Design e Sviluppo di un Tool per il Vulnerability Assessment di Protocolli Avanzati di Rete

Presentazione dell'Elaborato Finale

Laurea Triennale in Sicurezza dei Sistemi e delle Reti Informatiche

Veronica Falgiani (21191A)

21 ottobre 2025

UNIVERSITÀ DEGLI STUDI DI MILANO

Sommario

1 Introduzione

Vulnerability Scanner

1 Introduzione

I **Vulnerability Scanner** sono programmi che interagiscono con i dispositivi in rete inviando pacchetti e analizzando le risposte per individuare:

- **Vulnerabilità**
- **Configurazioni errate**
- **Versioni obsolete**

Criticità dello Stato dell'Arte

I vulnerability scanner citati precedentemente soffrono di alcune problematiche:

- **Complessità**
- **Dimensioni eccessive dell'applicativo**
- **Curva di apprendimento ripida**

Soluzione Proposta

1 Introduzione

L'**obiettivo** di questo elaborato consiste nello sviluppo di un vulnerability scanner che abbia come caratteristiche principali:

Modularità

Leggerezza

Semplicità d'uso

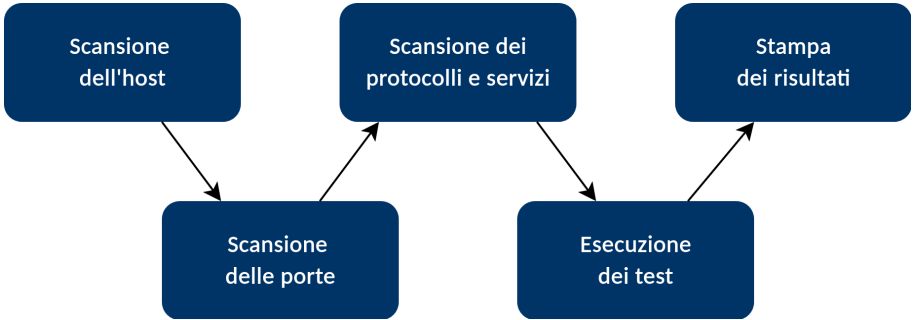
Sommario

2 Soluzione Proposta

Progettazione

2 Soluzione Proposta

Il funzionamento di un Vulnerability Scanner può essere riassunto in **cinque fasi**



Implementazione

2 Soluzione Proposta

Manuale di utilizzo

```
usage: main.py [-h] [-v] [-nt] [-hs HOST_SCAN] [-ps PORT_SCAN] ports host

Agent for Advanced Network Protocol Verification. This program needs sudo privileges to run.

positional arguments:
  ports                 Single port [x], multiple ports [x,y,z], port range [x:y] to scan or all ports [all]
  host                  Host to scan using ipv4 address

options:
  -h, --help            Show this help message and exit
  -v, --verbose         Increase output verbosity
  -nt, --no_tests       Scans the target for services but doesn't execute a vulnerability scan
  -hs, --host_scan HOST_SCAN
                        Host scan to execute: [p]ing, [s]yn, [a]ck, [u]dp (ping scan will be used by default)
  -ps, --port_scan PORT_SCAN
                        Port scan to execute: [c]onnect, [s]yn, [f]in, [n]ull, [x]mas, [u]dp (connect scan
                        will be used by default)
```

Scansione delle Porte

2 Soluzione Proposta

Scansione di tipo SYN

```
packet = IP(dst=self.ip) / TCP(dport=port, flags="S")
res = sr1(packet, timeout=3, verbose=0)

if res is None or (
    res.sprintf("%ICMP.type%") == 3
    and res.sprintf("%ICMP.code%") in [1, 2, 3, 9, 10,
        13]
):
    self.ports[port] = "filtered"

else:
    flag_res = res.sprintf("%TCP.flags%")

    if flag_res == "RA":
        self.ports[port] = "closed"
    elif flag_res == "SA":
        self.ports[port] = "open"
```

Funzionamento generico:

- **Invio di pacchetti** formattati in modo specifico
- **Analisi** dei contenuti delle **risposte**
- Definizione dello **stato delle porte**

