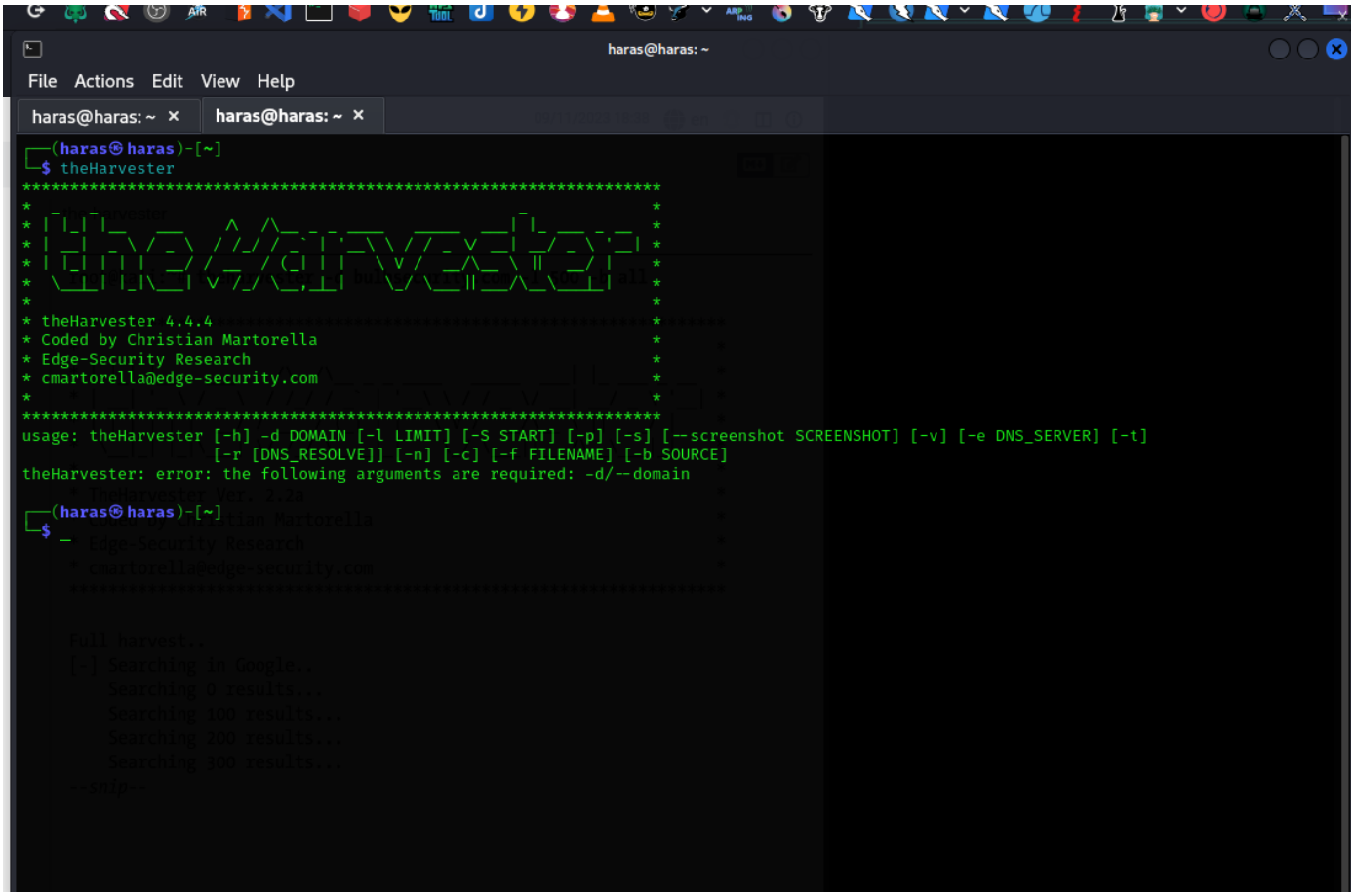
the harvester

the harvester

```
root@kali:~# theharvester -d bulbsecurity.com -l 500 -b all

*******************************************************************
*                                                                 *
* | |_| |__   ___    /\  /\__ _ _ ____   _____  ___| |_ ___ _ __  *
* | __| '_ \ / _ \  / /_/ / _` | '__\ \ / / _ \/ __| __/ _ \ '__| *
* | |_| | | |  __/ / __  / (_| | |   \ V /  __/\__ \ ||  __/ |    *
*  \__|_| |_|\___| \/ /_/ \__,_|_|    \_/ \___||___/\__\___|_|    *
*                                                                 *
* TheHarvester Ver. 2.2a                                          *
* Coded by Christian Martorella                                   *
* Edge-Security Research                                          *
* cmartorella@edge-security.com                                   *
*******************************************************************

Full harvest..
[-] Searching in Google..
    Searching 0 results...
    Searching 100 results...
    Searching 200 results...
    Searching 300 results...
--snip--
```



python logon.

which systems are active and which software we can talk to.

Manual Port Scanning

For example, in the previous chapter we saw that exploiting the MS08-067 vulnerability can be an easy win for attackers and pentesters alike. To use this exploit, we need to find a Windows 2000, XP, or 2003 box with an SMB server that is missing the MS08-067 Microsoft patch available on the network. We can get a good idea about the network-based attack surface by mapping the network range and querying systems for listening ports.

We can do this manually by connecting to ports with a tool such as telnet or Netcat and recording the results. Let's use Netcat to connect to the Windows XP machine on port 25, the default port for the Simple Mail Transfer Protocol (SMTP).

```
root@kali:~# nc -vv 192.168.20.10 25
nc: 192.168.20.10 (192.168.20.10) 25 [smtp]❶ open
nc: using stream socket
nc: using buffer size 8192
nc: read 66 bytes from remote
220 bookxp SMTP Server SLmail 5.5.0.4433 Ready
ESMTP spoken here
nc: wrote 66 bytes to local
```

```
root@kali:~# nmap -sS 192.168.20.10-12 -oA booknmap
Starting Nmap 6.40 ( http://nmap.org ) at 2015-12-18 07:28 EST
Nmap scan report for 192.168.20.10
Host is up (0.00056s latency).
Not shown: 991 closed ports
```

A Version Scan

Our SYN scan was stealthy, but it didn't tell us much about the software that is actually running on the listening ports. Compared to the detailed version information we got by connecting to port 25 with Netcat, the SYN scan's results are a bit lackluster. We can use a full TCP scan (`nmap -sT`) or go a step further and use Nmap's version scan (`nmap -sV`) to get more data. With the version scan shown in Listing 5-7, Nmap completes the connection and then attempts to determine what software is running and, if possible, the version, using techniques such as banner grabbing.

```
root@kali:~# nmap -sV 192.168.20.10-12 -oA bookversionnmap

Starting Nmap 6.40 ( http://nmap.org ) at 2015-12-18 08:29 EST
Nmap scan report for 192.168.20.10
Host is up (0.00046s latency).
Not shown: 991 closed ports
PORT     STATE SERVICE      VERSION
21/tcp   open  ftp          FileZilla ftpd 0.9.32 beta
25/tcp   open  smtp         SLmail smtpd 5.5.0.4433
79/tcp   open  finger       SLMail fingerd
80/tcp   open  http         Apache httpd 2.2.12 ((Win32) DAV/2 mod_ssl/2.2.12 OpenSSL/0.9.8k
                              mod_autoindex_color PHP/5.3.0 mod_perl/2.0.4 Perl/v5.10.0)
106/tcp  open  pop3pw       SLMail pop3pw
110/tcp  open  pop3         BVRP Software SLMAIL pop3d
135/tcp  open  msrpc        Microsoft Windows RPC
139/tcp  open  netbios-ssn
443/tcp  open  ssl/http     Apache httpd 2.2.12 ((Win32) DAV/2 mod_ssl/2.2.12 OpenSSL/0.9.8k
                              mod_autoindex_color PHP/5.3.0 mod_perl/2.0.4 Perl/v5.10.0)
445/tcp  open  microsoft-ds Microsoft Windows XP microsoft-ds
1025/tcp open  msrpc        Microsoft Windows RPC
```

In a UDP scan (`-sU`), Nmap sends a UDP packet to a port. Depending on the port, the packet sent is protocol specific. If it receives a response, the port is considered open. If the port is closed, Nmap will receive an ICMP Port Unreachable message. If Nmap receives no response whatsoever, then either the port is open and the program listening does not respond to Nmap's query, or the traffic is being filtered. Thus, Nmap is not always able to distinguish between an open UDP port and one that is filtered by a firewall. See Listing 5-8 for a UDP scan example.

```
root@kali:~# nmap -sU 192.168.20.10-12 -oA bookudp

Starting Nmap 6.40 ( http://nmap.org ) at 2015-12-18 08:39 EST
Stats: 0:11:43 elapsed; 0 hosts completed (3 up), 3 undergoing UDP Scan
UDP Scan Timing: About 89.42% done; ETC: 08:52 (0:01:23 remaining)
Nmap scan report for 192.168.20.10
Host is up (0.00027s latency).
Not shown: 990 closed ports
PORT     STATE         SERVICE
69/udp   open|filtered tftp ❶
123/udp  open          ntp
135/udp  open          msrpc
137/udp  open          netbios-ns
138/udp  open|filtered netbios-dgm
445/udp  open|filtered microsoft-ds
500/udp  open|filtered isakmp
1026/udp open          win-rpc
1065/udp open|filtered syscomlan
1900/udp open|filtered upnp
MAC Address: 00:0C:29:A5:C1:24 (VMware)
```

Listing 5-9: Running an Nmap scan on a specific port

Sure enough, when we tell Nmap to scan 3232, it returns `open`, which shows that this port is worth checking out, in addition to the default Nmap scanned ports. However, if we try to probe the port a bit more aggressively with a version scan (see Listing 5-10), the service listening on the port crashes, as shown in Figure 5-8.

A good rule of thumb is to specify ports 1 through 65535 on your pentests, just to make sure there's nothing listening on those other "uninteresting" ports.

```
root@kali:~# nmap -p 3232 -sV 192.168.20.10
Starting Nmap 6.40 ( http://nmap.org ) at 2015-04-28 10:19 EDT
Nmap scan report for 192.168.20.10
Host is up (0.00031s latency).
PORT     STATE SERVICE VERSION
3232/tcp open  unknown
1 service unrecognized despite returning data❶. If you know the service/
version, please submit the following fingerprint at http://www.insecure.org/
cgi-bin/servicefp-submit.cgi : ❷
SF-Port3232-TCP:V=6.25%I=7%D=4/28%Time=517D2FFC%P=i686-pc-linux-gnu%r(GetR
SF:equest,B8,"HTTP/1\.1\x20200\x20OK\r\nServer:\x20Zervit\x200\.4\r\n❸X-Pow
```

Technical Report

This section of the report offers technical details of the test. It should include the following:

Introduction An inventory of details such as scope, contacts, and so on.

Information gathering Details of the findings in the information-gathering phase. Of particular interest is the client's Internet footprint.

Vulnerability assessment Details of the findings of the vulnerability-analysis phase of the test.

Exploitation/vulnerability verification Details of the findings from the exploitation phase of the test.

Post exploitation Details of the findings of the post-exploitation phase of the test.

Risk/exposure A quantitative description of the risk discovered. This section estimates the loss if the identified vulnerabilities were exploited by an attacker.

Conclusion A final overview of the test.

How to do it...

To understand how ARP discovery works, we will start by using Scapy to craft custom packets that will allow us to identify hosts on the LAN using ARP. To begin using Scapy in Kali Linux, enter the `scapy` command from the terminal. You can then use the `display()` function to see the default configurations for any ARP object created in Scapy in the following manner:

```
root@KaliLinux:~# scapy
Welcome to Scapy (2.2.0)
>>> ARP().display()
###[ ARP ]###
  hwtype= 0x1
  ptype= 0x800
  hwlen= 6
  plen= 4
  op= who-has
  hwsrc= 00:0c:29:fd:01:05
  psrc= 172.16.36.232
  hwdst= 00:00:00:00:00:00
  pdst= 0.0.0.0
```

Notice that both the IP and MAC source addresses are automatically configured to the values associated with the host on which Scapy is being run. Except in the case that you are spoofing an alternate source address, these values will never have to be changed for any Scapy objects. The default opcode value for ARP is automatically set to `who-has`, which designates that the packet will be requesting an IP and MAC association. In this case, the only value we need to supply is the destination IP address. To do this, we can create an object using the ARP function by setting it equal to a variable. The name of the variable is irrelevant (in the example provided, the variable name, `arp_request`, is used). Have a look at the following commands:

```
>>> arp_request = ARP()
>>> arp_request.pdst = "172.16.36.135"
>>> arp_request.display()
###[ ARP ]###
  hwtype= 0x1
  ptype= 0x800
  hwlen= 6
  plen= 4
  op= who-has
```

```
hwsrc= 00:0c:29:65:fc:d2
psrc= 172.16.36.132
hwdst= 00:00:00:00:00:00
pdst= 172.16.36.135
```

Notice that the `display()` function can also be applied to the created ARP object to verify that the configuration values have been updated. For this exercise, use a destination IP address that corresponds to a live machine in your lab network. The `sr1()` function can then be used to send the request over the wire and return the first response:

```
>>> sr1(arp_request)
Begin emission:
.....................................*Finished to send 1 packets.


Received 39 packets, got 1 answers, remaining 0 packets
<ARP  hwtype=0x1 ptype=0x800 hwlen=6 plen=4 op=is-at
hwsrc=00:0c:29:3d:84:32 psrc=172.16.36.135 hwdst=00:0c:29:65:fc:d2
pdst=172.16.36.132 |<Padding  load='\x00\x00\x00\x00\x00\x00\x00\x00\x00\
x00\x00\x00\x00\x00\x00\x00\x00\x00' |>>
```

Alternatively, you can perform the same task by calling the function directly and passing any special configurations as arguments to it, as shown in the following command. This can avoid the clutter of using unnecessary variables and can also allow the completion of the entire task in a single line of code:

```
>>> sr1(ARP(pdst="172.16.36.135"))
Begin emission:
.........................*Finished to send 1 packets.


Received 26 packets, got 1 answers, remaining 0 packets
<ARP  hwtype=0x1 ptype=0x800 hwlen=6 plen=4 op=is-at
hwsrc=00:0c:29:3d:84:32 psrc=172.16.36.135 hwdst=00:0c:29:65:fc:d2
pdst=172.16.36.132 |<Padding  load='\x00\x00\x00\x00\x00\x00\x00\x00\x00\
x00\x00\x00\x00\x00\x00\x00\x00\x00' |>>
```

