John

`version: 1.9.0 arch: any all`

John Homepage | Package Tracker | Source Code Repository
Edit This Page

Metapackages

default everything large top10

Tools:
exploitation identify information-g… passwords post-exploitat…
reverse-engin… social-engine… top10 vulnerability web

Tool Documentation

Packages & Binaries

john

SIPdump base64conv bitlocker2john calc_stat cprepair
dmg2john eapmd5tojohn genmkvpwd gpg2john hccap2john
john keepass2john mailer mkvcalcproba putty2john
racf2john rar2john raw2dyna tgtsnarf uaf2john
unafs undrop unique unshadow vncpcap2john
wpapcap2john zip2john

john-data

- 1password2j…
- 7z2john
- DPAPImk2john
- adxcsouf2john
- aem2john
- aix2john
- andotp2john
- androidbacku…
- androidfde2jo…
- ansible2john
- apex2john
- applenotes2j…
- aruba2john
- atmail2john
- axcrypt2john
- bestcrypt2john
- bitcoin2john
- bitshares2john
- bitwarden2john
- bks2john
- blockchain2jo…
- ccache2john
- cisco2john
- cracf2john
- dashlane2john
- deepsound2j…
- diskcryptor2j…
- dmg2john
- ecryptfs2john
- ejabberd2john
- electrum2john
- encfs2john
- enpass2john
- enpass5tojohn
- ethereum2john
- filezilla2john
- geli2john
- hccapx2john
- htdigest2john
- ibmiscanner2…
- ikescan2john
- ios7tojohn
- itunes_backu…
- iwork2john
- kdcdump2john
- keychain2john
- keyring2john
- keystore2john
- kirbi2john
- known_hosts…
- krb2john
- kwallet2john
- lastpass2john
- ldif2john
- libreoffice2john
- lion2john
- lotus2john
- luks2john
- mac2john
- mcafee_epo2…
- monero2john
- money2john
- mosquitto2john
- mozilla2john
- multibit2john
- neo2john
- office2john
- openbsd_soft…
- openssl2john
- padlock2john
- pcap2john
- pdf2john
- pem2john
- pfx2john
- pgpdisk2john
- pgpsda2john
- pgpwde2john
- prosody2john
- ps_token2john
- pse2john
- pwsafe2john
- radius2john
- restic2john
- sap2john
- sense2john
- signal2john
- sipdump2john
- ssh2john
- sspr2john
- staroffice2john
- strip2john
- telegram2john
- tezos2john
- truecrypt2john
- vdi2john
- vmx2john
- zed2john

Learn more with OffSec

Pen-200 Pen-300

LIGHT DARK

Tool Documentation:

Mailer

```
root@kali:~# mailer
```

```
Usage: /usr/sbin/mailer PASSWORD-FILE
```

Unique

```
root@kali:~# unique
Usage: unique [-v] [-inp=fname] [-cut=len] [-mem=num] OUTPUT-FILE [-ex_fi

       reads from stdin 'normally', but can be overridden by optional -in
       If -ex_file=XX is used, then data from file XX is also used to
       unique the data, but nothing is ever written to XX. Thus, any data
       XX, will NOT output into OUTPUT-FILE (for making iterative diction
       -ex_file_only=XX assumes the file is 'unique', and only checks aga
       -cut=len  Will trim each input lines to 'len' bytes long, prior to
       the unique algorithm. The 'trimming' is done on any -ex_file[_only
       -mem=num.  A number that overrides the UNIQUE_HASH_LOG value from
       params.h.  The default is 21.  This can be raised, up to 25 (memor
       doubles each number).  If you go TOO large, unique will swap and t
       work VERY slow

       -v is for 'verbose' mode, outputs line counts during the run
```