Scansione di Protocolli e Servizi

2 Soluzione Proposta

Scansione del protocollo HTTP

```
try:
    # Tries to establish a connection using HTTP
    conn = HTTPConnection(url.netloc, timeout=3)
    conn.request("HEAD", url.path)
    res = conn.getresponse()

    banner = res.headers["server"]
    if banner is None:
        banner = "undefined"

    service["port"] = port
    service["protocol"] = "HTTP"
    service["service"] = banner

    self.services.append(service)

except Exception:
    pass
```

Funzionamento generico:

- **Connessione** alla porta utilizzando librerie specifiche dei protocolli
- Si tenta il **recupero del banner** del servizio
- Le **informazioni** ottenute vengono **salvate**

Un protocollo viene **individuato correttamente** solo quando la connessione alla porta tramite librerie **non genera errori**

File JSON di Test

2 Soluzione Proposta

Test per i protocolli

```
"vulns": {
  "ANONYMOUS LOGIN ENABLED" :{
    "description": "Anonymous login is enabled...",
    "send": "\n~~USER anonymous\n~~PASS\n",
    "recv": "230",
    "severity": "high"
  }
},
"login": {
  "send_str": "\n~~USER _username_\n~~PASS _password_
      \n",
  "recv_str": "230 Login successful."
},
"auth_vulns": {
  "BOUNCE ATTACK": {
    "description": "If not correctly configured...",
    "send": "PORT\n",
    "not_recv": "500",
    "severity": "medium"
  }
},
"serv_names": [ "vsftpd" ]
```

Test per i servizi

```
"vulns": {
  "BACKDOOR COMMAND EXECUTION": {
    "description": "Allows users to leverage a
        backdoor to make a command execution.",
    "send": "\n~~USER X:)\n~~PASS X\n~~id\n",
    "not_recv": "530",
    "severity": "high"
  }
},
"login": {},
"auth_vulns": {},
"vuln_serv_version": {
  "2.3.4": ["https://www.cve.org/CVERecord?id=CVE
      -2011-2523"],
  "3.0.2": ["https://www.cve.org/CVERecord?id=CVE
      -2015-1419"]
}
```

Esecuzione dei Test

2 Soluzione Proposta

```
sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
sock.settimeout(5)
sock.connect((ip, port))

for message in login_list:
    sock.send(message.encode())

for send in send_list:
    sock.send(send.encode())
    res = sock.recv(1024)

if (
    recv is not None
    and re.search(recv, res.decode())
    or not_recv is not None
    and not re.search(not_recv, res.decode())
):
    vuln = {}
    vuln["name"] = name
    vuln["service"] = service
    vuln["description"] = info["description"]
    vuln["severity"] = info["severity"]
    return vuln
```

Funzionamento:

- **Stabilire una connessione** normale o con SSL/TLS verso la porta
- **L'utente viene autenticato** se sono state fornite delle credenziali
- **Invio in sequenza dei comandi** recuperati dal file JSON
- **Analisi del messaggio ricevuto** per identificare la presenza o meno di vulnerabilità

Altri Test Svolti

2 Soluzione Proposta

Test autenticati

```
username = input()
password = getpass.getpass(f"{prot} - {service} password: ")
if username == "" and password == "":
    login_list = []
    return login_list
else:
    login_str = login["send_str"].replace("_username_", username)
    login_str = login_str.replace("_password_", password)

login_list = login_str.split("~~")
for message in login_list:
    sock.send(message.encode())
    res = sock.recv(1024)
```

Test versione del servizio

```
for version, cve in vuln_serv_version.items():
    if version in service:
        results.unsafe_ver = True
        results.unsafe_ver_cve = cve
```

Test versione SSL/TLS

```
if not ("TLSv1.3" in service or "TLSv1.2" in service):
    results.unsafe_tls = True
```

