S4, Build Week 1 - Progetto // NetRaiders, Marzo 2024

Partecipanti:
Matteo Leoni
Rosario Giaimo
Claudio Maida
Gianmarco Mazzoni
Lorenzo Moro
Stefano Di Prospero

Scaletta del progetto

18 marzo 2024

- Overview del progetto
- Configurazione DVWA e Metasploit2
- Bozza su Packet Tracer
- Creazione di Vulnerability/Port Scanner
- Abilitazione metodi HTTP su porte necessarie

I moduli elencati sono delle bozze e potrebbero essere soggetti a modifiche.

19 marzo 2024

- Finalizzazione schema Packet Tracer
- Esecuzione test bruteforce
- Sviluppo bozza di codice a scopo di bruteforce

I moduli elencati sono delle bozze e potrebbero essere soggetti a modifiche.

20 marzo 2024

- Finalizzazione programma test di bruteforce
- Penetration Testing

21 marzo 2024

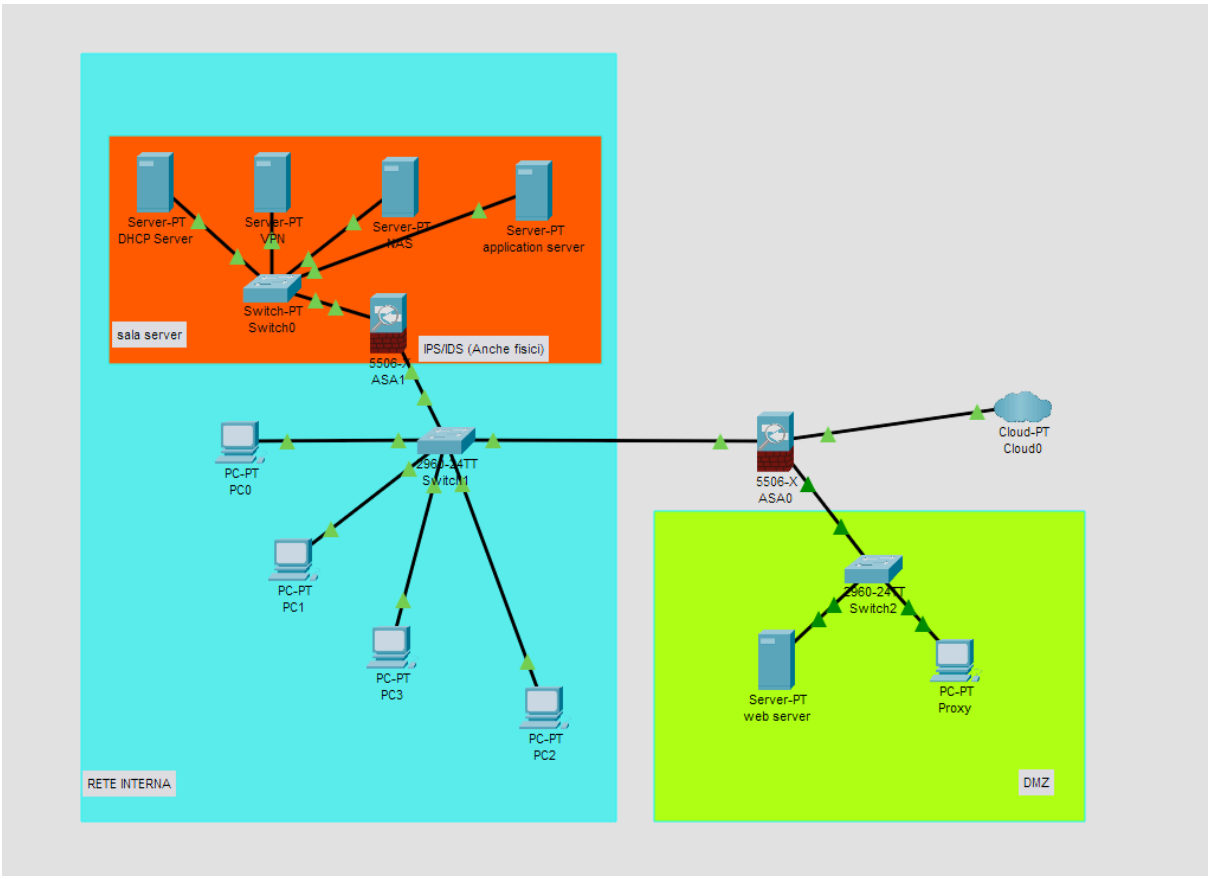
- Scrittura estensiva della documentazione e report di sicurezza
- Ottimizzazione degli applicativi

22 marzo 2024

- Revisione finale del progetto

18 marzo 2024

Ci siamo approcciati al progetto abbozzando uno schema su Cisco Packet Tracer, tenendo conto di diversi punti chiave, quali: presenza di Web Server che si connette ad Internet e un Application Server isolato accessibile solamente da Intranet.
Prima di tutto, l'Application Server dovrebbe essere posto dentro una sala dedicata, con accesso limitato, solamente agli addetti autorizzati, fisicamente e virtualmente.
Per continuare, il Web Server andrà invece piazzato all'esterno della Intranet, in un'apposita DMZ (Demilitarized Zone). Qui, il traffico in entrata e uscita passa per un proxy, per garantire un ulteriore livello di sicurezza.



Oggi abbiamo anche scritto un Port Scanner che utilizzeremo nei Penetration Test del 20 marzo. Sarà in futuro disponibile alla verifica nelle nostre repository.

```python
import socket

target = input("inserisci indirizzo IP da scansire:")
portrange = input("inserisci il port range da scansire (di default scansiona tutte le porte):")



if (portrange == ""):
    lowport = 0
    highport = 65535
else:
    lowport = int(portrange.split('-')[0])
    highport = int(portrange.split('-')[1])



print("scan host", target, "dalla porta", lowport, "alla porta", highport)

porte_aperte = []

for port in range(lowport, highport + 1):
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.settimeout(0.00000001)
    status = s.connect_ex((target, port))
    if status == 0:
        print(" ***  PORTA" , port, " APERTA ***")
        porte_aperte.append(port)
    else:
        print('- PORTA', port,  " CHIUSA")
    s.close()

if porte_aperte:
    print("\nRiepilogo porte aperte: ", porte_aperte)
else:
    print("\nnessuna porta aperta trovata")
```

Potrà essere implementato anche questo tool compilato oggi, a scopo di verifica dei metodi HTTP abilitati.

```python
import http.client

host = input("Inserire Host/IP del sistema target: ")
port = input("Inserire la porta del sistema target (default: 80): ")
path = input("Inserire path da controllare (default /): ").strip()

if (port == ""):
        port = 80
else:
        port = int(port)

if (path == ""):
        path = "/"

try:
        connection = http.client.HTTPConnection(host, port)
        connection.request("OPTIONS", path)
        response = connection.getresponse()
        print("Lo status e' : ", response.status)
        if response.status in (301,302,303,307,308):
                redirect = response.getheader("Location")
                print(f"Reindirizzamento ({response.status}) a: {redirect}")
        else:
                methods_enabled = response.getheader("allow")
                print("I metodi abilitati sono: ", methods_enabled)
        connection.close()
except ConnectionRefusedError:
        print("Connessione Fallita")
```

Nei prossimi giorni, prepareremo il tool per eseguire test di tipo Bruteforce sul sito, a scopo di rafforzare la sicurezza e informare i dipendenti su pratiche di sicurezza base.
Il codice è ancora in lavorazione e manca di commenti e della dovuta documentazione, che verrà scritta nei giorni a seguire, per essere quindi revisionata, il giorno prima della consegna.