john Usage Example

Using a wordlist (`–wordlist=/usr/share/john/password.lst`), apply mangling rules (`–rules`) and attempt to crack the password hashes in the given file (`unshadowed.txt`):

```
root@kali:~# john --wordlist=/usr/share/john/password.lst --rules unshado
Warning: detected hash type "sha512crypt", but the string is also recogni
Use the "--format=crypt" option to force loading these as that type inste
Loaded 1 password hash (sha512crypt [64/64])
toor             (root)
guesses: 1  time: 0:00:00:07 DONE (Mon May 19 08:13:05 2014)  c/s: 482  t
Use the "--show" option to display all of the cracked passwords reliably
```

```
kali@kali:~$ echo -n test2 | md5sum
ad0234829205b9033196ba818f7a872b  -
```

```
kali@kali:~$ echo -n test2 | md5sum | awk '{print $1}'
ad0234829205b9033196ba818f7a872b
kali@kali:~$ echo -n test2 | md5sum | awk '{print $1}' > hash
kali@kali:~$
kali@kali:~$ for x in $(seq 0 9); do echo test$x >> wordlists; done
kali@kali:~$ grep test2 wordlists
test2
kali@kali:~$ wc -l wordlists
10 wordlists
kali@kali:~$
kali@kali:~$ john --list=formats | grep -i 'md5'
descrypt, bsdicrypt, md5crypt, md5crypt-long, bcrypt, scrypt, LM, AFS,
aix-ssha512, andOTP, ansible, argon2, as400-des, as400-ssha1, asa-md5,
dahua, dashlane, diskcryptor, Django, django-scrypt, dmd5, dmg, dominose
mschapv2-naive, krb5pa-md5, mssql, mssql05, mssql12, multibit, mysqlna,
mysql-sha1, mysql, net-ah, nethalflm, netlm, netlmv2, net-md5, netntlmv2
netntlm, netntlm-naive, net-sha1, nk, notes, md5ns, nsec3, NT, o10glogon
PBKDF2-HMAC-MD4, PBKDF2-HMAC-MD5, PBKDF2-HMAC-SHA1, PBKDF2-HMAC-SHA256,
PHPS2, pix-md5, PKZIP, po, postgres, PST, PuTTY, pwsafe, qnx, RACF,
Raw-Keccak, Raw-Keccak-256, Raw-MD4, Raw-MD5, Raw-MD5u, Raw-SHA1,
Stribog-256, Stribog-512, STRIP, SunMD5, SybaseASE, Sybase-PROP, tacacs-
tcp-md5, telegram, tezos, Tiger, tc_aes_xts, tc_ripemd160, tc_ripemd160b
ZipMonster, plaintext, has-160, HMAC-MD5, HMAC-SHA1, HMAC-SHA224,
kali@kali:~$
kali@kali:~$ john  --format=raw-md5 --wordlist=wordlists hash
Created directory: /home/g0tmi1k/.john
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-MD5 [MD5 128/128 AVX 4x3])
Warning: no OpenMP support for this hash type, consider --fork=2
Press 'q' or Ctrl-C to abort, almost any other key for status
Warning: Only 10 candidates left, minimum 12 needed for performance.
test2            (?)
1g 0:00:00:00 DONE (2021-11-04 10:30) 100.0g/s 1000p/s 1000c/s 1000C/s t
Use the "--show --format=Raw-MD5" options to display all of the cracked
Session completed
kali@kali:~$
```

unique Usage Example

Using verbose mode (`-v`), read a list of passwords (`-inp=allwords.txt`) and save only unique words to a file (`uniques.txt`):

```
root@kali:~# unique -v -inp=allwords.txt uniques.txt
Total lines read 6089 Unique lines written 5083
```

Packages and Binaries:

john

John the Ripper is a tool designed to help systems administrators to find weak (easy to guess or crack through brute force) passwords, and even automatically mail users warning them about it, if it is desired.

Besides several crypt(3) password hash types most commonly found on various Unix flavors, supported out of the box are Kerberos AFS and Windows NT/2000/XP/2003 LM hashes, plus several more with contributed patches.