Notice that in each of these cases, a response is returned, indicating that the IP address of `172.16.36.135` is at the MAC address of `00:0C:29:3D:84:32`. If you perform the same task, but instead, assign a destination IP address that does not correspond to a live host on your lab network, you will not receive any response, and the function will continue to analyze the incoming traffic on the local interface indefinitely.

recon ng

Recon-ng 5.1 Cheat Sheet

https://github.com/lanmaster53/recon-ng
@BHInfoSecurity && @BBhacKing
https://www.blackhillsinfosec.com/
2019-11-21

4.x	5.x	Note
<nothing>	`marketplace install all`	installs the modules
`workspaces add`	`workspaces create`	makes a new workspace
`workspaces select x`	`workspaces load x`	moves into workspace x context
`add companies`	`db insert companies`	manually load seed data
`load $module`	`modules load $module`	"use" is gone
`ls`	`!ls` or `shell ls`	OS commands require a prefix
`set verbosity 2`	`options set VERBOSITY 2`	tab-completes from any case, but execution requires ALL CAPS
`set source query ...`	`options set SOURCE query ...`	case-sensitive 'SOURCE'
`show dashboard`	`dashboard show`	bare "dashboard" works, too

Object	Verbs
workspaces	`create` | `load` | `list` | `remove`
db	`insert` | `delete` | `query` | `schema` | `notes`
modules	`load` | `reload` | `search`
options	`set` | `unset` | `list` *`option names are ALL_CAPS`*
marketplace	`install` | `remove` | `info` | `refresh` | `search`
keys	`add` | `remove` | `list`
script	`record [filename]` | `execute [filename]` | `stop` | `status`
snapshots	`take` | `remove` | `list` | `load [snapshot_name]`
spool	`start [filename]` | `stop` | `status`

Handy Command	Why Handy
`marketplace install all`	Because there are no modules installed by default.
`show companies`	Shows the whole table, incluing the "rowid" which you sometimes need.
`db query select * from companies`	Basic query syntax. Gives same result as "show companies" except omits the "rowid" column.
`dashboard`	Shows which modules have been run in the current workspace.
`modules search $regex`	Gives a list of modules matching $regex. "modules" is plural, and $regex is a literal regex
`shell ls`	runs operating system's "ls" command in your current working directory
`!ls`	Same as above
`marketplace search`	Shows modules available in the marketplace (there is no "marketplace list" command)
`marketplace search $regex`	Shows marketplace modules matching $regex
`marketplace info $string`	Shows details of modules in the marketplace that contain $string

Things to Know
Read the wiki. It's not long. The inline 'help' command is far more helpful if you've read the wiki.
The framework warns you at startup with a red "key not set" message for specific modules whose API keys are not present.
Spool files are written to your current working directory in the filename you pass to 'spool record [filename]'
Script files are written to your current working directory in the filename you pass to 'script record [filename]'
Snapshots are stored in the .recon-ng/workspaces/$current_workspace directory and named automatically with embedded timestamp
The word for "get rid of" varies by object type among {delete, remove, unset}
The word for "create" varies by object type among {create, insert, add, install, record, take, start}

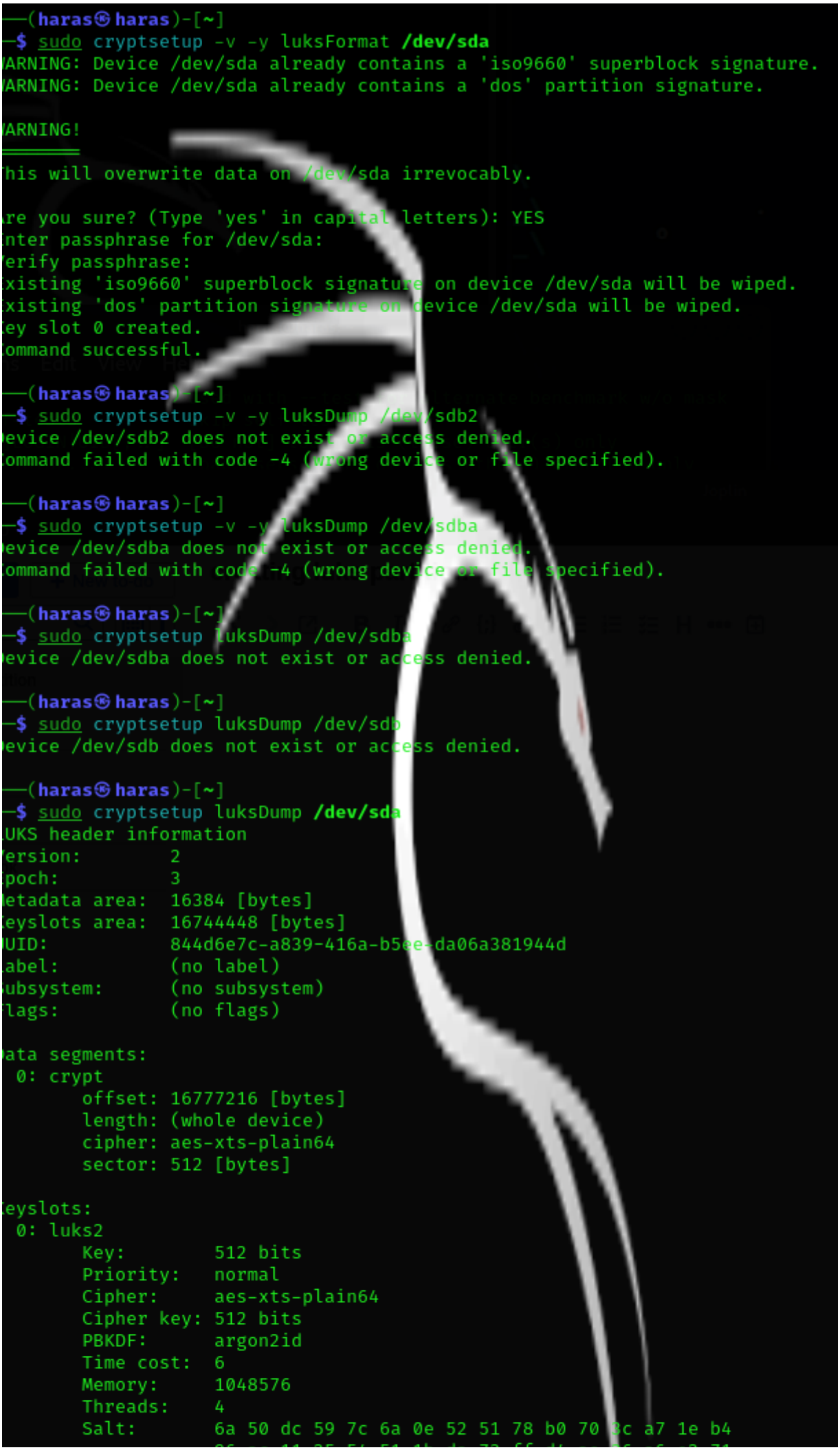
General Informations

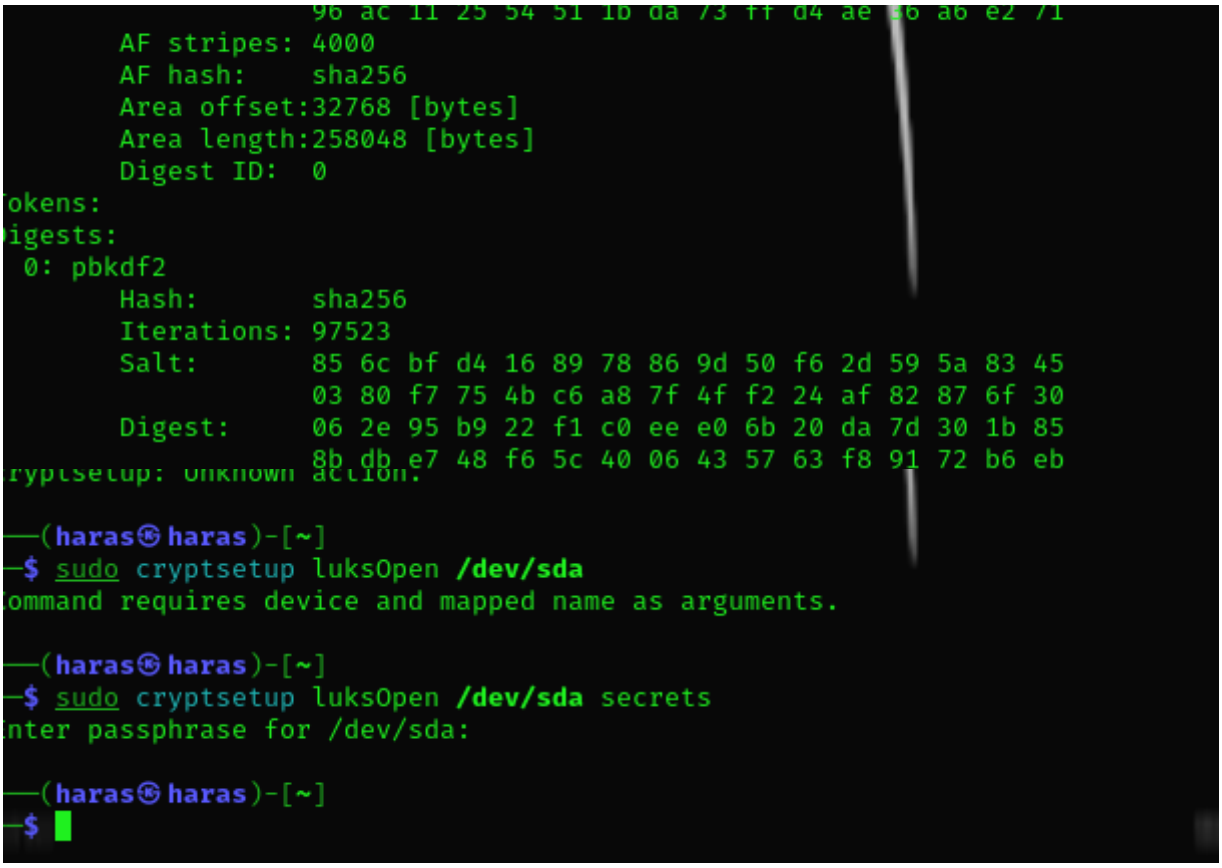
```
https://bitbucket.org/LaNMaSteR53/recon-ng/wiki/Usage%20G

# Its possible to add your own modules
# Reconnaissance → Usage of open sources (passive)
# Discovery → More active, packets sent to target

# After loading a module, the context of the framework ch
# These commands and options are unique to the module. (h
```

Usage

```
# Add API keys
> keys add shodan_api <key>
> keys list


# Modules
> show modules
> search <string>
> use recon/domains-hosts/builtwith
> show infos
> set <param> <value>


# Workspaces
> show workspaces
> workspaces list
> workspaces add Name
> workspaces select <ws>

# Add companies and domains to a schema
> show schema
> add compagnies company
> add domains domain.com
> show domains

# Contacts
> show contacts
```

Frameworks


(haras㉿haras)-[~]
$ sudo cryptsetup -v -y luksFormat /dev/sda
ARNING: Device /dev/sda already contains a 'iso9660' superblock signature.
ARNING: Device /dev/sda already contains a 'dos' partition signature.
ARNING!
his will overwrite data on /dev/sda irrevocably.
re you sure? (Type 'yes' in capital letters): YES
nter passphrase for /dev/sda:
erify passphrase:
xisting 'iso9660' superblock signature on device /dev/sda will be wiped.
xisting 'dos' partition signature on device /dev/sda will be wiped.
ey slot 0 created.
ommand successful.
(haras㉿haras)-[~]
$ sudo cryptsetup -v -y luksDump /dev/sdb2
evice /dev/sdb2 does not exist or access denied.
ommand failed with code -4 (wrong device or file specified).
(haras㉿haras)-[~]
$ sudo cryptsetup -v -y luksDump /dev/sdba
evice /dev/sdba does not exist or access denied.
ommand failed with code -4 (wrong device or file specified).
(haras㉿haras)-[~]
$ sudo cryptsetup luksDump /dev/sdba
evice /dev/sdba does not exist or access denied.
(haras㉿haras)-[~]
$ sudo cryptsetup luksDump /dev/sdb
evice /dev/sdb does not exist or access denied.
(haras㉿haras)-[~]
$ sudo cryptsetup luksDump /dev/sda
UKS header information
ersion: 2
poch: 3
etadata area: 16384 [bytes]
eyslots area: 16744448 [bytes]
UID: 844d6e7c-a839-416a-b5ee-da06a381944d
abel: (no label)
ubsystem: (no subsystem)
lags: (no flags)
ata segments:
0: crypt
offset: 16777216 [bytes]
length: (whole device)
cipher: aes-xts-plain64
sector: 512 [bytes]
eyslots:
0: luks2
Key: 512 bits
Priority: normal
Cipher: aes-xts-plain64
Cipher key: 512 bits
PBKDF: argon2id
Time cost: 6
Memory: 1048576
Threads: 4
Salt: 6a 50 dc 59 7c 6a 0e 52 51 78 b0 70 8c a7 1e b4


96 ac 11 25 54 51 1b da 73 ff d4 ae e6 a6 e2 71
AF stripes: 4000
AF hash: sha256
Area offset:32768 [bytes]
Area length:258048 [bytes]
Digest ID: 0
okens:
igests:
0: pbkdf2
Hash: sha256
Iterations: 97523
Salt: 85 6c bf d4 16 89 78 86 9d 50 f6 2d 59 5a 83 45
03 80 f7 75 4b c6 a8 7f 4f f2 24 af 82 87 6f 30
Digest: 06 2e 95 b9 22 f1 c0 ee e0 6b 20 da 7d 30 1b 85
8b db e7 48 f6 5c 40 06 43 57 63 f8 91 72 b6 eb
ryptsetup: Unknown action.
(haras㉿haras)-[~]
$ sudo cryptsetup luksOpen /dev/sda
ommand requires device and mapped name as arguments.
(haras㉿haras)-[~]
$ sudo cryptsetup luksOpen /dev/sda secrets
nter passphrase for /dev/sda:
(haras㉿haras)-[~]
$

Configuring to mount automatically --obtain the uuid. sudo cryptsetup luksUUID /dev/sdb1

copy the uuid ND PASTE IT INTO THE /ETC/CRYPTTAB FILE.

