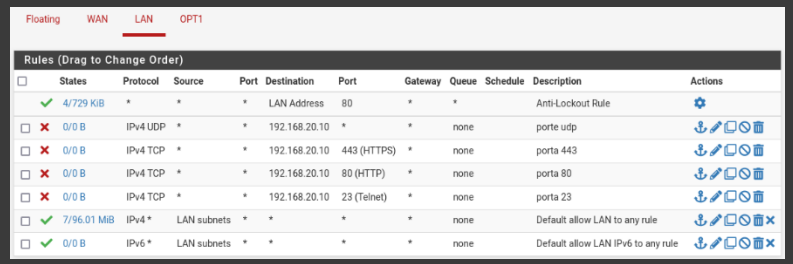
testporteDEFINITIVO.py | sdgasdgfsxgva.py | socket1.py

testporteDEFINITIVO.py > ...

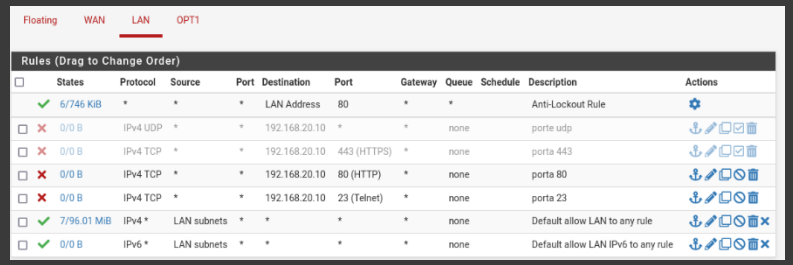
```python
import socket
target = "192.168.20.10"
start_port = 1
end_port = 1024
# porte_sicure_da_chiudere = list(range(1, 1024))

porte_note = {
    21: ("FTP", "Non Sicura"),
    22: ("SSH", "Sicura"),
    23: ("TELNET", "Non Sicura"),
    25: ("SMTP", "Potenzialmente Rischiosa"),
    53: ("DNS", "Sicura se ben configurata"),
    80: ("HTTP", "Non Sicura"),
    110: ("POP3", "Non Sicura"),
    111: ("RPCBind", "Potenzialmente Rischiosa"),
    139: ("NetBios", "Potenzialmente Rischiosa"),
    143: ("IMAP", "Non Sicura"),
    445: ("SMB", "Potenzialmente Rischiosa"),
    873: ("rsync", "Non Sicura"),
    993: ("IMAPS", "Sicura"),
    995: ("POP3S", "Sicura"),
    3389: ("RDP", "Non Sicura"),
    8080: ("HTTP Alternativo", "Non Sicura"),
}
def scan_ports(target, start_port, end_port):
    print(f"\n🔍 Scansione delle porte su {target} da {start_port} a
    {end_port}...\n")
    avvisi_sicurezza = []
    porte_sicure = []
    porte_conosciute_non_sicure = []
    porte_sconosciute = []
    for port in range(start_port, end_port + 1):
        sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        socket.setdefaulttimeout(0.5)
        result = sock.connect_ex((target, port))
        if result == 0:
            if port in porte_note:
                servizio, sicurezza = porte_note[port]
                if "Sicura" in sicurezza:
                    print(f"✅ Porta {port} è APERTA - Servizio: {servizio} -
                    Sicurezza: {sicurezza}")
                    porte_sicure.append((port, servizio, sicurezza))
                else:
                    print(f"⚠️ Porta {port} è APERTA - Servizio: {servizio} -
                    Sicurezza: {sicurezza}")
                    porte_conosciute_non_sicure.append((port, servizio,
                    sicurezza))
            else:
                print(f"🔍 Porta {port} è APERTA - Servizio sconosciuto")
                porte_sconosciute.append(port)
        else:
            print(f"❌ Porta {port} è CHIUSA")
        sock.close()
    print("\n📋 RIEPILOGO SICUREZZA:")
    if avvisi_sicurezza:
        print("\n⚠️  Porte aperte potenzialmente rischiose:")
        for idx, avviso in enumerate(avvisi_sicurezza, 1):
            print(f"{idx}. {avviso}")
    else:
        print("\n✅ Nessuna porta potenzialmente pericolosa rilevata.")
    if porte_sicure:
        print("\n🔐 Porte aperte considerate sicure:")
        for idx, (port, servizio, sicurezza) in enumerate(porte_sicure, 1):
            print(f"{idx}. Porta {port} - Servizio: {servizio} - Sicurezza:
            {sicurezza}")
    else:
        print("\n🚫 Nessuna porta sicura rilevata aperta.")
    if porte_conosciute_non_sicure:
        print("\n⚠️ Porte conosciute aperte e non sicure:")
        for idx, (port, servizio, sicurezza) in enumerate
        (porte_conosciute_non_sicure, 1):
            print(f"{idx}. Porta {port} - Servizio: {servizio} - Sicurezza:
            {sicurezza}")
    else:
        print("\n✅ Nessuna porta conosciuta e non sicura aperta.")
    if porte_sconosciute:
        print("\n🔍 Servizi sconosciuti aperti:")
        for idx, port in enumerate(porte_sconosciute, 1):
            print(f"{idx}. Porta {port} - Servizio sconosciuto")

scan_ports(target, start_port, end_port)
```