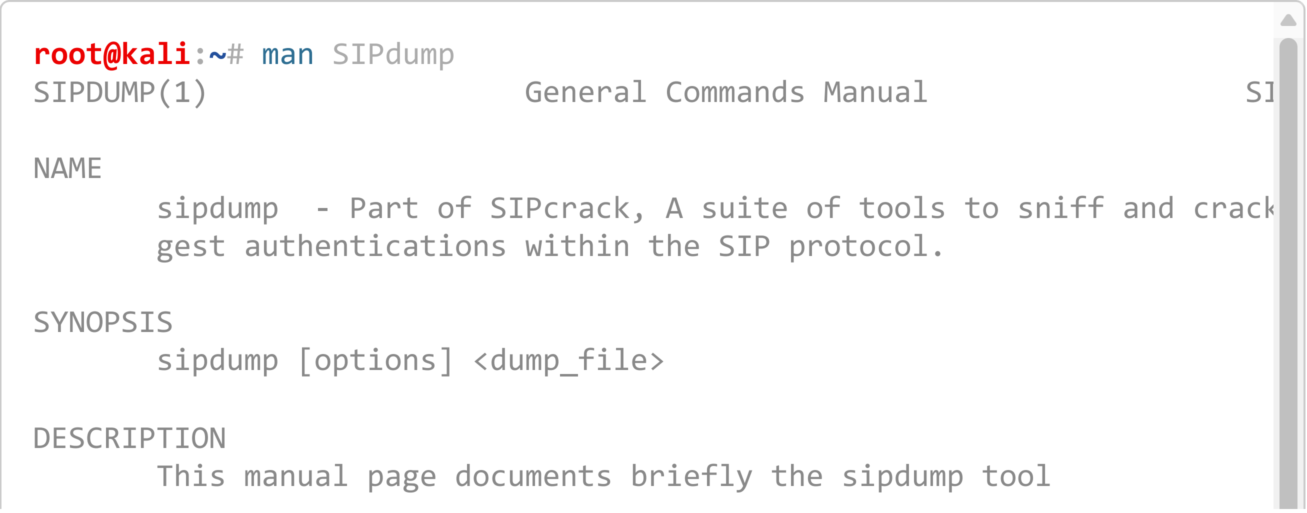
Installed size: `78.18 MB`
How to install: `sudo apt install john`

Dependencies:

- john-data
- libcrypt1
- libgomp1
- libssl3t64
- libc6
- libgmp10
- libpcap0.8t64
- zlib1g

SIPdump

Part of SIPcrack, A suite of tools to sniff and crack the digest authentications within the SIP protocol.



```
root@kali:~# man SIPdump
SIPDUMP(1)                    General Commands Manual                  SI

NAME
       sipdump  - Part of SIPcrack, A suite of tools to sniff and crack
       gest authentications within the SIP protocol.

SYNOPSIS
       sipdump [options] <dump_file>

DESCRIPTION
       This manual page documents briefly the sipdump tool
       Session Initiation Protocol (SIP) is a protocol developed  by  t
       MMUSIC  Working Group and is a proposed standard for initiating,
       ing, and terminating an interactive user session that involves
       dia  elements such as video, voice, instant messaging, online ga
       virtual reality.

       In November 2000, SIP was accepted as a 3GPP signaling protocol
       manent element of the IMS architecture.  It is one of the  leadi
       nalling  protocols for Voice over IP, along with H.323. In most
       lutions SIP is used to authenticate the SIPclient.  The protocol
       umented inside the RFC at www.ietf.org/rfc/rfc3261.txt

       SIPcrack is a SIP login sniffer/cracker that contains 2  program
       dump to capture the digest authentication and sipcrack to brutef
       hash using a wordlist or standard input.
       sipdump  dumps  SIP  digest  authentications.  If  a login is fo
       sniffed login is written to the dump file.  See  'sipdump  -h'
       tions.
       sipcrack bruteforces the user's password with the dump file gene
       sipdump.  If  a password is found, the sniffed and cracked login
       updated in the dump file.
       See 'sipcrack -h' for options.

OPTIONS
       A summary of options is included below.

       -i interface
```

base64conv

```
root@kali:~# base64conv -h
base64conv: invalid option -- 'h'
Usage: base64conv [-l] [-i intype] [-o outtype] [-q] [-w] [-e] [-f flag]
 - data must match input_type i.e. if hex, then data should be in hex
 - if data is not present, then base64conv will read data from std input
 - if data read from stdin, max size of any line is 256k

  -q will only output resultant string. No extra junk text
  -e turns on buffer overwrite error checking logic
  -l performs a 'length' test

  -r ifname  process whole file ifname (this is the input file)
  -w ofname  The output filename for whole file processing
             NOTE, -r and -w have to be used as a pair

Input/Output types:
  raw      raw data byte
  hex      hexadecimal string (for input, case does not matter)
  mime     base64 mime encoding
  crypt    base64 crypt character set encoding
  cryptBS  base64 crypt encoding, byte swapped

Flags (note more than 1 -f command switch can be given at one time):
  HEX_UPCASE         output or length UPCASED (input case auto handled)
  HEX_LOCASE         output or length locased (input case auto handled)
  MIME_TRAIL_EQ      output mime adds = chars (input = auto handled)
  CRYPT_TRAIL_DOTS   output crypt adds . chars (input . auto handled)
  MIME_PLUS_TO_DOT   mime converts + to . (passlib encoding)
  MIME_DASH_UNDER    mime convert +/ into -_ (passlib encoding)
```