YOU WILL PASTE IT TWICE.

LUKS-_____ UUID=_____--

edit the etc/fstab file adding the last line in the file for your new ncrypted partiction note that you have to use luks- followed by the UUID number

/dev/mapper/luks-_____ /secrets xfs
defaults 0 0

TO OPEN A LUKS DEVICE

sudo cryptsetup luksOpen /dev/sdb secret
Enter passphrase for /dev/sdb:

use the mount command to see if successfull
mout |grep 'secrets'

**sdk tools used in binary :::::

**—(haras㉿haras)-[~/Downloads/sdk/cmdline-tools/bin]

exploit links

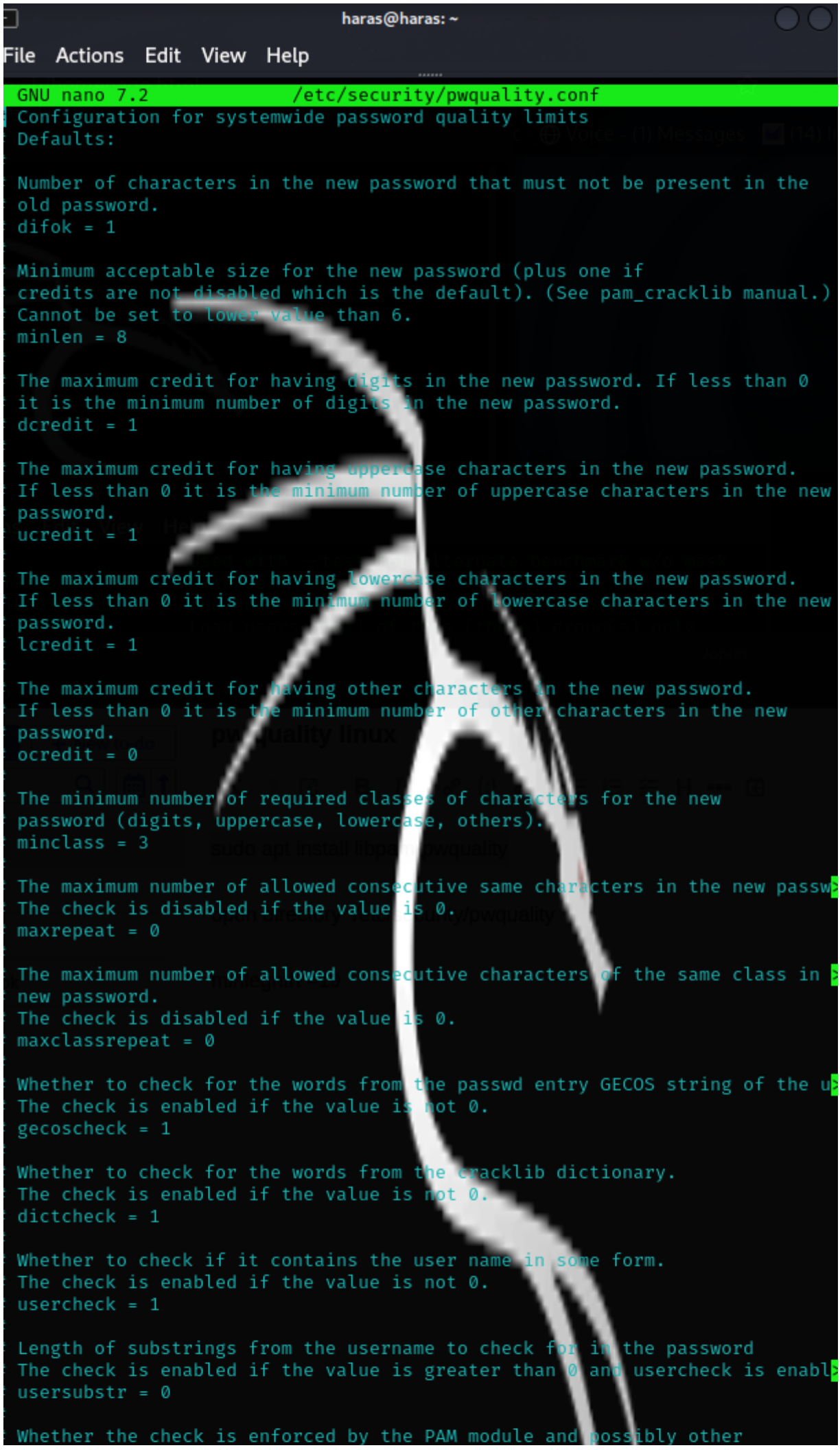
is also missing this patch, attempt to trigger the vulnerability, and develop a working exploit. But developing exploits by hand takes both time and skill, and the window of opportunity for your pentest may be closing.

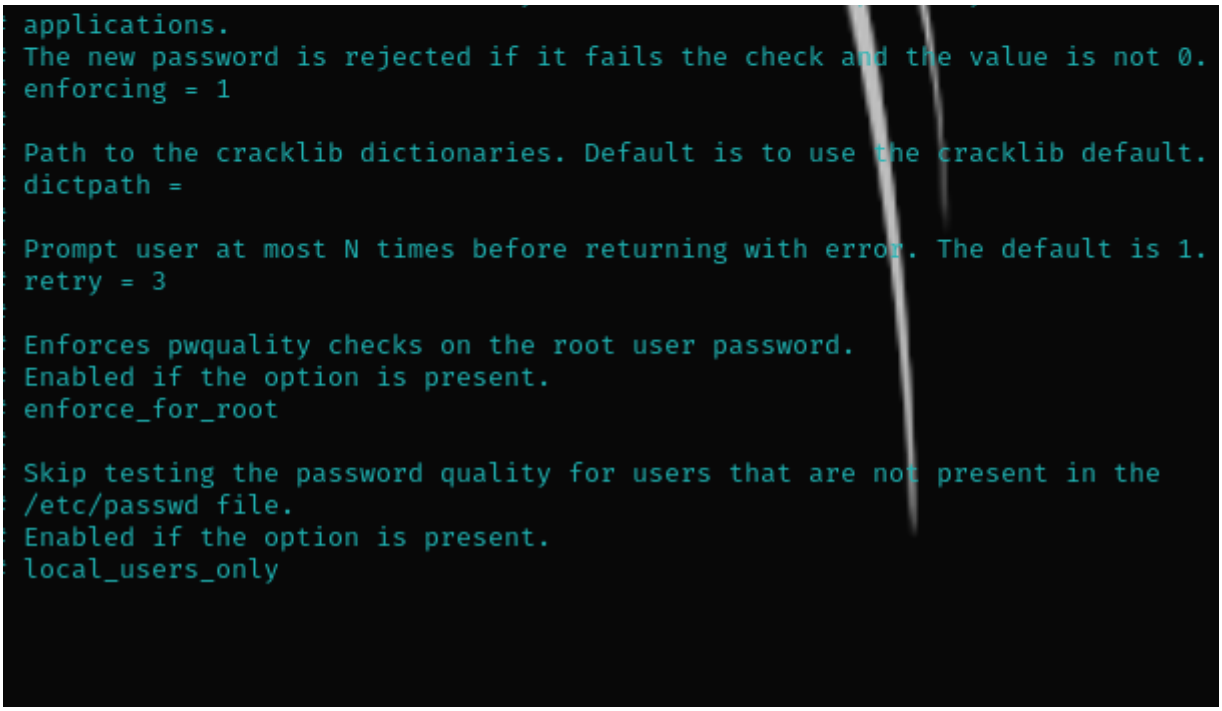
You could instead search for code that exploits this vulnerability on the Internet. Sites like Packet Storm Security (*http://www.packetstormsecurity.com/*), SecurityFocus (*http://www.securityfocus.com/*), and Exploit Database (*http://www.exploit-db.com/*) provide repositories of known exploit code. But be forewarned: Not all public exploit code does what it claims to do. Some exploit code may destroy the target system or even attack your system

sudo apt install libpam-pwquality

open directory /etc/security/pwquality

minlegnth =19


haras@haras: ~
File Actions Edit View Help
GNU nano 7.2 /etc/security/pwquality.conf
Configuration for systemwide password quality limits
Defaults:
Number of characters in the new password that must not be present in the
old password.
difok = 1
Minimum acceptable size for the new password (plus one if
credits are not disabled which is the default). (See pam_cracklib manual.)
Cannot be set to lower value than 6.
minlen = 8
The maximum credit for having digits in the new password. If less than 0
it is the minimum number of digits in the new password.
dcredit = 1
The maximum credit for having uppercase characters in the new password.
If less than 0 it is the minimum number of uppercase characters in the new
password.
ucredit = 1
The maximum credit for having lowercase characters in the new password.
If less than 0 it is the minimum number of lowercase characters in the new
password.
lcredit = 1
The maximum credit for having other characters in the new password.
If less than 0 it is the minimum number of other characters in the new
password.
ocredit = 0
The minimum number of required classes of characters for the new
password (digits, uppercase, lowercase, others).
minclass = 3
The maximum number of allowed consecutive same characters in the new passw
The check is disabled if the value is 0.
maxrepeat = 0
The maximum number of allowed consecutive characters of the same class in
new password.
The check is disabled if the value is 0.
maxclassrepeat = 0
Whether to check for the words from the passwd entry GECOS string of the u
The check is enabled if the value is not 0.
gecoscheck = 1
Whether to check for the words from the cracklib dictionary.
The check is enabled if the value is not 0.
dictcheck = 1
Whether to check if it contains the user name in some form.
The check is enabled if the value is not 0.
usercheck = 1
Length of substrings from the username to check for in the password
The check is enabled if the value is greater than 0 and usercheck is enabl
usersubstr = 0
Whether the check is enforced by the PAM module and possibly other



MISC and tricks

```
https://www.notsosecure.com/one-rule-to-rule-them-all/
```

```
# MAX POWER
# force the CUDA GPU interface, optimize for <32 char passwords and set the workload to insane (-w 4).
# It is supposed to make the computer unusable during the cracking process
# Finnally, use both the GPU and CPU to handle the cracking
--force -O -w 4 --opencl-device-types 1,2
```

Wrapcat - Automating hashcat commands

```
https://twitter.com/Haax9_/status/1340354639464722434?s=20
https://github.com/Haax9/Wrapcat

$ python wrapcat.py -m 1000 -f HASH_FILE.txt -p POT_FILE.txt --full --save
```

Attack modes

```
-a 0 # Straight : hash dict
-a 1 # Combination : hash dict dict
-a 3 # Bruteforce : hash mask
-a 6 # Hybrid wordlist + mask : hash dict mask
-a 7 # Hybrid mask + wordlist : hash mask dict
```

Charsets

```
?l # Lowercase a-z
?u # Uppercase A-Z
?d # Decimals
?h # Hex using lowercase chars
?H # Hex using uppercase chars
?s # Special chars
?a # All (l,u,d,s)
?b # Binary
```

Options

```
-m # Hash type
-a # Attack mode
-r # Rules file
-V # Version
--status # Keep screen updated
-b # Benchmark
--runtime # Abort after X seconds
--session [text] # Set session name
--restore # Restore/Resume session
-o filename # Output to filename
--username # Ignore username field in a hash
--potfile-disable # Ignore potfile and do not write
--potfile-path # Set a potfile path
-d # Specify an OpenCL Device
-D # Specify an OpenCL Device Type
-l # List OpenCL Devices & Types
-O # Optimized Kernel, Passwords <32 chars
-i # Increment (bruteforce)
--increment-min # Start increment at X chars
--increment-max # Stop increment at X chars
```

Examples

```
# Benchmark MD4 hashes
hashcat -b -m 900

# Create a hashcat session to hash Kerberos 5 tickets using wordlist
hashcat -m 13100 -a 0 --session crackin1 hashes.txt wordlist.txt -o output.pot

# Crack MD5 hashes using all char in 7 char passwords
hashcat -m 0 -a 3 -i hashes.txt ?a?a?a?a?a?a?a -o output.pot

# Crack SHA1 by using wordlist with 2 char at the end
hashcat -m 100 -a 6 hashes.txt wordlist.txt ?a?a -o output.pot

# Crack WinZip hash using mask (Summer2018!)
hashcat -m 13600 -a 3 hashes.txt ?u?l?l?l?l?l?l?d?d?d?d! -o output.pot

# Crack MD5 hashes using dictionnary and rules
hashcat -a 0 -m 0 example0.hash example.dict -r rules/best64.rules

# Crack MD5 using combinator function with 2 dictionnaries
hashcat -a 1 -m 0 example0.hash example.dict example.dict

# Cracking NTLM hashes
hashcat64 -m 1000 -a 0 -w 4 --force --opencl-device-types 1,2 -O d:\hashsample.hash "d:\WORDLISTS\realuniq.lst" -r OneRuleToRuleThemAll.rule

# Cracking hashes from kerberoasting
hashcat64 -m 13100 -a 0 -w 4 --force --opencl-device-types 1,2 -O d:\krb5tgs.hash d:\WORDLISTS\realhuman_phill.txt -r OneRuleToRuleThemAll.rule
```

```
# Cracking hashes from kerberoasting
hashcat64 -m 13100 -a 0 -w 4 --force --opencl-device-types 1,2 -O d:\krb5tgs.hash d:\WORDLISTS\realhuman_phill.txt -r OneRuleToRuleThemAll.rule
```

```
# You can use hashcat to perform combined attacks
# For example by using wordlist + mask + rules
hashcat -a 6 -m 0 prenoms.txt ?d?d?d?d -r rules/yourule.rule

# Single rule used to uppercase first letter --> Marie2018
hashcat -a 6 -m 0 prenoms.txt ?d?d?d?d -j 'c'
```