Visualizzate le porte aperte e le possibili minacce, chiudiamo le porte per difendere al meglio i vari reparti, scrivendo una regola all'interno del firewall emulato da pfSense:

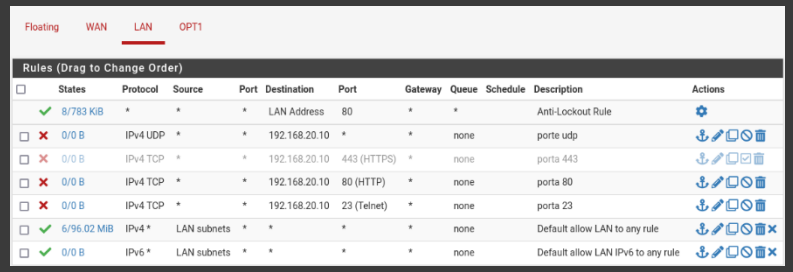
- Per il laboratorio di analisi, i referti medici, la radiologia e le sale prelievi si è deciso di chiudere le porte "23", "80", "443" e tutte le UDP perché inutilizzate dagli stessi:



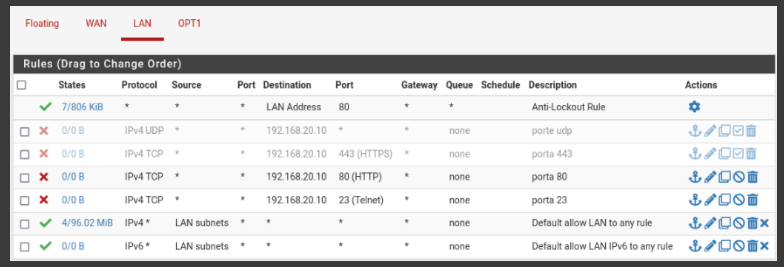
- Per l'HR, la dirigenza e il marketing andremo a chiudere le porte "23" e "80":



- Per gli sportelli e l'accettazione Web abbiamo chiuso le porte "23", "80" e tutte le UDP:



- Per le sale, medici generici e gli specialisti abbiamo chiuso le porte “23” e “80”:

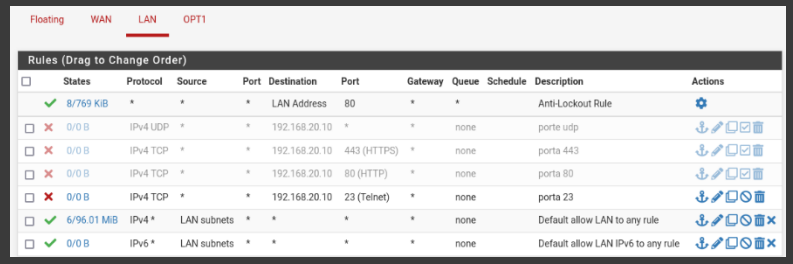


Floating WAN LAN OPT1

Rules (Drag to Change Order)