Sommario

3 Risultati

Risultati Intermedi

3 Risultati

Utilizzando il flag `-v` o `--verbose` vengono stampati **risultati intermedi**...

```
--- Checking host ---
Host is up
--- Checking ports ---
PORT        STATUS
21          open
22          open
23          open
--- Checking protocols and services ---
PORT        PROTOCOL        SERVICE
21          FTP             (vsFTPd 2.3.4)
22          SSH             SSH-2.0-OpenSSH_4.7p1 Debian-8ubuntu1
23          TELNET          undefined
```

...e lo **stato attuale dell'esecuzione**

```
--- Checking protocols and services ---
Scanning 80 for FTP
```

```
--- Testing protocols and services ---
Scanning 21 with FTP - (vsFTPd 2.3.4) using BACKDOOR COMMAND EXECUTION [1/1]
```

Risultati TXT

3 Risultati

```
##### RESULTS FOR 192.168.100.175 #####

PORT      PROTOCOL     SERVICE
---------------------------

21          FTP              (vsFTPd 2.3.4)
|
| ---------------- VERSION CHECK ----------------
|\--- THIS SERVICE VERSION IS VULNERABLE AND NEEDS TO BE UPDATED!
|      reference:
|      - CVE-2011-2523: https://www.cve.org/CVERecord?id=CVE-2011-2523
|
| ---------------- VULNERABILITIES ----------------
|\--- BACKDOOR COMMAND EXECUTION
|      description: Allows users to leverage a backdoor to make a command execution.
|      severity: high
|
| ---------------- AUTHENTICATED VULNERABILITIES ----------------
|\--- BOUNCE ATTACK
|      description: If not correctly configured the PORT command can use the victim machine to request
       access to port indirectly. This can be used to scan hosts ports discretely.
|      severity: medium
```

Risultati JSON

3 Risultati


```
    "port": 21,
    "protocol": "FTP",
    "service": "(vsFTPd 2.3.4)",
    "unsafe_version": true,
    "unsafe_version_cve": [
        "https://www.cve.org/CVERecord?id=CVE-2011-2523"
    ],
    "vulnerabilities": [
        {
            "name": "BACKDOOR COMMAND EXECUTION",
            "service": "(vsFTPd 2.3.4)",
            "description": "Allows users to leverage a backdoor to make a command execution.",
            "severity": "high"
        }
    ],
    "auth_vulnerabilities": [
        {
            "name": "BOUNCE ATTACK",
            "service": "(vsFTPd 2.3.4)",
            "description": "If not correctly configured the PORT command can use the victim machine to request
                  access to port indirectly. This can be used to scan hosts ports discretely.",
            "severity": "medium"
        }
```

Risultati HTML

3 Risultati

FTP-21
SSH-22
TELNET-23
SMTP-25
DNS-53
HTTP-80
SMB-139
SMB-445

Port 21 - FTP - (vsFTPd 2.3.4)

SERVICE VERSION

This service version is vulnerable, an update is mandatory!

Reference CVE:

- CVE-2011-2523

VULNERABILITIES

- HIGH: 2
  - **ANONYMOUS LOGIN ENABLED**
    description: Anonymous login is enabled, everyone can access the service
  - **BACKDOOR COMMAND EXECUTION**
    description: Allows users to leverage a backdoor to make a command execution.
- MEDIUM: 0
- LOW: 0

AUTHENTICATED VULNERABILITIES

- HIGH: 0
- MEDIUM: 1
  - **BOUNCE ATTACK**
    description: If not correctly configured the PORT command can use the victim machine to request access to port indirectly. This can be used to scan hosts ports discretely.
- LOW: 0

Sommario

4 Conclusioni

Conclusioni e Sviluppi Futuri

4 Conclusioni

Obiettivi raggiunti:

- **Modularità** del codice e dei test
- **Leggerezza** del programma
- **Semplicità** nell'utilizzo e nella scrittura dei test

Sviluppi futuri:

- Scrittura di test **automatizzata**
- **Ampliamento** dei protocolli supportati
- Scansione di **più host contemporaneamente**
- Esecuzione **automatica e periodica**
- Evasione dei **firewall**

Design e Sviluppo di un Tool per il Vulnerability Assessment di Protocolli Avanzati di Rete

Grazie per l'attenzione