Scenario - Cracking large files (eg NTDS.dit)

```
# Start by making a specific potfile and cracked files (clean environment)
# - domain_ntds.dit
# - domain_ntds_potfile.pot

# Goal is to run many different instances with different settings, so each one have
# to be quite quick

# You can generate wordlist using CeWL
# It usually works pretty well
cewl -d 5 -m 4 -w OUTFILE -v URL
cewl -d 5 -m 4 -w OUTFILE -o -v URL

# With some basic dictionnary cracking (use known wordlists)
# rockyou, hibp, crackstation, richelieu, kaonashi, french and english
.\hashcat64.exe -m 1000 hashs.txt --potfile-path potfile.pot -a 0 rockyou.txt --force -O

# Then start to use wordlists + masks + simple rule
# For special chars, you can use a custom charset : "?!%$&#-_@+=* "
# Multiple tests, multiples masks and multiples wordlists (including generated ones)
.\hashcat64.exe -m 1000 hashs.txt -a 6 .\french\* '?d?d?d?d' -j c --increment --force -O
.\hashcat64.exe -m 1000 hashs.txt -a 6 .\french\* -1 .\charsets\custom.chr '?1' -j c --force -O
.\hashcat64.exe -m 1000 hashs.txt -a 6 .\french\* -1 .\charsets\custom.chr '?d?1' -j c --force -O
.\hashcat64.exe -m 1000 hashs.txt -a 6 .\french\* -1 .\charsets\custom.chr '?d?d?1' -j c --force -O
.\hashcat64.exe -m 1000 hashs.txt -a 6 .\french\* -1 .\charsets\custom.chr '?d?d?d?1' -j c --force -O
.\hashcat64.exe -m 1000 hashs.txt -a 6 .\french\* -1 .\charsets\custom.chr '?d?d?d?d?1' -j c --force -O
.\hashcat64.exe -m 1000 hashs.txt -a 6 CEWL_WORDLIST.txt -1 .\charsets\custom.chr '?d?d?d?d?1' -j c --force -O
.\hashcat64.exe ...

# Same commands and behavior but using mask after the tested word (mode 7)
.\hashcat64.exe -m 1000 hashs.txt -a 7 '?d?d?d?d' .\french\* -j c --increment --force -O

# Then, wordlists + complex rules
# Once again run against multiple wordlists (including generated ones)
# Kaonashi and OneRuleToRuleThemAll can produce maaaaaassive cracking time
.\hashcat64.exe -m 1000 hashs.txt --potfile-path potfile.pot -a 0 french.txt -r .rules\best64.rule --force -O
.\hashcat64.exe -m 1000 hashs.txt --potfile-path potfile.pot -a 0 french.txt -r .rules\OneRuleToRuleThemAll.rule --force -O
.\hashcat64.exe -m 1000 hashs.txt --potfile-path potfile.pot -a 0 french.txt -r .rules\best64.rule --force -O
.\hashcat64.exe ...
```

```
# Same commands and behavior but using mask after the tested word (mode 7)
.\hashcat64.exe -m 1000 hashs.txt -a 7 '?d?d?d?d' .\french\* -j c --increment --force -O

# Then, wordlists + complex rules
# Once again run against multiple wordlists (including generated ones)
# Kaonashi and OneRuleToRuleThemAll can produce maaaaaassive cracking time
.\hashcat64.exe -m 1000 hashs.txt --potfile-path potfile.pot -a 0 french.txt -r .rules\best64.rule --force -O
.\hashcat64.exe -m 1000 hashs.txt --potfile-path potfile.pot -a 0 french.txt -r .rules\OneRuleToRuleThemAll.rule --force -O
.\hashcat64.exe -m 1000 hashs.txt --potfile-path potfile.pot -a 0 french.txt -r .rules\best64.rule --force -O
.\hashcat64.exe ...

# Then smart bruteforce using masks (custom charset can be usefull too)
# Can be quite long, depending on the mask. Many little tests with different masks
# Knowing for example that password is min 8 char long, only 8+ masks
# Play by incrementing or decrementing char vs decimal (you can also use specific charset to reduce time)
.\hashcat64.exe -m 1000 hashs.txt --potfile-path potfile.pot -a 3 '?u?l?l?l?d?d?d?d' --force -O
.\hashcat64.exe -m 1000 hashs.txt --potfile-path potfile.pot -a 3 '?u?l?l?l?l?d?d?d' --force -O
.\hashcat64.exe -m 1000 hashs.txt --potfile-path potfile.pot -a 3 '?u?l?l?l?l?l?d?d' --force -O
.\hashcat64.exe -m 1000 hashs.txt --potfile-path potfile.pot -a 3 -1 .\charset\custom '?u?l?l?l?l?l?d?1' --force -O
.\hashcat64.exe ...

# Then increment mask size and play again
# Can be longer for 9 char and above.. Up to you to decide which masks and how long you wanna wait
.\hashcat64.exe -m 1000 hashs.txt --potfile-path potfile.pot -a 3 '?u?l?l?l?d?d?d?d?d' --force -O
.\hashcat64.exe -m 1000 hashs.txt --potfile-path potfile.pot -a 3 '?u?l?l?l?l?d?d?d?d' --force -O
.\hashcat64.exe -m 1000 hashs.txt --potfile-path potfile.pot -a 3 '?u?l?l?l?l?l?d?d?d' --force -O
.\hashcat64.exe ...

# If you have few hashes and small/medium wordlist, you can use random rules
# And make several loops
.\hashcat64.exe -m 1000 hashs.txt --potfile-path potfile.pot -a 0 wl.txt -g 1000000  --force -O -w 3

# You can use combination attacks
# For example, combine different names, or combine names with dates.. Then apply masks
# Directly using hashcat
.\hashcat64.exe -m 1000 hashs.txt --potfile-path potfile.pot -a 1 wordlist1.txt wordlist2.txt --force -O
# Or in memory feeding, it allows you to use rules but not masks
.\combinator.exe wordlist1.txt wordlist2.txt | .\hashcat64.exe -m 1000 hashs.txt --potfile-path potfile.pot -a 0 -rules .\rules\best64.rule --force
# Or create the wordlist before and use it
.\combinator.exe wordlist1.txt wordlist2.txt
.\hashcat64.exe -m 1000 hashs.txt --potfile-path potfile.pot -a 6 combinedwordlist.txt '?d?d?d?d' -j c --increment --force -O

# Finally use your already cracked passwords to build a new wordlist
.\hashcat64.exe -m 1000 hashs.txt --potfile-path potfile.pot --show | %{$_.split(':')[1]} > cracked.txt
.\hashcat64.exe -m 1000 hashs.txt -a 6 cracked.txt '?d?d?d?d' -j c --increment --force -O
.\hashcat64.exe -m 1000 hashs.txt -a 0 cracked.txt -r .rules\OneRuleToRuleThemAll.rule --force -O

# You can also checks the target in popular leaks to find some password
# Then try reuse or rules on them
```

Passcracking Hash & Files

Options

```
hashcat (v6.2.6) starting in help mode

Usage: hashcat [options]... hash|hashfile|hccapxfile [dictionary|mask|directory]...

- [ Options ] -

 Options Short / Long           | Type | Description                                                | Example
================================+======+============================================================+=======================
 -m, --hash-type                | Num  | Hash-type, references below (otherwise autodetect)         | -m 1000
 -a, --attack-mode              | Num  | Attack-mode, see references below                          | -a 3
 -V, --version                  |      | Print version                                              |
 -h, --help                     |      | Print help                                                 |
     --quiet                    |      | Suppress output                                            |
     --hex-charset              |      | Assume charset is given in hex                             |
     --hex-salt                 |      | Assume salt is given in hex                                |
     --hex-wordlist             |      | Assume words in wordlist are given in hex                  |
     --force                    |      | Ignore warnings                                            |
     --deprecated-check-disable |      | Enable deprecated plugins                                  |
     --status                   |      | Enable automatic update of the status screen               |
     --status-json              |      | Enable JSON format for status output                       |
     --status-timer             | Num  | Sets seconds between status screen updates to X            | --status-timer=1
     --stdin-timeout-abort      | Num  | Abort if there is no input from stdin for X seconds        | --stdin-timeout-abort=3
     --machine-readable         |      | Display the status view in a machine-readable format       |
     --keep-guessing            |      | Keep guessing the hash after it has been cracked           |
     --self-test-disable        |      | Disable self-test functionality on startup                 |
     --loopback                 |      | Add new plains to induct directory                         |
     --markov-hcstat2           | File | Specify hcstat2 file to use                                | --markov-hcstat2=my.hcs
     --markov-disable           |      | Disables markov-chains, emulates classic brute-force       |
     --markov-classic           |      | Enables classic markov-chains, no per-position             |
     --markov-inverse           |      | Enables inverse markov-chains, no per-position             |
 -t, --markov-threshold         | Num  | Threshold X when to stop accepting new markov-chains       | -t 50
     --runtime                  | Num  | Abort session after X seconds of runtime                   | --runtime=10
     --session                  | Str  | Define specific session name                               | --session=mysession
     --restore                  |      | Restore session from --session                             |
     --restore-disable          |      | Do not write restore file                                  |
     --restore-file-path        | File | Specific path to restore file                              | --restore-file-path=x.re
 -o, --outfile                  | File | Define outfile for recovered hash                          | -o outfile.txt
     --outfile-format           | Str  | Outfile format to use, separated with commas               | --outfile-format=1,3
     --outfile-autohex-disable  |      | Disable the use of $HEX[] in output plains                 |
     --outfile-check-timer      | Num  | Sets seconds between outfile checks to X                   | --outfile-check-timer=3
     --wordlist-autohex-disable |      | Disable the conversion of $HEX[] from the wordlist         |
 -p, --separator                | Char | Separator char for hashlists and outfile                   | -p :
     --stdout                   |      | Do not crack a hash, instead print candidates only         |
     --show                     |      | Compare hashlist with potfile; show cracked hashes         |
     --left                     |      | Compare hashlist with potfile; show uncracked hashes       |
     --username                 |      | Enable ignoring of usernames in hashfile                   |
     --remove                   |      | Enable removal of hashes once they are cracked             |
     --remove-timer             | Num  | Update input hash file each X seconds                      | --remove-timer=30
     --potfile-disable          |      | Do not write potfile                                       |
     --potfile-path             | File | Specific path to potfile                                   | --potfile-path=my.pot
     --encoding-from            | Code | Force internal wordlist encoding from X                    | --encoding-from=iso-885
     --encoding-to              | Code | Force internal wordlist encoding to X                      | --encoding-to=utf-32le
     --debug-mode               | Num  | Defines the debug mode (hybrid only by using rules)        | --debug-mode=4
     --debug-file               | File | Output file for debugging rules                            | --debug-file=good.log
     --induction-dir            | Dir  | Specify the induction directory to use for loopback        | --induction=inducts
     --outfile-check-dir        | Dir  | Specify the outfile directory to monitor for plains        | --outfile-check-dir=x
     --logfile-disable          |      | Disable the logfile                                        |
     --hccapx-message-pair      | Num  | Load only message pairs from hccapx matching X             | --hccapx-message-pair=2
     --nonce-error-corrections  | Num  | The BF size range to replace AP's nonce last bytes         | --nonce-error-correctio
     --keyboard-layout-mapping  | File | Keyboard layout mapping table for special hash-modes       | --keyb=german.hckmap
     --truecrypt-keyfiles       | File | Keyfiles to use, separated with commas                     | --truecrypt-keyf=x.png
     --veracrypt-keyfiles       | File | Keyfiles to use, separated with commas                     | --veracrypt-keyf=x.txt
     --veracrypt-pim-start      | Num  | VeraCrypt personal iterations multiplier start             | --veracrypt-pim-start=4
     --veracrypt-pim-stop       | Num  | VeraCrypt personal iterations multiplier stop              | --veracrypt-pim-stop=50
 -b, --benchmark                |      | Run benchmark of selected hash-modes                       |
     --benchmark-all            |      | Run benchmark of all hash-modes (requires -b)              |
     --speed-only               |      | Return expected speed of the attack, then quit             |
     --progress-only            |      | Return ideal progress step size and time to process        |
 -c, --segment-size             | Num  | Sets size in MB to cache from the wordfile to X            | -c 32
     --bitmap-min               | Num  | Sets minimum bits allowed for bitmaps to X                 | --bitmap-min=24
     --bitmap-max               | Num  | Sets maximum bits allowed for bitmaps to X                 | --bitmap-max=24
     --cpu-affinity             | Str  | Locks to CPU devices, separated with commas                | --cpu-affinity=1,2,3
     --hook-threads             | Num  | Sets number of threads for a hook (per compute unit)       | --hook-threads=8
```

```
    --hash-info               |      | Show information for each hash-mode                    |
    --example-hashes          |      | Alias of --hash-info                                   |
    --backend-ignore-cuda     |      | Do not try to open CUDA interface on startup           |
    --backend-ignore-hip      |      | Do not try to open HIP interface on startup            |
    --backend-ignore-metal    |      | Do not try to open Metal interface on startup          |
    --backend-ignore-opencl   |      | Do not try to open OpenCL interface on startup         |
-I, --backend-info            |      | Show system/evironment/backend API info                | -I or -II
-d, --backend-devices         | Str  | Backend devices to use, separated with commas          | -d 1
-D, --opencl-device-types     | Str  | OpenCL device-types to use, separated with commas      | -D 1
-O, --optimized-kernel-enable |      | Enable optimized kernels (limits password length)      |
-M, --multiply-accel-disable  |      | Disable multiply kernel-accel with processor count     |
-w, --workload-profile        | Num  | Enable a specific workload profile, see pool below     | -w 3
-n, --kernel-accel            | Num  | Manual workload tuning, set outerloop step size to X   | -n 64
-u, --kernel-loops            | Num  | Manual workload tuning, set innerloop step size to X   | -u 256
-T, --kernel-threads          | Num  | Manual workload tuning, set thread count to X          | -T 64
    --backend-vector-width    | Num  | Manually override backend vector-width to X            | --backend-vector=4
    --spin-damp               | Num  | Use CPU for device synchronization, in percent         | --spin-damp=10
    --hwmon-disable           |      | Disable temperature and fanspeed reads and triggers    |
    --hwmon-temp-abort        | Num  | Abort if temperature reaches X degrees Celsius         | --hwmon-temp-abort=100
    --scrypt-tmto             | Num  | Manually override TMTO value for scrypt to X           | --scrypt-tmto=3
-s, --skip                    | Num  | Skip X words from the start                            | -s 1000000
-l, --limit                   | Num  | Limit X words from the start + skipped words           | -l 1000000
    --keyspace                |      | Show keyspace base:mod values and quit                 |
-j, --rule-left               | Rule | Single rule applied to each word from left wordlist    | -j 'c'
-k, --rule-right              | Rule | Single rule applied to each word from right wordlist   | -k '^-'
-r, --rules-file              | File | Multiple rules applied to each word from wordlists     | -r rules/best64.rule
-g, --generate-rules          | Num  | Generate X random rules                                | -g 10000
    --generate-rules-func-min | Num  | Force min X functions per rule                         |
    --generate-rules-func-max | Num  | Force max X functions per rule                         |
    --generate-rules-func-sel | Str  | Pool of rule operators valid for random rule engine    | --generate-rules-func-se
    --generate-rules-seed     | Num  | Force RNG seed set to X                                |
-1, --custom-charset1         | CS   | User-defined charset ?1                                | -1 ?l?d?u
-2, --custom-charset2         | CS   | User-defined charset ?2                                | -2 ?l?d?s
-3, --custom-charset3         | CS   | User-defined charset ?3                                |
-4, --custom-charset4         | CS   | User-defined charset ?4                                |
    --identify                |      | Shows all supported algorithms for input hashes        | --identify my.hash
-i, --increment               |      | Enable mask increment mode                             |
    --increment-min           | Num  | Start mask incrementing at X                           | --increment-min=4
    --increment-max           | Num  | Stop mask incrementing at X                            | --increment-max=8
-S, --slow-candidates         |      | Enable slower (but advanced) candidate generators      |
    --brain-server            |      | Enable brain server                                    |
    --brain-server-timer      | Num  | Update the brain server dump each X seconds (min:60)   | --brain-server-timer=300
-z, --brain-client            |      | Enable brain client, activates -S                      |
    --brain-client-features   | Num  | Define brain client features, see below                | --brain-client-features=
    --brain-host              | Str  | Brain server host (IP or domain)                       | --brain-host=127.0.0.1
    --brain-port              | Port | Brain server port                                      | --brain-port=13743
    --brain-password          | Str  | Brain server authentication password                   | --brain-password=bZfhCvG
    --brain-session           | Hex  | Overrides automatically calculated brain session       | --brain-session=0x2ae611
    --brain-session-whitelist | Hex  | Allow given sessions only, separated with commas       | --brain-session-whitelis

- [ Hash modes ] -

      # | Name                                                       | Category
  ======+============================================================+======================================
    900 | MD4                                                        | Raw Hash
      0 | MD5                                                        | Raw Hash
    100 | SHA1                                                       | Raw Hash
   1300 | SHA2-224                                                   | Raw Hash
   1400 | SHA2-256                                                   | Raw Hash
  10800 | SHA2-384                                                   | Raw Hash
   1700 | SHA2-512                                                   | Raw Hash
  17300 | SHA3-224                                                   | Raw Hash
  17400 | SHA3-256                                                   | Raw Hash
  17500 | SHA3-384                                                   | Raw Hash
  17600 | SHA3-512                                                   | Raw Hash
   6000 | RIPEMD-160                                                 | Raw Hash
    600 | BLAKE2b-512                                                | Raw Hash
  11700 | GOST R 34.11-2012 (Streebog) 256-bit, big-endian           | Raw Hash
  11800 | GOST R 34.11-2012 (Streebog) 512-bit, big-endian           | Raw Hash
   6900 | GOST R 34.11-94                                            | Raw Hash
  17010 | GPG (AES-128/AES-256 (SHA-1($pass)))                       | Raw Hash
   5100 | Half MD5                                                   | Raw Hash
  17700 | Keccak-224                                                 | Raw Hash
  17800 | Keccak-256                                                 | Raw Hash
  17900 | Keccak-384                                                 | Raw Hash
  18000 | Keccak-512                                                 | Raw Hash
   6100 | Whirlpool                                                  | Raw Hash
  10100 | SipHash                                                    | Raw Hash
     70 | md5(utf16le($pass))                                        | Raw Hash
    170 | sha1(utf16le($pass))                                       | Raw Hash
   1470 | sha256(utf16le($pass))                                     | Raw Hash
```