	States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
✓	7/806 KiB	*	*	*	LAN Address	80	*	*		Anti-Lockout Rule	
✗	0/0 B	IPv4 UDP	*	*	192.168.20.10	*	*	none		porte udp	
✗	0/0 B	IPv4 TCP	*	*	192.168.20.10	443 (HTTPS)	*	none		porta 443	
✗	0/0 B	IPv4 TCP	*	*	192.168.20.10	80 (HTTP)	*	none		porta 80	
✗	0/0 B	IPv4 TCP	*	*	192.168.20.10	23 (Telnet)	*	none		porta 23	
✓	4/96.02 MiB	IPv4 *	LAN subnets	*	*	*	*	none		Default allow LAN to any rule	
✓	0/0 B	IPv6 *	LAN subnets	*	*	*	*	none		Default allow LAN IPv6 to any rule	

- Per i servizi IT, Web Dev, Social Dev lasciamo tutte le porte aperte escluse la “23” ovvero la TELNET:



Floating WAN LAN OPT1

Rules (Drag to Change Order)

	States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
✓	8/769 KiB	*	*	*	LAN Address	80	*	*		Anti-Lockout Rule	
✗	0/0 B	IPv4 UDP	*	*	192.168.20.10	*	*	none		porte udp	
✗	0/0 B	IPv4 TCP	*	*	192.168.20.10	443 (HTTPS)	*	none		porta 443	
✗	0/0 B	IPv4 TCP	*	*	192.168.20.10	80 (HTTP)	*	none		porta 80	
✗	0/0 B	IPv4 TCP	*	*	192.168.20.10	23 (Telnet)	*	none		porta 23	
✓	6/96.01 MiB	IPv4 *	LAN subnets	*	*	*	*	none		Default allow LAN to any rule	
✓	0/0 B	IPv6 *	LAN subnets	*	*	*	*	none		Default allow LAN IPv6 to any rule	

Abbiamo chiuso tutte le porte non necessarie in base all'utilizzo che si intende fare del web server:

- Per il trasferimento di dati sul web si raccomanda di utilizzare la porta 443, per il protocollo HTTPS;
- L'utilizzo di SSL per la crittografia dei dati trasmessi tra Client-Server offre una maggiore protezione dei dati;
- L'utilizzo di certificati digitali garantisce l'autenticazione del server;
- Per la gestione del web server da remoto utilizzeremo la porta 22, per il protocollo SSH, assicurandoci di implementare alcune misure:
    - Configurare il Firewall per consentire l'accesso solamente agli utenti autorizzati
    - Implementare l'autenticazione a due fattori
    - Utilizzare chiavi SSH per l'autenticazione.

Test Socket di rete (Bonus)

Nella traccia era presente anche un obiettivo bonus che chiedeva di creare un tool che permettesse di catturare il socket di rete.
Al fine di eseguire quanto richiesto dalla traccia abbiamo creato due programmi:

- Con il primo programma vedremo come il tool avrà collegato con successo il socket all'ip statico del web server ricevendo un messaggio di conferma con il servizio GET (HTTP/1.1 200 OK) confermando il collegamento tra il Client principale e il web server.

