John The Ripper

Cheat Sheet

Password Cracking & Hash Analysis Reference

Author: 0xNetrunner
Date: December 5th, 2025
Version: 1.0
Category: Password Security

Created with LaTeX ♥ Rose Pine Dawn Theme

Contents

1 Introduction

John the Ripper (JtR) is a free, open-source password security auditing and password recovery tool available for many operating systems. It is designed to detect weak Unix passwords, but can also crack a variety of other hashes.

Info

John the Ripper is one of the most popular password testing and breaking programs as it combines a number of password crackers into one package, autodetects password hash types, and includes a customizable cracker.

1.1 Installation

Kali Linux / Debian:

```
sudo apt update && sudo apt install john
```

Build from Source (Jumbo version):

```
git clone https://github.com/openwall/john.git
cd john/src
./configure && make -s clean && make -sj4
# Binary located at ../run/john
```

Tip

The **Jumbo** version includes community patches and supports many more hash formats than the core version. Always prefer Jumbo for penetration testing.

2 Basic Usage

2.1 Quick Start

The most basic usage of John is to simply run it against a file containing hashes:

```
john hashfile.txt
```

John will automatically detect the hash type and begin cracking using its default modes.

2.2 Core Command Structure

```
john [options] <password-file>
```

```
# Basic attack with wordlist
john --wordlist=/usr/share/wordlists/rockyou.txt hashes.txt

# Show cracked passwords
john --show hashes.txt

# Specify hash format
john --format=raw-md5 hashes.txt

```

```
# Use multiple CPU cores
john --fork=4 hashes.txt
```

3 Attack Modes

John supports several attack modes, each with different strategies for cracking passwords.

3.1 Wordlist Mode

The most common attack mode. John tries each word from a wordlist as a potential password.

```
# Basic wordlist attack
john --wordlist=rockyou.txt hashes.txt

# Wordlist with rules (mangling)
john --wordlist=rockyou.txt --rules hashes.txt

# Specific rule set
john --wordlist=rockyou.txt --rules=Jumbo hashes.txt
```

Warning

Large wordlists like `rockyou.txt` contain millions of entries. Ensure you have sufficient memory and storage.

3.2 Incremental Mode

Brute-force mode that tries all possible character combinations.

```
# Default incremental mode
john --incremental hashes.txt

# Specific character set
john --incremental=Digits hashes.txt
john --incremental=Alpha hashes.txt
john --incremental=Alnum hashes.txt

# Custom length limits
john --incremental --min-length=6 --max-length=8 hashes.txt
```

3.3 Single Crack Mode

Uses login names, GECOS fields, and home directory names as candidate passwords with mangling rules applied.

```
john --single hashes.txt
```

3.4 Mask Mode

Allows custom attack patterns using placeholders.

```
# ?d = digit, ?l = lowercase, ?u = uppercase, ?s = special
john --mask='?u?l?l?l?d?d?d?d' hashes.txt

```

```
# Example: Password123
john --mask='?u?l?l?l?l?l?l?l?d?d?d' hashes.txt

# Custom character sets
john --mask='[Pp]assword?d?d?d' hashes.txt
```

Tip

Mask Placeholders:
`?d` = digits (0-9)
`?l` = lowercase (a-z)
`?u` = uppercase (A-Z)
`?s` = special characters
`?a` = all printable ASCII
`?h` = hex lowercase
`?H` = hex uppercase

4 Hash Extraction Tools

John includes numerous `*2john` utilities to extract hashes from various file formats.

4.1 Archive Files

```
# ZIP files
zip2john protected.zip > zip.hash

# RAR files
rar2john protected.rar > rar.hash

# 7-Zip files
7z2john protected.7z > 7z.hash
```

4.2 Document Files

```
# PDF files
pdf2john document.pdf > pdf.hash

# Microsoft Office
office2john document.docx > office.hash

# OpenDocument
libreoffice2john document.odt > odt.hash
```

4.3 System & Network

```
# SSH private keys
ssh2john id_rsa > ssh.hash

# KeePass databases
keepass2john database.kdbx > keepass.hash

# GPG keys
gpg2john private.key > gpg.hash
```

```
# WiFi captures
wpapcap2john capture.cap > wifi.hash

# /etc/shadow (combine with passwd)
unshadow /etc/passwd /etc/shadow > unshadowed.txt
```

5 # Common Hash Formats

Format Flag	Description	Example Pattern
raw-md5	Plain MD5 hash	5d41402abc4b2a76...
raw-sha1	Plain SHA1 hash	aaf4c61ddcc5e8a2...
raw-sha256	Plain SHA256 hash	2cf24dba5fb0a30e...
raw-sha512	Plain SHA512 hash	cf83e1357eefb8bd...
nt	Windows NTLM hash	32ed87bdb5fdc5e9...
lm	Windows LM hash (legacy)	aad3b435b51404ee...
md5crypt	Linux MD5 crypt	1salt$hash...
sha512crypt	Linux SHA512 crypt	6salt$hash...
bcrypt	Bcrypt hash	$2a$10$salt...
descrypt	Traditional DES crypt	XXjzXYqV.HVL2
krb5tgs	Kerberos TGS-REP	$krb5tgs$23$*...
mssql12	MS SQL 2012+	0x0200...
mysql-sha1	MySQL 4.1+	*2470C0C06DEE...

```
# List all supported formats
john --list=formats

# Search for a specific format
john --list=formats | grep -i mysql

# Show format details
john --list=format-details --format=raw-sha256
```

6 Rules & Word Mangling

Rules transform wordlist entries to generate password variations.