10870	sha384(utf16le($pass))	Raw Hash
1770	sha512(utf16le($pass))	Raw Hash
610	BLAKE2b-512($pass.$salt)	Raw Hash salted and/or iterated
620	BLAKE2b-512($salt.$pass)	Raw Hash salted and/or iterated
10	md5($pass.$salt)	Raw Hash salted and/or iterated
20	md5($salt.$pass)	Raw Hash salted and/or iterated
3800	md5($salt.$pass.$salt)	Raw Hash salted and/or iterated
3710	md5($salt.md5($pass))	Raw Hash salted and/or iterated
4110	md5($salt.md5($pass.$salt))	Raw Hash salted and/or iterated
4010	md5($salt.md5($salt.$pass))	Raw Hash salted and/or iterated
21300	md5($salt.sha1($salt.$pass))	Raw Hash salted and/or iterated
40	md5($salt.utf16le($pass))	Raw Hash salted and/or iterated
2600	md5(md5($pass))	Raw Hash salted and/or iterated
3910	md5(md5($pass).md5($salt))	Raw Hash salted and/or iterated
3500	md5(md5(md5($pass)))	Raw Hash salted and/or iterated
4400	md5(sha1($pass))	Raw Hash salted and/or iterated
4410	md5(sha1($pass).$salt)	Raw Hash salted and/or iterated
20900	md5(sha1($pass).md5($pass).sha1($pass))	Raw Hash salted and/or iterated
21200	md5(sha1($salt).md5($pass))	Raw Hash salted and/or iterated
4300	md5(strtoupper(md5($pass)))	Raw Hash salted and/or iterated
30	md5(utf16le($pass).$salt)	Raw Hash salted and/or iterated
110	sha1($pass.$salt)	Raw Hash salted and/or iterated
120	sha1($salt.$pass)	Raw Hash salted and/or iterated
4900	sha1($salt.$pass.$salt)	Raw Hash salted and/or iterated
4520	sha1($salt.sha1($pass))	Raw Hash salted and/or iterated
24300	sha1($salt.sha1($pass.$salt))	Raw Hash salted and/or iterated
140	sha1($salt.utf16le($pass))	Raw Hash salted and/or iterated
19300	sha1($salt1.$pass.$salt2)	Raw Hash salted and/or iterated
14400	sha1(CX)	Raw Hash salted and/or iterated
4700	sha1(md5($pass))	Raw Hash salted and/or iterated
4710	sha1(md5($pass).$salt)	Raw Hash salted and/or iterated
21100	sha1(md5($pass.$salt))	Raw Hash salted and/or iterated
18500	sha1(md5(md5($pass)))	Raw Hash salted and/or iterated
4500	sha1(sha1($pass))	Raw Hash salted and/or iterated
4510	sha1(sha1($pass).$salt)	Raw Hash salted and/or iterated
5000	sha1(sha1($salt.$pass.$salt))	Raw Hash salted and/or iterated
130	sha1(utf16le($pass).$salt)	Raw Hash salted and/or iterated
1410	sha256($pass.$salt)	Raw Hash salted and/or iterated
1420	sha256($salt.$pass)	Raw Hash salted and/or iterated
22300	sha256($salt.$pass.$salt)	Raw Hash salted and/or iterated
20720	sha256($salt.sha256($pass))	Raw Hash salted and/or iterated
21420	sha256($salt.sha256_bin($pass))	Raw Hash salted and/or iterated
1440	sha256($salt.utf16le($pass))	Raw Hash salted and/or iterated
20800	sha256(md5($pass))	Raw Hash salted and/or iterated
20710	sha256(sha256($pass).$salt)	Raw Hash salted and/or iterated
21400	sha256(sha256_bin($pass))	Raw Hash salted and/or iterated
1430	sha256(utf16le($pass).$salt)	Raw Hash salted and/or iterated
10810	sha384($pass.$salt)	Raw Hash salted and/or iterated
10820	sha384($salt.$pass)	Raw Hash salted and/or iterated
10840	sha384($salt.utf16le($pass))	Raw Hash salted and/or iterated
10830	sha384(utf16le($pass).$salt)	Raw Hash salted and/or iterated
1710	sha512($pass.$salt)	Raw Hash salted and/or iterated
1720	sha512($salt.$pass)	Raw Hash salted and/or iterated
1740	sha512($salt.utf16le($pass))	Raw Hash salted and/or iterated
1730	sha512(utf16le($pass).$salt)	Raw Hash salted and/or iterated
50	HMAC-MD5 (key = $pass)	Raw Hash authenticated
60	HMAC-MD5 (key = $salt)	Raw Hash authenticated
150	HMAC-SHA1 (key = $pass)	Raw Hash authenticated
160	HMAC-SHA1 (key = $salt)	Raw Hash authenticated
1450	HMAC-SHA256 (key = $pass)	Raw Hash authenticated
1460	HMAC-SHA256 (key = $salt)	Raw Hash authenticated
1750	HMAC-SHA512 (key = $pass)	Raw Hash authenticated
1760	HMAC-SHA512 (key = $salt)	Raw Hash authenticated
11750	HMAC-Streebog-256 (key = $pass), big-endian	Raw Hash authenticated
11760	HMAC-Streebog-256 (key = $salt), big-endian	Raw Hash authenticated
11850	HMAC-Streebog-512 (key = $pass), big-endian	Raw Hash authenticated
11860	HMAC-Streebog-512 (key = $salt), big-endian	Raw Hash authenticated
28700	Amazon AWS4-HMAC-SHA256	Raw Hash authenticated
11500	CRC32	Raw Checksum
27900	CRC32C	Raw Checksum
28000	CRC64Jones	Raw Checksum
18700	Java Object hashCode()	Raw Checksum
25700	MurmurHash	Raw Checksum
27800	MurmurHash3	Raw Checksum
14100	3DES (PT = $salt, key = $pass)	Raw Cipher, Known-plaintext attack
14000	DES (PT = $salt, key = $pass)	Raw Cipher, Known-plaintext attack
26401	AES-128-ECB NOKDF (PT = $salt, key = $pass)	Raw Cipher, Known-plaintext attack
26402	AES-192-ECB NOKDF (PT = $salt, key = $pass)	Raw Cipher, Known-plaintext attack

```
 26403 | AES-256-ECB NOKDF (PT = $salt, key = $pass)               | Raw Cipher, Known-plaintext attack
 15400 | ChaCha20                                                 | Raw Cipher, Known-plaintext attack
 14500 | Linux Kernel Crypto API (2.4)                            | Raw Cipher, Known-plaintext attack
 14900 | Skip32 (PT = $salt, key = $pass)                         | Raw Cipher, Known-plaintext attack
 11900 | PBKDF2-HMAC-MD5                                          | Generic KDF
 12000 | PBKDF2-HMAC-SHA1                                         | Generic KDF
 10900 | PBKDF2-HMAC-SHA256                                       | Generic KDF
 12100 | PBKDF2-HMAC-SHA512                                       | Generic KDF
  8900 | scrypt                                                   | Generic KDF
   400 | phpass                                                   | Generic KDF
 16100 | TACACS+                                                  | Network Protocol
 11400 | SIP digest authentication (MD5)                          | Network Protocol
  5300 | IKE-PSK MD5                                              | Network Protocol
  5400 | IKE-PSK SHA1                                             | Network Protocol
 25100 | SNMPv3 HMAC-MD5-96                                       | Network Protocol
 25000 | SNMPv3 HMAC-MD5-96/HMAC-SHA1-96                          | Network Protocol
 25200 | SNMPv3 HMAC-SHA1-96                                      | Network Protocol
 26700 | SNMPv3 HMAC-SHA224-128                                   | Network Protocol
 26800 | SNMPv3 HMAC-SHA256-192                                   | Network Protocol
 26900 | SNMPv3 HMAC-SHA384-256                                   | Network Protocol
 27300 | SNMPv3 HMAC-SHA512-384                                   | Network Protocol
  2500 | WPA-EAPOL-PBKDF2                                         | Network Protocol
  2501 | WPA-EAPOL-PMK                                            | Network Protocol
 22000 | WPA-PBKDF2-PMKID+EAPOL                                   | Network Protocol
 22001 | WPA-PMK-PMKID+EAPOL                                      | Network Protocol
 16800 | WPA-PMKID-PBKDF2                                         | Network Protocol
 16801 | WPA-PMKID-PMK                                            | Network Protocol
  7300 | IPMI2 RAKP HMAC-SHA1                                     | Network Protocol
 10200 | CRAM-MD5                                                 | Network Protocol
 16500 | JWT (JSON Web Token)                                     | Network Protocol
 29200 | Radmin3                                                  | Network Protocol
 19600 | Kerberos 5, etype 17, TGS-REP                            | Network Protocol
 19800 | Kerberos 5, etype 17, Pre-Auth                           | Network Protocol
 28800 | Kerberos 5, etype 17, DB                                 | Network Protocol
 19700 | Kerberos 5, etype 18, TGS-REP                            | Network Protocol
 19900 | Kerberos 5, etype 18, Pre-Auth                           | Network Protocol
 28900 | Kerberos 5, etype 18, DB                                 | Network Protocol
  7500 | Kerberos 5, etype 23, AS-REQ Pre-Auth                    | Network Protocol
 13100 | Kerberos 5, etype 23, TGS-REP                            | Network Protocol
 18200 | Kerberos 5, etype 23, AS-REP                             | Network Protocol
  5500 | NetNTLMv1 / NetNTLMv1+ESS                                | Network Protocol
 27000 | NetNTLMv1 / NetNTLMv1+ESS (NT)                           | Network Protocol
  5600 | NetNTLMv2                                                | Network Protocol
 27100 | NetNTLMv2 (NT)                                           | Network Protocol
 29100 | Flask Session Cookie ($salt.$salt.$pass)                 | Network Protocol
  4800 | iSCSI CHAP authentication, MD5(CHAP)                     | Network Protocol
  8500 | RACF                                                     | Operating System
  6300 | AIX {smd5}                                               | Operating System
  6700 | AIX {ssha1}                                              | Operating System
  6400 | AIX {ssha256}                                            | Operating System
  6500 | AIX {ssha512}                                            | Operating System
  3000 | LM                                                       | Operating System
 19000 | QNX /etc/shadow (MD5)                                    | Operating System
 19100 | QNX /etc/shadow (SHA256)                                 | Operating System
 19200 | QNX /etc/shadow (SHA512)                                 | Operating System
 15300 | DPAPI masterkey file v1 (context 1 and 2)                | Operating System
 15310 | DPAPI masterkey file v1 (context 3)                      | Operating System
 15900 | DPAPI masterkey file v2 (context 1 and 2)                | Operating System
 15910 | DPAPI masterkey file v2 (context 3)                      | Operating System
  7200 | GRUB 2                                                   | Operating System
 12800 | MS-AzureSync PBKDF2-HMAC-SHA256                          | Operating System
 12400 | BSDi Crypt, Extended DES                                 | Operating System
  1000 | NTLM                                                     | Operating System
  9900 | Radmin2                                                  | Operating System
  5800 | Samsung Android Password/PIN                             | Operating System
 28100 | Windows Hello PIN/Password                               | Operating System
 13800 | Windows Phone 8+ PIN/password                            | Operating System
  2410 | Cisco-ASA MD5                                            | Operating System
  9200 | Cisco-IOS $8$ (PBKDF2-SHA256)                            | Operating System
  9300 | Cisco-IOS $9$ (scrypt)                                   | Operating System
  5700 | Cisco-IOS type 4 (SHA256)                                | Operating System
  2400 | Cisco-PIX MD5                                            | Operating System
  8100 | Citrix NetScaler (SHA1)                                  | Operating System
 22200 | Citrix NetScaler (SHA512)                                | Operating System
  1100 | Domain Cached Credentials (DCC), MS Cache                | Operating System
  2100 | Domain Cached Credentials 2 (DCC2), MS Cache 2           | Operating System
  7000 | FortiGate (FortiOS)                                      | Operating System
```