socket2.py

```python
import socket
def capture_socket(host, port):
    try:
        s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        print(f"Socket creato con successo")
    except socket.error as err_msg:
        print(f"Creazione del socket fallita: {err_msg}")
        return None
    try:

        s.connect((host, port))
        print(f"Socket connesso a {host}:{port}")
    except socket.error as err_msg:
        print(f"Connessione del socket fallita: {err_msg}")
        s.close()
        return None
    return s
def main():
    """
    Funzione principale per eseguire la cattura del socket.
    """
    host = '192.168.20.10'
    port = 80
    captured_socket = capture_socket(host, port)
    if captured_socket:
        try:
            request = "GET / HTTP/1.1\r\nHost: 192.168.20.10/r\nConnection: close\r\n\r\n"
            captured_socket.sendall(request.encode())
            print("Richiesta inviata con successo")
            response = b''
            while True:
                data = captured_socket.recv(4096)
                if not data:
                    break
                response += data
            print("Risposta ricevuta:")
            print(response.decode('utf-8', errors='ignore'))
        except socket.error as err_msg:
            print(f"Errore durante l'invio/ricezione dati: {err_msg}")
        finally:
            captured_socket.close()
            print("Socket chiuso")
if __name__ == "__main__":
    main()
```

```
┌──(pistacchio㉿pistacchio)-[~/Desktop/VISUAL/project]
└─$ /bin/python /home/pistacchio/Desktop/VISUAL/project/socket2.py
Socket creato con successo
Socket connesso a 192.168.20.10:80
Richiesta inviata con successo
Risposta ricevuta:
HTTP/1.1 200 OK
Date: Wed, 23 Apr 2025 16:07:35 GMT
Server: Apache/2.2.8 (Ubuntu) DAV/2
X-Powered-By: PHP/5.2.4-2ubuntu5.10
Content-Length: 891
Connection: close
Content-Type: text/html

<html><head><title>Metasploitable2 - Linux</title></head><body>
<pre>


Warning: Never expose this VM to an untrusted network!

Contact: msfdev[at]metasploit.com

Login with msfadmin/msfadmin to get started


</pre>
<ul>
<li><a href="/twiki/">TWiki</a></li>
<li><a href="/phpMyAdmin/">phpMyAdmin</a></li>
<li><a href="/mutillidae/">Mutillidae</a></li>
<li><a href="/dvwa/">DVWA</a></li>
<li><a href="/dav/">WebDAV</a></li>
</ul>
</body>
</html>


Socket chiuso

┌──(pistacchio㉿pistacchio)-[~/Desktop/VISUAL/project]
└─$
```

- Con il secondo programma vedremo come il tool sarà in grado di mettersi in ascolto sulla porta 4444 del Client

socket1.py > ...

```python
import socket

SVR_ADDR="192.168.10.10"
SVR_PORT=4444
s=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind((SVR_ADDR, SVR_PORT))
s.listen(1)
print("Server aperto")
conn, addr = s.accept()
print("Connesso da: ", addr)
while True:
    data=conn.recv(1024)
    decoded=data.decode("utf-8")
    print("Messaggio ricevuto: ", decoded.strip())
    if decoded=="end\n": break
    conn.sendall(b"Messaggio ricevuto\n")
    print(data.decode("utf-8"), end="")
conn.close()
```

```
msfadmin@metasploitable:~$ netcat 192.168.10.10 4444
ciao
Messaggio ricevuto
end
msfadmin@metasploitable:~$ _
```

Lanciando il comando Netcat da Web Server verifichiamo se il nostro tool sia stato in grado di catturare la connessione.
Nel nostro Client infatti ci mostrerà la porta che ha utilizzato per potersi connettere: la 35816

```
┌──(pistacchio㉿pistacchio)-[~/Desktop/VISUAL/project]
└─$ /bin/python /home/pistacchio/Desktop/VISUAL/project/socket1.py
Server aperto
Connesso da:  ('192.168.20.10', 35816)
Messaggio ricevuto:  ciao
ciao
Messaggio ricevuto:  end
```

Conclusioni

Il progetto di rete proposto per Theta offre una soluzione completa e strutturata, basata su un'architettura di subnetting fisico per la segmentazione della rete. L'utilizzo di hardware affidabile e performante garantirà un'infrastruttura solida. La sicurezza sarà gestita dal firewall Netgate 6100 MAX con funzionalità IDS/IPS integrate e da ulteriori configurazioni software per la protezione interna.

L'offerta include l'installazione e la configurazione professionale degli apparati di rete da parte del nostro team tecnico. Seguiranno un collaudo finale per garantirne il corretto funzionamento e un'utile formazione di base per gli utenti finali sulla gestione di eventuali inconvenienti minori.
Restiamo a disposizione per ulteriori chiarimenti e per procedere con le fasi successive del progetto.

I prossimi passi suggeriti includono:

- Un incontro con Theta per discutere e approvare il progetto.
- La definizione precisa del piano di implementazione e delle tempistiche.
- L'avvio del processo di acquisto dell'hardware e del software.

In allegato alla relazione verrà fornito il preventivo, i programmi python creati al fine di eseguire i test della rete e la nostra presentazione aziendale che ci ha concesso di far partire i ponti con l'azienda Theta.