bitlocker2john

```
root@kali:~# bitlocker2john -h

Usage: bitlocker2john -i <Image of encrypted memory unit>

Options:

  -h            Show this help
  -i            Image path of encrypted memory unit encrypted with BitLo
```

calc_stat

```
root@kali:~# calc_stat -h
Usage: calc_stat [-p] dictionary_file statfile
        -p: include non printable and 8-bit characters
```

cprepair

```
root@kali:~# cprepair -h
Codepage repair (c) magnum 2014-2019

Input can be a mix of codepages, UTF-8 and double-encoded UTF-8, and wit
a mix of Windows (CRLF) and Unix (LF) line endings, or missing line endi
on last lines.  If no file name is given, STDIN is used.
```

```
Output is UTF-8 with LF line endings and no silly BOM.

Usage: cprepair [options] [file(s)]
Options:
 -i <cp>   Codepage to assume for 8-bit input. Default is CP1252 (MS Lat
 -f <cp>   Alternate codepage when no ASCII letters (a-z, A-Z) seen (defa
           is to not treat them differently)
 -n        Do not guess (leave 8-bit as-is)
 -s        Suppress lines that does not need fixing.
 -d        Debug (show conversions).
 -l        List supported encodings.
 -p        Only convert stuff after first ':' (.pot file).
 -P        Suppress output lines with unprintable ASCII and, when used t
           with -n option, also suppress lines with invalid UTF-8
```

dmg2john

eapmd5tojohn

```
root@kali:~# eapmd5tojohn -h
Usage: eapmd5tojohn -r <pcap file>
```

genmkvpwd

```
root@kali:~# genmkvpwd -h
Usage: genmkvpwd statfile max_lvl [max_len] [start] [end]
```

gpg2john

hccap2john

john

A tool to find weak passwords of your users

```
root@kali:~# john -h
John the Ripper 1.9.0-jumbo-1+bleeding-aec1328d6c 2021-11-02 10:45:52 +
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
--single=:rule[,..]        Same, using "immediate" rule(s)
--single-seed=WORD[,WORD]  Add static seed word(s) for all salts in sir
--single-wordlist=FILE     *Short* wordlist with static seed words/morp
--single-user-seed=FILE    Wordlist with seeds per username (user:passw
                           format)
--single-pair-max=N        Override max. number of word pairs generatec
--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 stdir
                  --pipe   like --stdin, but bulk reads, and allows rul
--rules[=SECTION[,..]]     Enable word mangling rules (for wordlist or
                           modes), using default or named rules
--rules=:rule[;..]]        Same, using "immediate" rule(s)
--rules-stack=SECTION[,..] Stacked rules, applied after regular rules c
                           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
--loopback[=FILE]          Like --wordlist, but extract words from a .p
--mem-file-size=SIZE       Size threshold for wordlist preload (default
--dupe-suppression         Suppress all dupes in wordlist (and force pr
--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.c
--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 (negati
                           relative to word length) (8)
```

keepass2john

```
root@kali:~# keepass2john -h
keepass2john: invalid option -- 'h'
```

```
Usage: keepass2john [-k <keyfile>] <.kdbx database(s)>
```

mailer

Script to warn users about their weak passwords

```
root@kali:~# man mailer
MAILER(8)                    System Manager's Manual                   MA

NAME
       mailer - script to warn users about their weak passwords

SYNOPSIS
       mailer password-files

DESCRIPTION
       This  manual page documents briefly the mailer command, which is
       the john package.  This manual page was written for the Debian GN
       distribution because the original program does not have a  manual
       john,  better known as John the Ripper, is a tool to find weak pa
       of users in a server.
       The mailer tool is useful to inform users which have been  found
       using weak passwords by mail.

       You  should edit the message mailer will send to the users, but r
       to copy the script to a safe place before editing it, as it's  ge
       a bad idea to modify things living in /usr.

SEE ALSO
       john(8), unafs(8), unique(8), unshadow(8).

       The  programs are documented fully by John's documentation, which
       be available in /usr/share/doc/john or other location, depending
       system.

AUTHOR
       This manual page was written by Jordi  Mallach  <jordi@debian.org
       the Debian GNU/Linux system (but may be used by others).
       John  the  Ripper and mailer were written by Solar Designer <sola
       wall.com>. The complete list of contributors can be found in the
       file in the documentation directory.

john                               June 03, 2004                       MA
```