26300	FortiGate256 (FortiOS256)	Operating System
125	ArubaOS	Operating System
501	Juniper IVE	Operating System
22	Juniper NetScreen/SSG (ScreenOS)	Operating System
15100	Juniper/NetBSD sha1crypt	Operating System
26500	iPhone passcode (UID key + System Keybag)	Operating System
122	macOS v10.4, macOS v10.5, macOS v10.6	Operating System
1722	macOS v10.7	Operating System
7100	macOS v10.8+ (PBKDF2-SHA512)	Operating System
3200	bcrypt $2*$, Blowfish (Unix)	Operating System
500	md5crypt, MD5 (Unix), Cisco-IOS 1 (MD5)	Operating System
1500	descrypt, DES (Unix), Traditional DES	Operating System
29000	sha1($salt.sha1(utf16le($username).':'.utf16le($pass)))	Operating System
7400	sha256crypt 5, SHA256 (Unix)	Operating System
1800	sha512crypt 6, SHA512 (Unix)	Operating System
24600	SQLCipher	Database Server
131	MSSQL (2000)	Database Server
132	MSSQL (2005)	Database Server
1731	MSSQL (2012, 2014)	Database Server
24100	MongoDB ServerKey SCRAM-SHA-1	Database Server
24200	MongoDB ServerKey SCRAM-SHA-256	Database Server
12	PostgreSQL	Database Server
11100	PostgreSQL CRAM (MD5)	Database Server
28600	PostgreSQL SCRAM-SHA-256	Database Server
3100	Oracle H: Type (Oracle 7+)	Database Server
112	Oracle S: Type (Oracle 11+)	Database Server
12300	Oracle T: Type (Oracle 12+)	Database Server
7401	MySQL A (sha256crypt)	Database Server
11200	MySQL CRAM (SHA1)	Database Server
200	MySQL323	Database Server
300	MySQL4.1/MySQL5	Database Server
8000	Sybase ASE	Database Server
8300	DNSSEC (NSEC3)	FTP, HTTP, SMTP, LDAP Server
25900	KNX IP Secure - Device Authentication Code	FTP, HTTP, SMTP, LDAP Server
16400	CRAM-MD5 Dovecot	FTP, HTTP, SMTP, LDAP Server
1411	SSHA-256(Base64), LDAP {SSHA256}	FTP, HTTP, SMTP, LDAP Server
1711	SSHA-512(Base64), LDAP {SSHA512}	FTP, HTTP, SMTP, LDAP Server
24900	Dahua Authentication MD5	FTP, HTTP, SMTP, LDAP Server
10901	RedHat 389-DS LDAP (PBKDF2-HMAC-SHA256)	FTP, HTTP, SMTP, LDAP Server
15000	FileZilla Server >= 0.9.55	FTP, HTTP, SMTP, LDAP Server
12600	ColdFusion 10+	FTP, HTTP, SMTP, LDAP Server
1600	Apache $apr1$ MD5, md5apr1, MD5 (APR)	FTP, HTTP, SMTP, LDAP Server
141	Episerver 6.x < .NET 4	FTP, HTTP, SMTP, LDAP Server
1441	Episerver 6.x >= .NET 4	FTP, HTTP, SMTP, LDAP Server
1421	hMailServer	FTP, HTTP, SMTP, LDAP Server
101	nsldap, SHA-1(Base64), Netscape LDAP SHA	FTP, HTTP, SMTP, LDAP Server
111	nsldaps, SSHA-1(Base64), Netscape LDAP SSHA	FTP, HTTP, SMTP, LDAP Server
7700	SAP CODVN B (BCODE)	Enterprise Application Software (EAS)
7701	SAP CODVN B (BCODE) from RFC_READ_TABLE	Enterprise Application Software (EAS)
7800	SAP CODVN F/G (PASSCODE)	Enterprise Application Software (EAS)
7801	SAP CODVN F/G (PASSCODE) from RFC_READ_TABLE	Enterprise Application Software (EAS)
10300	SAP CODVN H (PWDSALTEDHASH) iSSHA-1	Enterprise Application Software (EAS)
133	PeopleSoft	Enterprise Application Software (EAS)
13500	PeopleSoft PS_TOKEN	Enterprise Application Software (EAS)
21500	SolarWinds Orion	Enterprise Application Software (EAS)
21501	SolarWinds Orion v2	Enterprise Application Software (EAS)
24	SolarWinds Serv-U	Enterprise Application Software (EAS)
8600	Lotus Notes/Domino 5	Enterprise Application Software (EAS)
8700	Lotus Notes/Domino 6	Enterprise Application Software (EAS)
9100	Lotus Notes/Domino 8	Enterprise Application Software (EAS)
26200	OpenEdge Progress Encode	Enterprise Application Software (EAS)
20600	Oracle Transportation Management (SHA256)	Enterprise Application Software (EAS)
4711	Huawei sha1(md5($pass).$salt)	Enterprise Application Software (EAS)
20711	AuthMe sha256	Enterprise Application Software (EAS)
22400	AES Crypt (SHA256)	Full-Disk Encryption (FDE)
27400	VMware VMX (PBKDF2-HMAC-SHA1 + AES-256-CBC)	Full-Disk Encryption (FDE)
14600	LUKS v1 (legacy)	Full-Disk Encryption (FDE)
29541	LUKS v1 RIPEMD-160 + AES	Full-Disk Encryption (FDE)
29542	LUKS v1 RIPEMD-160 + Serpent	Full-Disk Encryption (FDE)
29543	LUKS v1 RIPEMD-160 + Twofish	Full-Disk Encryption (FDE)
29511	LUKS v1 SHA-1 + AES	Full-Disk Encryption (FDE)
29512	LUKS v1 SHA-1 + Serpent	Full-Disk Encryption (FDE)
29513	LUKS v1 SHA-1 + Twofish	Full-Disk Encryption (FDE)
29521	LUKS v1 SHA-256 + AES	Full-Disk Encryption (FDE)
29522	LUKS v1 SHA-256 + Serpent	Full-Disk Encryption (FDE)
29523	LUKS v1 SHA-256 + Twofish	Full-Disk Encryption (FDE)
29531	LUKS v1 SHA-512 + AES	Full-Disk Encryption (FDE)
29532	LUKS v1 SHA-512 + Serpent	Full-Disk Encryption (FDE)

```
29533 | LUKS v1 SHA-512 + Twofish                                 | Full-Disk Encryption (FDE)
13711 | VeraCrypt RIPEMD160 + XTS 512 bit (legacy)                | Full-Disk Encryption (FDE)
13712 | VeraCrypt RIPEMD160 + XTS 1024 bit (legacy)               | Full-Disk Encryption (FDE)
13713 | VeraCrypt RIPEMD160 + XTS 1536 bit (legacy)               | Full-Disk Encryption (FDE)
13741 | VeraCrypt RIPEMD160 + XTS 512 bit + boot-mode (legacy)    | Full-Disk Encryption (FDE)
13742 | VeraCrypt RIPEMD160 + XTS 1024 bit + boot-mode (legacy)   | Full-Disk Encryption (FDE)
13743 | VeraCrypt RIPEMD160 + XTS 1536 bit + boot-mode (legacy)   | Full-Disk Encryption (FDE)
29411 | VeraCrypt RIPEMD160 + XTS 512 bit                         | Full-Disk Encryption (FDE)
29412 | VeraCrypt RIPEMD160 + XTS 1024 bit                        | Full-Disk Encryption (FDE)
29413 | VeraCrypt RIPEMD160 + XTS 1536 bit                        | Full-Disk Encryption (FDE)
29441 | VeraCrypt RIPEMD160 + XTS 512 bit + boot-mode             | Full-Disk Encryption (FDE)
29442 | VeraCrypt RIPEMD160 + XTS 1024 bit + boot-mode            | Full-Disk Encryption (FDE)
29443 | VeraCrypt RIPEMD160 + XTS 1536 bit + boot-mode            | Full-Disk Encryption (FDE)
13751 | VeraCrypt SHA256 + XTS 512 bit (legacy)                   | Full-Disk Encryption (FDE)
13752 | VeraCrypt SHA256 + XTS 1024 bit (legacy)                  | Full-Disk Encryption (FDE)
13753 | VeraCrypt SHA256 + XTS 1536 bit (legacy)                  | Full-Disk Encryption (FDE)
13761 | VeraCrypt SHA256 + XTS 512 bit + boot-mode (legacy)       | Full-Disk Encryption (FDE)
13762 | VeraCrypt SHA256 + XTS 1024 bit + boot-mode (legacy)      | Full-Disk Encryption (FDE)
13763 | VeraCrypt SHA256 + XTS 1536 bit + boot-mode (legacy)      | Full-Disk Encryption (FDE)
29451 | VeraCrypt SHA256 + XTS 512 bit                            | Full-Disk Encryption (FDE)
29452 | VeraCrypt SHA256 + XTS 1024 bit                           | Full-Disk Encryption (FDE)
29453 | VeraCrypt SHA256 + XTS 1536 bit                           | Full-Disk Encryption (FDE)
29461 | VeraCrypt SHA256 + XTS 512 bit + boot-mode                | Full-Disk Encryption (FDE)
29462 | VeraCrypt SHA256 + XTS 1024 bit + boot-mode               | Full-Disk Encryption (FDE)
29463 | VeraCrypt SHA256 + XTS 1536 bit + boot-mode               | Full-Disk Encryption (FDE)
13721 | VeraCrypt SHA512 + XTS 512 bit (legacy)                   | Full-Disk Encryption (FDE)
13722 | VeraCrypt SHA512 + XTS 1024 bit (legacy)                  | Full-Disk Encryption (FDE)
13723 | VeraCrypt SHA512 + XTS 1536 bit (legacy)                  | Full-Disk Encryption (FDE)
29421 | VeraCrypt SHA512 + XTS 512 bit                            | Full-Disk Encryption (FDE)
29422 | VeraCrypt SHA512 + XTS 1024 bit                           | Full-Disk Encryption (FDE)
29423 | VeraCrypt SHA512 + XTS 1536 bit                           | Full-Disk Encryption (FDE)
13771 | VeraCrypt Streebog-512 + XTS 512 bit (legacy)             | Full-Disk Encryption (FDE)
13772 | VeraCrypt Streebog-512 + XTS 1024 bit (legacy)            | Full-Disk Encryption (FDE)
13773 | VeraCrypt Streebog-512 + XTS 1536 bit (legacy)            | Full-Disk Encryption (FDE)
13781 | VeraCrypt Streebog-512 + XTS 512 bit + boot-mode (legacy) | Full-Disk Encryption (FDE)
13782 | VeraCrypt Streebog-512 + XTS 1024 bit + boot-mode (legacy)| Full-Disk Encryption (FDE)
13783 | VeraCrypt Streebog-512 + XTS 1536 bit + boot-mode (legacy)| Full-Disk Encryption (FDE)
29471 | VeraCrypt Streebog-512 + XTS 512 bit                      | Full-Disk Encryption (FDE)
29472 | VeraCrypt Streebog-512 + XTS 1024 bit                     | Full-Disk Encryption (FDE)
29473 | VeraCrypt Streebog-512 + XTS 1536 bit                     | Full-Disk Encryption (FDE)
29481 | VeraCrypt Streebog-512 + XTS 512 bit + boot-mode          | Full-Disk Encryption (FDE)
29482 | VeraCrypt Streebog-512 + XTS 1024 bit + boot-mode         | Full-Disk Encryption (FDE)
29483 | VeraCrypt Streebog-512 + XTS 1536 bit + boot-mode         | Full-Disk Encryption (FDE)
13731 | VeraCrypt Whirlpool + XTS 512 bit (legacy)                | Full-Disk Encryption (FDE)
13732 | VeraCrypt Whirlpool + XTS 1024 bit (legacy)               | Full-Disk Encryption (FDE)
13733 | VeraCrypt Whirlpool + XTS 1536 bit (legacy)               | Full-Disk Encryption (FDE)
29431 | VeraCrypt Whirlpool + XTS 512 bit                         | Full-Disk Encryption (FDE)
29432 | VeraCrypt Whirlpool + XTS 1024 bit                        | Full-Disk Encryption (FDE)
29433 | VeraCrypt Whirlpool + XTS 1536 bit                        | Full-Disk Encryption (FDE)
23900 | BestCrypt v3 Volume Encryption                            | Full-Disk Encryption (FDE)
16700 | FileVault 2                                               | Full-Disk Encryption (FDE)
27500 | VirtualBox (PBKDF2-HMAC-SHA256 & AES-128-XTS)             | Full-Disk Encryption (FDE)
27600 | VirtualBox (PBKDF2-HMAC-SHA256 & AES-256-XTS)             | Full-Disk Encryption (FDE)
20011 | DiskCryptor SHA512 + XTS 512 bit                          | Full-Disk Encryption (FDE)
20012 | DiskCryptor SHA512 + XTS 1024 bit                         | Full-Disk Encryption (FDE)
20013 | DiskCryptor SHA512 + XTS 1536 bit                         | Full-Disk Encryption (FDE)
22100 | BitLocker                                                 | Full-Disk Encryption (FDE)
12900 | Android FDE (Samsung DEK)                                 | Full-Disk Encryption (FDE)
 8800 | Android FDE <= 4.3                                        | Full-Disk Encryption (FDE)
18300 | Apple File System (APFS)                                  | Full-Disk Encryption (FDE)
 6211 | TrueCrypt RIPEMD160 + XTS 512 bit (legacy)                | Full-Disk Encryption (FDE)
 6212 | TrueCrypt RIPEMD160 + XTS 1024 bit (legacy)               | Full-Disk Encryption (FDE)
 6213 | TrueCrypt RIPEMD160 + XTS 1536 bit (legacy)               | Full-Disk Encryption (FDE)
 6241 | TrueCrypt RIPEMD160 + XTS 512 bit + boot-mode (legacy)    | Full-Disk Encryption (FDE)
 6242 | TrueCrypt RIPEMD160 + XTS 1024 bit + boot-mode (legacy)   | Full-Disk Encryption (FDE)
 6243 | TrueCrypt RIPEMD160 + XTS 1536 bit + boot-mode (legacy)   | Full-Disk Encryption (FDE)
29311 | TrueCrypt RIPEMD160 + XTS 512 bit                         | Full-Disk Encryption (FDE)
29312 | TrueCrypt RIPEMD160 + XTS 1024 bit                        | Full-Disk Encryption (FDE)
29313 | TrueCrypt RIPEMD160 + XTS 1536 bit                        | Full-Disk Encryption (FDE)
29341 | TrueCrypt RIPEMD160 + XTS 512 bit + boot-mode             | Full-Disk Encryption (FDE)
29342 | TrueCrypt RIPEMD160 + XTS 1024 bit + boot-mode            | Full-Disk Encryption (FDE)
29343 | TrueCrypt RIPEMD160 + XTS 1536 bit + boot-mode            | Full-Disk Encryption (FDE)
 6221 | TrueCrypt SHA512 + XTS 512 bit (legacy)                   | Full-Disk Encryption (FDE)
 6222 | TrueCrypt SHA512 + XTS 1024 bit (legacy)                  | Full-Disk Encryption (FDE)
 6223 | TrueCrypt SHA512 + XTS 1536 bit (legacy)                  | Full-Disk Encryption (FDE)
29321 | TrueCrypt SHA512 + XTS 512 bit                            | Full-Disk Encryption (FDE)
29322 | TrueCrypt SHA512 + XTS 1024 bit                           | Full-Disk Encryption (FDE)
29323 | TrueCrypt SHA512 + XTS 1536 bit                           | Full-Disk Encryption (FDE)
```

```
 29333 | TrueCrypt Whirlpool + XTS 1536 bit                                | Full-Disk Encryption (FDE)
 12200 | eCryptfs                                                         | Full-Disk Encryption (FDE)
 10400 | PDF 1.1 - 1.3 (Acrobat 2 - 4)                                    | Document
 10410 | PDF 1.1 - 1.3 (Acrobat 2 - 4), collider #1                       | Document
 10420 | PDF 1.1 - 1.3 (Acrobat 2 - 4), collider #2                       | Document
 10500 | PDF 1.4 - 1.6 (Acrobat 5 - 8)                                    | Document
 25400 | PDF 1.4 - 1.6 (Acrobat 5 - 8) - user and owner pass              | Document
 10600 | PDF 1.7 Level 3 (Acrobat 9)                                      | Document
 10700 | PDF 1.7 Level 8 (Acrobat 10 - 11)                                | Document
  9400 | MS Office 2007                                                   | Document
  9500 | MS Office 2010                                                   | Document
  9600 | MS Office 2013                                                   | Document
 25300 | MS Office 2016 - SheetProtection                                 | Document
  9700 | MS Office <= 2003 $0/$1, MD5 + RC4                               | Document