6.1 Built-in Rule Sets

```
# Default rules
john --wordlist=words.txt --rules hashes.txt

# Specific rule sets
john --wordlist=words.txt --rules=Single hashes.txt
john --wordlist=words.txt --rules=Wordlist hashes.txt
john --wordlist=words.txt --rules=Extra hashes.txt
john --wordlist=words.txt --rules=Jumbo hashes.txt
john --wordlist=words.txt --rules=KoreLogic hashes.txt
john --wordlist=words.txt --rules=All hashes.txt
```

6.2 Rule Syntax

Rule	Description	Example
:	No-op (use word as-is)	password
l	Convert to lowercase	PASSWORD → password
u	Convert to uppercase	password → PASSWORD
c	Capitalize first letter	password → Password
C	Lowercase first, uppercase rest	password → pASSWORD
t	Toggle case of all characters	PaSsWoRd → pAsSwOrD
r	Reverse the word	password → drowssap
d	Duplicate the word	pass → passpass
$X	Append character X	pass → pass1
^X	Prepend character X	pass → 1pass
sXY	Replace X with Y	pass → p@ss
@X	Delete all occurrences of X	password → pssword

```
# Custom rules in john.conf
[List.Rules:MyRules]
:
c
c $1
c $1 $2 $3
c $!
sa@ se3 si1 so0
```

7 Session Management

```
# Create a named session
john --session=mysession hashes.txt

# Restore a session
john --restore=mysession

# Check session status
john --status=mysession

# List all sessions
ls ~/.john/*.rec
```

Info

Sessions are automatically saved periodically. If John is interrupted, you can resume from where it left off using `--restore`.

8 Performance Optimization

8.1 CPU Optimization

```
# Use multiple CPU cores (fork)
john --fork=8 hashes.txt

# OpenMP threading (alternative)
```

```
john --format=raw-sha256-opencl hashes.txt

# Set process priority
nice -n -20 john hashes.txt
```

8.2 GPU Acceleration

```
# List OpenCL devices
john --list=opencl-devices

# Use specific GPU
john --format=raw-sha256-opencl --device=1 hashes.txt

# Use all GPUs
john --format=nt-opencl hashes.txt
```

Warning

GPU cracking requires the Jumbo version with OpenCL support. Not all hash formats have GPU implementations.

9 Useful Options Reference

Option	Description
`-wordlist=FILE`	Use specified wordlist
`-format=NAME`	Force hash format
`-rules[=NAME]`	Enable word mangling rules
`-incremental[=MODE]`	Incremental (brute-force) mode
`-single`	Single crack mode
`-mask=MASK`	Mask attack mode
`-show`	Display cracked passwords
`-pot=FILE`	Use alternate pot file
`-fork=N`	Fork N processes
`-node=N/M`	Distributed cracking (node N of M)
`-session=NAME`	Name the session
`-restore[=NAME]`	Restore interrupted session
`-status[=NAME]`	Show session status
`-min-length=N`	Minimum password length
`-max-length=N`	Maximum password length
`-encoding=NAME`	Input file encoding
`-list=WHAT`	List capabilities (formats, rules, etc.)

10 Common Workflows

10.1 Linux Shadow File

```
# Step 1: Combine passwd and shadow
unshadow /etc/passwd /etc/shadow > unshadowed.txt

# Step 2: Crack
john unshadowed.txt
```

```
6
7 # Step 3: View results
8 john --show unshadowed.txt
```

10.2 Windows NTLM Hashes

```
1 # From hashdump or secretsdump output
2 john --format=nt --wordlist=rockyou.txt ntlm_hashes.txt
```

10.3 Protected ZIP File

```
1 # Extract hash
2 zip2john secret.zip > zip.hash
3
4 # Crack
5 john --wordlist=rockyou.txt zip.hash
6
7 # Get password
8 john --show zip.hash
```

10.4 WiFi WPA/WPA2

```
1 # Convert capture file
2 wpapcap2john capture.cap > wifi.hash
3
4 # Crack
5 john --wordlist=rockyou.txt --format=wpapsk wifi.hash
```

11 Output & Potfile

```
1 # Show cracked passwords
2 john --show hashes.txt
3
4 # Show cracked in specific format
5 john --show --format=raw-md5 hashes.txt
6
7 # Use custom potfile
8 john --pot=custom.pot hashes.txt
9
10 # Default potfile location
11 cat ~/.john/john.pot
```

Tip

The potfile stores all previously cracked passwords. John automatically skips hashes that are already in the potfile. To re-crack, use `--pot=/dev/null` or delete the potfile.

12 Quick Reference Card

Essential Commands

- `john hashes.txt` – Auto-detect and crack
- `john -wordlist=rockyou.txt hashes.txt` – Wordlist attack
- `john -incremental hashes.txt` – Brute-force
- `john -show hashes.txt` – Show cracked
- `john -format=raw-md5 hashes.txt` – Specify format
- `john -restore=session` – Resume session
- `john -list=formats` – List hash formats

Danger

Only use John the Ripper on systems you own or have explicit written permission to test. Unauthorized password cracking is illegal and unethical.

13 Resources

- **Official Website:** https://www.openwall.com/john/
- **GitHub (Jumbo):** https://github.com/openwall/john
- **Documentation:** https://www.openwall.com/john/doc/
- **Mailing List:** https://www.openwall.com/lists/john-users/