mkvcalcproba

putty2john

racf2john

rar2john

```
root@kali:~# rar2john -h
rar2john: invalid option -- 'h'
Usage: rar2john [-v] <rar file(s)>
Killed
```

raw2dyna

tgtsnarf

```
root@kali:~# tgtsnarf --help
tgtsnarf: invalid option -- '-'
Usage: tgtsnarf [-A] realm host [users...]
```

uaf2john

unafs

Script to warn users about their weak passwords

```
root@kali:~# unafs -h
Usage: unafs DATABASE-FILE CELL-NAME
```

undrop

unique

Removes duplicates from a wordlist

```
root@kali:~# man unique
UNIQUE(8)                    System Manager's Manual                    UNI

NAME
       unique - removes duplicates from a wordlist

SYNOPSIS
       unique output-file

DESCRIPTION
       This  manual page documents briefly the unique command, which is p
       the john package.  This manual page was written for the Debian GNU
       distribution because the original program does not have a  manual
       john,  better known as John the Ripper, is a tool to find weak pas
       of users in a server.
       The unique tool finds and removes  duplicate  entries  from  a  wo
       (read  from stdin), without changing the order. This is important
       crease the performance of john when using the wordlist method.

SEE ALSO
       john(8), mailer(8), unafs(8), unshadow(8).

       The programs are documented fully by John's documentation, which
       be available in /usr/share/doc/john or other location, depending o
       system.

AUTHOR
       This  manual  page  was written by Jordi Mallach <jordi@debian.org
       the Debian GNU/Linux system (but may be used by others).
       John the Ripper and mailer were written by Solar  Designer  <solar
       wall.com>. The complete list of contributors can be found in the C
       file in the documentation directory.

john                              June 03, 2004                         UNI
```

unshadow

Combines passwd and shadow files

```
root@kali:~# unshadow -h
Usage: unshadow PASSWORD-FILE SHADOW-FILE
```

vncpcap2john

wpapcap2john

```
root@kali:~# wpapcap2john -h
Converts PCAP or IVS2 files to JtR format.
Supported encapsulations: 802.11, Prism, Radiotap, PPI and TZSP over UDP
Usage: wpapcap2john [options] <file[s]>

-c                Show only complete auths (incomplete ones might be wrong
                  but we can crack what passwords were tried).
-v                Bump verbosity (can be used several times, try -vv)
-d                Do not suppress dupe hashes (per AP/STA pair)
-r                Ignore replay-count (may output fuzzed-anonce handshakes
-f <n>            Force anonce fuzzing with +/- <n>
-e <essid:mac>    Manually add Name:MAC pair(s) in case the file lacks bea
                  eg. -e "Magnum WIFI:6d:61:67:6e:75:6d"
-m <mac>          Ignore any packets not involving this mac address
```

zip2john

```
root@kali:~# zip2john -h
zip2john: invalid option -- 'h'
Usage: zip2john [options] [zip file(s)]
 -s Scan archive from the beginning, looking for local file headers. This
    is less reliable than going by the central index, but might work bett
    with corrupted or split archives.
Options for 'old' PKZIP encrypted files only:
 -a <filename>   This is a 'known' ASCII file. This can be faster, IF al
    files are larger, and you KNOW that at least one of them starts out 
    'pure' ASCII data.
 -o <filename>   Only use this file from the .zip file.
 -c This will create a 'checksum only' hash.  If there are many encrypte
    files in the .zip file, then this may be an option, and there will be
    enough data that false positives will not be seen.  Up to 8 files are
    supported. These hashes do not reveal actual file data.
```

```
  -m Use "file magic" as known-plain if applicable. This can be faster bu
     not 100% safe in all situations.

 NOTE: By default it is assumed that all files in each archive have the sa
 password. If that's not the case, the produced hash may be uncrackable.
 To avoid this, use -o option to pick a file at a time.
```

john-data

John the Ripper is a tool designed to help systems administrators to find weak (easy to guess or crack through brute force) passwords, and even automatically mail users warning them about it, if it is desired.

This package contains architecture-independent character sets usable by john and architecture-independent scripts.