  9710 | MS Office <= 2003 $0/$1, MD5 + RC4, collider #1                  | Document
  9720 | MS Office <= 2003 $0/$1, MD5 + RC4, collider #2                  | Document
  9810 | MS Office <= 2003 $3, SHA1 + RC4, collider #1                    | Document
  9820 | MS Office <= 2003 $3, SHA1 + RC4, collider #2                    | Document
  9800 | MS Office <= 2003 $3/$4, SHA1 + RC4                              | Document
 18400 | Open Document Format (ODF) 1.2 (SHA-256, AES)                    | Document
 18600 | Open Document Format (ODF) 1.1 (SHA-1, Blowfish)                 | Document
 16200 | Apple Secure Notes                                               | Document
 23300 | Apple iWork                                                      | Document
  6600 | 1Password, agilekeychain                                         | Password Manager
  8200 | 1Password, cloudkeychain                                         | Password Manager
  9000 | Password Safe v2                                                 | Password Manager
  5200 | Password Safe v3                                                 | Password Manager
  6800 | LastPass + LastPass sniffed                                      | Password Manager
 13400 | KeePass 1 (AES/Twofish) and KeePass 2 (AES)                      | Password Manager
 29700 | KeePass 1 (AES/Twofish) and KeePass 2 (AES) - keyfile only mode  | Password Manager
 23400 | Bitwarden                                                        | Password Manager
 16900 | Ansible Vault                                                    | Password Manager
 26000 | Mozilla key3.db                                                  | Password Manager
 26100 | Mozilla key4.db                                                  | Password Manager
 23100 | Apple Keychain                                                   | Password Manager
 11600 | 7-Zip                                                            | Archive
 12500 | RAR3-hp                                                          | Archive
 23800 | RAR3-p (Compressed)                                              | Archive
 23700 | RAR3-p (Uncompressed)                                            | Archive
 13000 | RAR5                                                             | Archive
 17220 | PKZIP (Compressed Multi-File)                                    | Archive
 17200 | PKZIP (Compressed)                                               | Archive
 17225 | PKZIP (Mixed Multi-File)                                         | Archive
 17230 | PKZIP (Mixed Multi-File Checksum-Only)                           | Archive
 17210 | PKZIP (Uncompressed)                                             | Archive
 20500 | PKZIP Master Key                                                 | Archive
 20510 | PKZIP Master Key (6 byte optimization)                           | Archive
 23001 | SecureZIP AES-128                                                | Archive
 23002 | SecureZIP AES-192                                                | Archive
 23003 | SecureZIP AES-256                                                | Archive
 13600 | WinZip                                                           | Archive
 18900 | Android Backup                                                   | Archive
 24700 | Stuffit5                                                         | Archive
 13200 | AxCrypt 1                                                        | Archive
 13300 | AxCrypt 1 in-memory SHA1                                         | Archive
 23500 | AxCrypt 2 AES-128                                                | Archive
 23600 | AxCrypt 2 AES-256                                                | Archive
 14700 | iTunes backup < 10.0                                             | Archive
 14800 | iTunes backup >= 10.0                                            | Archive
  8400 | WBB3 (Woltlab Burning Board)                                     | Forums, CMS, E-Commerce
  2612 | PHPS                                                             | Forums, CMS, E-Commerce
   121 | SMF (Simple Machines Forum) > v1.1                               | Forums, CMS, E-Commerce
  3711 | MediaWiki B type                                                 | Forums, CMS, E-Commerce
  4521 | Redmine                                                          | Forums, CMS, E-Commerce
 24800 | Umbraco HMAC-SHA1                                                | Forums, CMS, E-Commerce
    11 | Joomla < 2.5.18                                                  | Forums, CMS, E-Commerce
 13900 | OpenCart                                                         | Forums, CMS, E-Commerce
 11000 | PrestaShop                                                       | Forums, CMS, E-Commerce
 16000 | Tripcode                                                         | Forums, CMS, E-Commerce
  7900 | Drupal7                                                          | Forums, CMS, E-Commerce
  4522 | PunBB                                                            | Forums, CMS, E-Commerce
  2811 | MyBB 1.2+, IPB2+ (Invision Power Board)                          | Forums, CMS, E-Commerce
  2611 | vBulletin < v3.8.5                                               | Forums, CMS, E-Commerce
  2711 | vBulletin >= v3.8.5                                              | Forums, CMS, E-Commerce
 25600 | bcrypt(md5($pass)) / bcryptmd5                                   | Forums, CMS, E-Commerce
 25800 | bcrypt(sha1($pass)) / bcryptsha1                                 | Forums, CMS, E-Commerce
 28400 | bcrypt(sha512($pass)) / bcryptsha512                             | Forums, CMS, E-Commerce
    21 | osCommerce, xt:Commerce                                          | Forums, CMS, E-Commerce
 18100 | TOTP (HMAC-SHA1)                                                 | One-Time Password
  2000 | STDOUT                                                           | Plaintext
 99999 | Plaintext                                                        | Plaintext
```

```
  21600 | Web2py pbkdf2-sha512                                       | Framework
  10000 | Django (PBKDF2-SHA256)                                    | Framework
    124 | Django (SHA-1)                                            | Framework
  12001 | Atlassian (PBKDF2-HMAC-SHA1)                              | Framework
  19500 | Ruby on Rails Restful-Authentication                      | Framework
  27200 | Ruby on Rails Restful Auth (one round, no sitekey)        | Framework
  30000 | Python Werkzeug MD5 (HMAC-MD5 (key = $salt))              | Framework
  30120 | Python Werkzeug SHA256 (HMAC-SHA256 (key = $salt))        | Framework
  20200 | Python passlib pbkdf2-sha512                              | Framework
  20300 | Python passlib pbkdf2-sha256                              | Framework
  20400 | Python passlib pbkdf2-sha1                                | Framework
  24410 | PKCS#8 Private Keys (PBKDF2-HMAC-SHA1 + 3DES/AES)         | Private Key
  24420 | PKCS#8 Private Keys (PBKDF2-HMAC-SHA256 + 3DES/AES)       | Private Key
  15500 | JKS Java Key Store Private Keys (SHA1)                    | Private Key
  22911 | RSA/DSA/EC/OpenSSH Private Keys ($0$)                     | Private Key
  22921 | RSA/DSA/EC/OpenSSH Private Keys ($6$)                     | Private Key
  22931 | RSA/DSA/EC/OpenSSH Private Keys ($1, $3$)                 | Private Key
  22941 | RSA/DSA/EC/OpenSSH Private Keys ($4$)                     | Private Key
  22951 | RSA/DSA/EC/OpenSSH Private Keys ($5$)                     | Private Key
  23200 | XMPP SCRAM PBKDF2-SHA1                                    | Instant Messaging Service
  28300 | Teamspeak 3 (channel hash)                                | Instant Messaging Service
  22600 | Telegram Desktop < v2.1.14 (PBKDF2-HMAC-SHA1)             | Instant Messaging Service
  24500 | Telegram Desktop >= v2.1.14 (PBKDF2-HMAC-SHA512)          | Instant Messaging Service
  22301 | Telegram Mobile App Passcode (SHA256)                     | Instant Messaging Service
     23 | Skype                                                     | Instant Messaging Service
  29600 | Terra Station Wallet (AES256-CBC(PBKDF2($pass)))          | Cryptocurrency Wallet
  26600 | MetaMask Wallet                                           | Cryptocurrency Wallet
  21000 | BitShares v0.x - sha512(sha512_bin(pass))                 | Cryptocurrency Wallet
  28501 | Bitcoin WIF private key (P2PKH), compressed               | Cryptocurrency Wallet
  28502 | Bitcoin WIF private key (P2PKH), uncompressed             | Cryptocurrency Wallet
  28503 | Bitcoin WIF private key (P2WPKH, Bech32), compressed      | Cryptocurrency Wallet
  28504 | Bitcoin WIF private key (P2WPKH, Bech32), uncompressed    | Cryptocurrency Wallet
  28505 | Bitcoin WIF private key (P2SH(P2WPKH)), compressed        | Cryptocurrency Wallet
  28506 | Bitcoin WIF private key (P2SH(P2WPKH)), uncompressed      | Cryptocurrency Wallet
  11300 | Bitcoin/Litecoin wallet.dat                               | Cryptocurrency Wallet
  16600 | Electrum Wallet (Salt-Type 1-3)                           | Cryptocurrency Wallet
  21700 | Electrum Wallet (Salt-Type 4)                             | Cryptocurrency Wallet
  21800 | Electrum Wallet (Salt-Type 5)                             | Cryptocurrency Wallet
  12700 | Blockchain, My Wallet                                     | Cryptocurrency Wallet
  15200 | Blockchain, My Wallet, V2                                 | Cryptocurrency Wallet
  18800 | Blockchain, My Wallet, Second Password (SHA256)           | Cryptocurrency Wallet
  25500 | Stargazer Stellar Wallet XLM                              | Cryptocurrency Wallet
  16300 | Ethereum Pre-Sale Wallet, PBKDF2-HMAC-SHA256              | Cryptocurrency Wallet
  15600 | Ethereum Wallet, PBKDF2-HMAC-SHA256                       | Cryptocurrency Wallet
  15700 | Ethereum Wallet, SCRYPT                                   | Cryptocurrency Wallet
  22500 | MultiBit Classic .key (MD5)                               | Cryptocurrency Wallet
  27700 | MultiBit Classic .wallet (scrypt)                         | Cryptocurrency Wallet
  22700 | MultiBit HD (scrypt)                                      | Cryptocurrency Wallet
  28200 | Exodus Desktop Wallet (scrypt)                            | Cryptocurrency Wallet

- [ Brain Client Features ] -

  # | Features
 ===+========
  1 | Send hashed passwords
  2 | Send attack positions
  3 | Send hashed passwords and attack positions

- [ Outfile Formats ] -

  # | Format
 ===+========
  1 | hash[:salt]
  2 | plain
  3 | hex_plain
  4 | crack_pos
  5 | timestamp absolute
  6 | timestamp relative

- [ Rule Debugging Modes ] -
```

```
  # | Format
 ===+========
  1 | Finding-Rule
  2 | Original-Word
  3 | Original-Word:Finding-Rule
  4 | Original-Word:Finding-Rule:Processed-Word
  5 | Original-Word:Finding-Rule:Processed-Word:Wordlist

- [ Attack Modes ] -

  # | Mode
 ===+======
  0 | Straight
  1 | Combination
  3 | Brute-force
  6 | Hybrid Wordlist + Mask
  7 | Hybrid Mask + Wordlist
  9 | Association

- [ Built-in Charsets ] -

  ? | Charset
 ===+=========
  l | abcdefghijklmnopqrstuvwxyz [a-z]
  u | ABCDEFGHIJKLMNOPQRSTUVWXYZ [A-Z]
  d | 0123456789                 [0-9]
  h | 0123456789abcdef           [0-9a-f]
  H | 0123456789ABCDEF           [0-9A-F]
  s |  !"#$%&'()*+,-./:;<=>?@[\]^_`{|}~
  a | ?l?u?d?s
  b | 0x00 - 0xff

- [ OpenCL Device Types ] -

  # | Device Type
 ===+=============
  1 | CPU
  2 | GPU
  3 | FPGA, DSP, Co-Processor

- [ Workload Profiles ] -

  # | Performance | Runtime | Power Consumption | Desktop Impact
 ===+=============+=========+===================+=================
  1 | Low         |   2 ms  | Low               | Minimal
  2 | Default     |  12 ms  | Economic          | Noticeable
  3 | High        |  96 ms  | High              | Unresponsive
  4 | Nightmare   | 480 ms  | Insane            | Headless

- [ License ] -

  hashcat is licensed under the MIT license
  Copyright and license terms are listed in docs/license.txt

- [ Basic Examples ] -

  Attack-          | Hash- |
  Mode             | Type  | Example command
 ==================+=======+==================================================================
  Wordlist         | $P$   | hashcat -a 0 -m 400 example400.hash example.dict
  Wordlist + Rules | MD5   | hashcat -a 0 -m 0 example0.hash example.dict -r rules/best64.rule
  Brute-Force      | MD5   | hashcat -a 3 -m 0 example0.hash ?a?a?a?a?a?a
  Combinator       | MD5   | hashcat -a 1 -m 0 example0.hash example.dict example.dict
  Association      | $1$   | hashcat -a 9 -m 500 example500.hash 1word.dict -r rules/best64.rule

If you still have no idea what just happened, try the following pages:

* https://hashcat.net/wiki/#howtos_videos_papers_articles_etc_in_the_wild
* https://hashcat.net/faq/

If you think you need help by a real human come to the hashcat Discord:

* https://hashcat.net/discord
```