Installed size: `61.07 MB`
How to install: `sudo apt install john-data`

Dependencies:
python3

1password2john

7z2john

DPAPImk2john

adxcsouf2john

aem2john

aix2john

```
root@kali:~# aix2john -h
usage: aix2john [-h] [-s] [-f FILENAME]

options:
  -h, --help   show this help message and exit
  -s           Use this option if "lpa_options = std_hash=true" is activ
  -f FILENAME  Specify the AIX shadow file filename to read (usually
               /etc/security/passwd)
```

andotp2john

androidbackup2john

androidfde2john

```
root@kali:~# androidfde2john -h
Usage: /usr/bin/androidfde2john <data partition / image> <footer partiti
```

ansible2john

apex2john

applenotes2john

aruba2john

atmail2john

axcrypt2john

bestcrypt2john

bitcoin2john

bitshares2john

bitwarden2john

bks2john

```
root@kali:~# bks2john -h
Usage: bks2john [options] <.bks / .uber file(s)>

Options:
  -h, --help            show this help message and exit
  -t TYPE, --type=TYPE  BKS keystore type (bks / uber)
```

blockchain2john

```
root@kali:~# blockchain2john -h
usage: /usr/bin/blockchain2john [blockchain wallet files]

options:
  -h, --help  show this help message and exit
  --json      is the wallet using v2 format?
  --base64    does the wallet contain only a base64 string?
```

ccache2john

cisco2john

```
root@kali:~# cisco2john -h
Usage:  /usr/bin/cisco2john [cisco config file(s)] >>hashfile 2>>seed.tx
        /usr/bin/cisco2john/john -format:md5 -wordlist:seed.txt -rules ha
```

cracf2john

dashlane2john

deepsound2john

```
root@kali:~# deepsound2john -h
usage: deepsound2john [-h] [--verbose] file [file ...]

positional arguments:
  file

options:
  -h, --help     show this help message and exit
  --verbose, -v
```

diskcryptor2john

dmg2john

ecryptfs2john

ejabberd2john

electrum2john

```
root@kali:~# electrum2john -h
Usage: electrum2john [options]

Options:
  -h, --help  show this help message and exit
  -t          force generation of truncated hashes
```

encfs2john

enpass2john

enpass5tojohn

ethereum2john

filezilla2john

geli2john

hccapx2john

```
root@kali:~# hccapx2john -h
usage: hccapx2john [-h] [-nc NC] [--no-mp] hccapx

hccapx2john, process hccapx file into a format suitable for use with JtR

positional arguments:
  hccapx      hccapx file to process

options:
  -h, --help  show this help message and exit
  -nc NC      AP nonce correction to be used, 0 to disable, default 8
  --no-mp     disable message_pair BE/LE/nc detection
```

htdigest2john

ibmiscanner2john

ikescan2john

ios7tojohn

itunes_backup2john

iwork2john

kdcdump2john

keychain2john

keyring2john

```
root@kali:~# keyring2john -h
usage: keyring2john [-h] KEYRING_FILE

keyring2john.py -> convert Gnome Keyring files to john format.

positional arguments:
  KEYRING_FILE  Input Gnome Keyring file
```

```
options:
  -h, --help    show this help message and exit
```

keystore2john

```
root@kali:~# keystore2john -h
Traceback (most recent call last):
  File "/usr/bin/keystore2john", line 80, in process_file
    fd = open(filename, "rb")
FileNotFoundError: [Errno 2] No such file or directory: '-h'

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "/usr/bin/keystore2john", line 187, in <module>
    process_file(sys.argv[i])
    ~~~~~~~~~~~~^^^^^^^^^^^^^
  File "/usr/bin/keystore2john", line 83, in process_file
    sys.stderr.write("! %s: %s\n" % filename, str(e))
                     ~~~~~~~~~~~~~^~~~~~~~~
TypeError: not enough arguments for format string
```

kirbi2john

```
root@kali:~# kirbi2john -h
usage: kirbi2john [-h] [-o [crack_file]] file.kirbi [file.kirbi ...]

Read Mimikatz kerberos ticket then modify it and save it in crack_file

positional arguments:
  file.kirbi         File name to crack. Use asterisk '*' for many files.
                     are exported with mimikatz or from extracttgsrepfrompe

options:
  -h, --help         show this help message and exit
  -o [crack_file]    File to save crackable output to (default is stdout
```

known_hosts2john

krb2john

kwallet2john

lastpass2john

ldif2john

libreoffice2john

lion2john

lotus2john

luks2john

mac2john

```
root@kali:~# mac2john -h
-h : [Errno 2] No such file or directory: '-h'
```