Default Values

Attribute	Value	Note
--hash-type	0	
--attack-mode	0	
--version	0	
--help	0	
--quiet	0	
--hex-charset	0	
--hex-salt	0	
--hex-wordlist	0	
--force	0	do **NOT** use this unless you are a developer
--status	0	
--status-json	0	
--status-timer	10	
--stdin-timeout-abort	120	
--machine-readable	0	
--keep-guessing	0	
--self-test-disable	0	
--loopback	0	
--markov-hcstat2	hashcat.hcstat2	
--markov-disable	0	
--markov-classic	0	
--markov-threshold	0	
--runtime	0	
--session	hashcat	
--restore	0	
--restore-disable	0	
--restore-file-path		
--outfile		
--outfile-format	1,2	
--outfile-autohex-disable	0	
--outfile-check-timer	5	
--wordlist-autohex-disable	0	
--separator	:	
--stdout	0	
--show	0	
--left	0	
--username	0	
--remove	0	
--remove-timer	60	
--potfile-disable	0	
--potfile-path	hashcat.potfile	
--encoding-from	utf8	
--encoding-to	utf8	
--debug-mode	0	
--debug-file		
--induction-dir	hashcat.induct	
--outfile-check-dir	hashcat.outfiles	
--logfile-disable	0	

--hccapx-message-pair	0	
--nonce-error-corrections	8	
--keyboard-layout-mapping		
--truecrypt-keyfiles		
--veracrypt-keyfiles		
--veracrypt-pim-start	485	
--veracrypt-pim-stop	485	
--benchmark	0	
--benchmark-all	0	
--speed-only	0	
--progress-only	0	
--segment-size	33554432	
--bitmap-min	16	
--bitmap-max	18	
--cpu-affinity		
--hook-threads	0	
--example-hashes	0	
--backend-ignore-cuda	0	
--backend-ignore-opencl	0	
--backend-info	0	
--backend-devices		default: all GPUs if available
--opencl-device-types		default: GPUs if available
--optimized-kernel-enable	0	
--workload-profile	2	
--kernel-accel	0	default: autotune
--kernel-loops	0	default: autotune
--kernel-threads	0	default: autotune
--backend-vector-width	0	default: autotune
--spin-damp	0	
--hwmon-disable	0	
--hwmon-temp-abort	90	
--scrypt-tmto	0	default: autotune
--skip	0	
--limit	0	
--keyspace	0	
--rule-left	:	
--rule-right	:	
--rules-file		
--generate-rules	0	
--generate-rules-func-min	1	
--generate-rules-func-max	4	
--generate-rules-seed	0	
--custom-charset1	?l?d?u	#
--custom-charset2	?l?d	#
--custom-charset3	?l?d*!$@_	#
--custom-charset4		#
--increment	0	
--increment-min	1	
--increment-max	PW_MAX	limit depends on algo and pure/optimized mode
--slow-candidates	0	

--brain-server	0	
--brain-server-timer	300	
--brain-client	0	
--brain-client-features	2	
--brain-host	localhost	
--brain-port	6863	
--brain-password	$random	
--brain-session	$computed	
--brain-session-whitelist		

Note: if you do not specify any mask while performing a mask attack (-a 3), then the following default mask is used: **?1?2?2?2?2?2?2?3?3?3?3?d?d?d?d**

# Indicates all the custom charset values which **only** work together with the default mask (i.e. whenever no mask is specified at all). This is especially in since otherwise some users confuse ?1 (question mark and number one) with ?l (question mark and lowercase letter l) etc.

Supported attack modes

- Brute-Force attack (-a 3)
- Combinator attack (-a 1)
- Dictionary attack (-a 0)
- Hybrid attack (-a 6, -a 7)
- Mask attack (-a 3)
- Rule-based attack (-r option to -a 0)
- Toggle-Case attack (only supported by using rule files)
- Association attack (a -9)

Status output

The status output fields are as follows:

Status	Description
Session	The session name for this session of hashcat. Defaults to 'hashcat'. Can be changed with --session
Status	Status of this session - can be one of Paused, Running, Exhausted (not all hashes were cracked but attack has completed - this is no ror!), or Cracked (all target hashes cracked)
Hash.Mode	Which hash mode (hash type) is being attacked - name and number
Hash.Target	The target hash or hash file being attacked
Time.Started	Wall-clock (absolute) attack star time, and relative elapsed time
Time.Estimated	When the attack is estimated to finish, wall-clock and remaining time
Kernel.Feature	Whether the kernel being used is an optimized kernel or not
Guess.Base	The base input of the attack - often a wordlist or mask
Guess.Mod	The modifying input for the attack - varies by attack type
Guess.Queue	Shows how many guess base inputs (wordlists, masks) are in this attack, and which one we are currently using
Speed.#N	Cracking speed (in hashes per second) and speed parameters, per device
Speed.#*	Cracking speed (in hashes per second) and speed parameters, aggregated over all devices
Recovered	How many target hashes have been recovered - across all attacks (total), and during this attack (new)
Remaining	How many targets remain that have not been cracked
Recovered/Time	Recovery rate per minute, hour, and day - current and average
Progress	Progress within this attack, as a percentage of candidates
Rejected	How many candidates were rejected as inapplicable (too long, etc)
Restore.Point	How long until we will reach the next restore point - note that this can be very long depending on hash type
Restore.Sub.#N	TBD
Candidate.Engine	How candidates are being generated (Device Generator, etc.)
Candidates.#N	The range of candidates being tried on the specified device(s)
Hardware.Mon.#N	Per-device hardware status - temperature, fan speed, utilization, core clock speed, memory speed, bus speed

The status input options are as follows:

Option	Purpose
[s]tatus	Force display of full status information. Useful for when you want to see the status immediately instead of waiting until the –status-timer time has been reached
[p]ause / [r]es-ume	Pause and resume the current attack
[b]ypass	Skip the current attack or mask and move to the next one. If there are no attacks left in this session, hashcat will quit
[c]heckpoint	Quit at the next checkpoint (instead of quitting immediately)
[f]inish	Allow the current attack to complete, and then quit.
[q]uit	Quit immediately (without waiting for the next checkpoint)

Parsing the restore-file

john the ripper

```
└─$ john --help
John the Ripper 1.9.0-jumbo-1+bleeding-aec1328d6c 2021-11-02 10:45:52 +0100 O
MP [linux-gnu 64-bit x86_64 AVX2 AC]
Copyright (c) 1996-2021 by Solar Designer and others
Homepage: https://www.openwall.com/john/

Usage: john [OPTIONS] [PASSWORD-FILES]

--help                      Print usage summary
--single[=SECTION[,..]]     "Single crack" mode, using default or named rules
--single=:rule[,..]         Same, using "immediate" rule(s)
--single-seed=WORD[,WORD]   Add static seed word(s) for all salts in single mo
de
--single-wordlist=FILE      *Short* wordlist with static seed words/morphemes
--single-user-seed=FILE     Wordlist with seeds per username (user:password[s]
                            format)
--single-pair-max=N         Override max. number of word pairs generated (6)
--no-single-pair            Disable single word pair generation
--[no-]single-retest-guess  Override config for SingleRetestGuess
--wordlist[=FILE] --stdin   Wordlist mode, read words from FILE or stdin
                  --pipe    like --stdin, but bulk reads, and allows rules
--rules[=SECTION[,..]]      Enable word mangling rules (for wordlist or PRINCE
                            modes), using default or named rules
--rules=:rule[;..]]         Same, using "immediate" rule(s)
--rules-stack=SECTION[,..]  Stacked rules, applied after regular rules or to
                            modes that otherwise don't support rules
--rules-stack=:rule[;..]    Same, using "immediate" rule(s)
--rules-skip-nop            Skip any NOP ":" rules (you already ran w/o rules)
--loopback[=FILE]           Like --wordlist, but extract words from a .pot fil
e
--mem-file-size=SIZE        Size threshold for wordlist preload (default 2048
MB)
--dupe-suppression          Suppress all dupes in wordlist (and force preload)
--incremental[=MODE]        "Incremental" mode [using section MODE]
--incremental-charcount=N   Override CharCount for incremental mode
--external=MODE             External mode or word filter
--mask[=MASK]               Mask mode using MASK (or default from john.conf)
--markov[=OPTIONS]          "Markov" mode (see doc/MARKOV)
--mkv-stats=FILE            "Markov" stats file
--prince[=FILE]             PRINCE mode, read words from FILE
--prince-loopback[=FILE]    Fetch words from a .pot file
--prince-elem-cnt-min=N     Minimum number of elements per chain (1)
--prince-elem-cnt-max=[-]N  Maximum number of elements per chain (negative N i
s
                            relative to word length) (8)
--prince-skip=N             Initial skip
--prince-limit=N            Limit number of candidates generated
--prince-wl-dist-len        Calculate length distribution from wordlist
--prince-wl-max=N           Load only N words from input wordlist
--prince-case-permute       Permute case of first letter
--prince-mmap               Memory-map infile (not available with case permute
)
--prince-keyspace           Just show total keyspace that would be produced
                            (disregarding skip and limit)
--subsets[=CHARSET]         "Subsets" mode (see doc/SUBSETS)
--subsets-required=N        The N first characters of "subsets" charset are
                            the "required set"
--subsets-min-diff=N        Minimum unique characters in subset
--subsets-max-diff=[-]N     Maximum unique characters in subset (negative N is
                            relative to word length)
--subsets-prefer-short      Prefer shorter candidates over smaller subsets
--subsets-prefer-small      Prefer smaller subsets over shorter candidates
--make-charset=FILE         Make a charset, FILE will be overwritten
--stdout[=LENGTH]           Just output candidate passwords [cut at LENGTH]
--session=NAME              Give a new session the NAME
```

```
--status[=NAME]            Print status of a session [called NAME]
--restore[=NAME]           Restore an interrupted session [called NAME]
--[no-]crack-status        Emit a status line whenever a password is cracked
--progress-every=N         Emit a status line every N seconds
--show[=left]              Show cracked passwords [if =left, then uncracked]
--show=formats             Show information about hashes in a file (JSON)
--show=invalid             Show lines that are not valid for selected format(
s)
--test[=TIME]              Run tests and benchmarks for TIME seconds each
                           (if TIME is explicitly 0, test w/o benchmark)
--stress-test[=TIME]       Loop self tests forever
--test-full=LEVEL          Run more thorough self-tests
--no-mask                  Used with --test for alternate benchmark w/o mask
--skip-self-tests          Skip self tests
--users=[-]LOGIN|UID[,..]  [Do not] load this (these) user(s) only
--groups=[-]GID[,..]       Load users [not] of this (these) group(s) only
--shells=[-]SHELL[,..]     Load users with[out] this (these) shell(s) only
```

```
--salts=[-]COUNT[:MAX]     Load salts with[out] COUNT [to MAX] hashes, or
--salts=#M[-N]             Load M [to N] most populated salts
--costs=[-]C[:M][,...]     Load salts with[out] cost value Cn [to Mn]. For
                           tunable cost parameters, see doc/OPTIONS
--fork=N                   Fork N processes
--node=MIN[-MAX]/TOTAL     This node's number range out of TOTAL count
--save-memory=LEVEL        Enable memory saving, at LEVEL 1..3
--log-stderr               Log to screen instead of file
--verbosity=N              Change verbosity (1-5 or 6 for debug, default 3)
--no-log                   Disables creation and writing to john.log file
--bare-always-valid=Y      Treat bare hashes as valid (Y/N)
--catch-up=NAME            Catch up with existing (paused) session NAME
--config=FILE              Use FILE instead of john.conf or john.ini
--encoding=NAME            Input encoding (eg. UTF-8, ISO-8859-1). See also
                           doc/ENCODINGS.
--input-encoding=NAME      Input encoding (alias for --encoding)
--internal-codepage=NAME   Codepage used in rules/masks (see doc/ENCODINGS)
--target-encoding=NAME     Output encoding (used by format)
--force-tty                Set up terminal for reading keystrokes even if we'
re
                           not the foreground process
--field-separator-char=C   Use 'C' instead of the ':' in input and pot files
--[no-]keep-guessing       Try finding plaintext collisions
--list=WHAT                List capabilities, see --list=help or doc/OPTIONS
--length=N                 Shortcut for --min-len=N --max-len=N
--min-length=N             Request a minimum candidate length in bytes
--max-length=N             Request a maximum candidate length in bytes
--max-candidates=[-]N      Gracefully exit after this many candidates tried.
                           (if negative, reset count on each crack)
--max-run-time=[-]N        Gracefully exit after this many seconds (if negati
ve,
                           reset timer on each crack)
--mkpc=N                   Request a lower max. keys per crypt
--no-loader-dupecheck      Disable the dupe checking when loading hashes
--pot=NAME                 Pot file to use
--regen-lost-salts=N       Brute force unknown salts (see doc/OPTIONS)
--reject-printable         Reject printable binaries
--tune=HOW                 Tuning options (auto/report/N)
--subformat=FORMAT         Pick a benchmark format for --format=crypt
--format=[NAME|CLASS][,..] Force hash of type NAME. The supported formats can
                           be seen with --list=formats and --list=subformats.
                           See also doc/OPTIONS for more advanced selection o
f
                           format(s), including using classes and wildcards.

┌──(haras㉿haras)-[~]
└─$
```