mcafee_epo2john

monero2john

money2john

mosquitto2john

```
root@kali:~# mosquitto2john -h
usage: mosquitto2john [-h] [-hc] [passwd_file ...]

positional arguments:
  passwd_file     Path to the source mosquitto_passwd file(s).

options:
  -h, --help      show this help message and exit
  -hc, --hashcat  Convert hashes to hashcat friendly formats.

Find more Information:
    See doc/README-mosquitto.md for info/troubleshooting.
```

mozilla2john

multibit2john

neo2john

office2john

openbsd_softraid2john

openssl2john

```
root@kali:~# openssl2john -h
Usage: openssl2john [options]

Options:
  -h, --help    show this help message and exit
  -p PLAINTEXT
  -a MINASCII
```

```
    -c CIPHER
    -m MD
```

padlock2john

pcap2john

pdf2john

```
root@kali:~# pdf2john --help
Syntax:  pdf2john.pl <.pdf file(s)>
```

pem2john

pfx2john

pgpdisk2john

pgpsda2john

pgpwde2john

prosody2john

ps_token2john

```
root@kali:~# ps_token2john -h
Based on tokenchpoken v0.5 beta's parse.py file
```

```
Oracle PS_TOKEN cracker. Token parser

Alexey Tyurin - a.tyurin at erpscan.io
ERPScan Research Group - https://www.erpscan.io

usage: ps_token2john [-h] -c COOKIE

options:
  -h, --help  show this help message and exit
  -c COOKIE   Set a victim's PS_TOKEN cookie for parsing
```

pse2john

pwsafe2john

radius2john

restic2john

sap2john

sense2john

signal2john

sipdump2john

ssh2john

```
root@kali:~# ssh2john -h
```

```
[Errno 2] No such file or directory: '-h'
```

sspr2john

```
root@kali:~# sspr2john -h
usage: sspr2john [-h] -H HOST [-p PORT] -b BASEDN [-s] [-D BINDDN]
                 [-w PASSWORD]

Utility to retrieve NetIQ SSPR hashes from a LDAP server.

options:
  -h, --help            show this help message and exit
  -H, --host HOST       Format like ad.example.net or 192.168.124.10
  -p, --port PORT       Format like 389 or 636
  -b, --basedn BASEDN   Format like CN=Users,DC=EXAMPLE,DC=NET
  -s, --secure          Use LDAPS (LDAP OVER SSL), recommended
  -D, --binddn BINDDN   Format like CN=<username>,CN=Users,DC=EXAMPLE,DC=
                        or <username>
  -w, --password PASSWORD
                        Password for LDAP bind
```

staroffice2john

strip2john

telegram2john

tezos2john

```
root@kali:~# tezos2john -h
usage: tezos2john [-h] [-i] [-I]

Creates Tezos File For John The Ripper

options:
  -h, --help            show this help message and exit
  -i, --ignoreRules, --ignorerules
```

```
                        Ignore All Rules, seed words, checksum, ...
  -I, --ignoreICORules, --ignoreicorules
                        Do Not Check To See If It Is A Valid ICO Format
                        seed words)
```

truecrypt2john

```
root@kali:~# truecrypt2john -h
Usage: truecrypt2john [options]

Options:
  -h, --help  show this help message and exit
  -b
```

vdi2john

vmx2john

zed2john

Learn more with

Want to learn more about john? get access to in-depth training and hands-on labs:

- PEN-200: 16.2. Password Attacks: Password Cracking Fundamentals
- PEN-300: 18.1.1. Linux Lateral Movement: SSH Keys
- PEN-300: 18.2.5. Linux Lateral Movement: Exploiting Playbooks for Ansible Credentials

PEN-200 course

PEN-300 course

Updated on: 2025-Nov-18

Edit this page

jboss-autopwn

joplin

Partnerships

Linode

Microsoft Azure

Microsoft Store (WSL)

Vagrant

Development

Bug Tracker

Continuous Integration

Network Mirror

Package Tracker

GitLab

Community

Discord

Support Forum

PeerTube

Follow Us

Bluesky

Facebook

Instagram

Mastodon

Substack

X

Newsletter

RSS

Policies

Cookie Policy

Privacy Policy

Trademark Policy

OffSec™

Kali Linux is part of OffSec's Community Projects
Learn more about OffSec's free, open-source penetration testing tools for cybersecurity professionals