

### Beispiele

<code>\$ john --single --format =&lt;Hashtyp&gt; &lt;Passwortdatei&gt;</code>	Angriff mit Single Crack Mode <i>Nutzt Informationen aus Passwortdatei, um gezielten Angriff durchzuführen.</i> <i>Bsp.: User = "john", Password = "john123"</i>
<code>\$ john --wordlist=&lt;Wortliste&gt; --format=&lt;Hashtyp&gt; &lt;Passwortdatei&gt;</code>	Angriff mit spezifischer Wortliste
<code>\$ john --incremental[=Zeichensatz] &lt;Passwortdatei&gt;</code>	Angriff mit inkrementellem Modus (Brute Force)

### Was ist John the Ripper

- Open-Source Tool für Password-Cracking
- Standardmäßig CLI, GUI verfügbar
- Erkennt oft Hashtypen automatisch
- Unterstützt alle gängigen Hashmethoden

### Voraussetzungen

- Der Hashwert des gesuchten Passworts, z.B. aus /etc/shadow (Linux)
- Ggf. der verwendete Hashing-Algorithmus

### Funktionsweise von John

1. Hash-Typ (Verschlüsselungstyp) des gegebenen Hashwerts bestimmen
2. Generierung von Hashes für verschiedene Zeichenketten
3. Anhalten, wenn ein generierter Hash mit dem gegebenen Hash übereinstimmt (→ Hashkollision)

### Basisbefehl

```
$ john [option] <Passwortdatei>
```

### Masken

Regex-ähnliche Ausdrücke...

... zur Erstellung von Passwörtern im Brute-Force

```
$ john --mask='[Pp][Aa@][Ss5][Ss5]-[Ww][Oo0][Rr][Dd]' password.hash — z.B. P@55w0rD
```

... zur Anpassung von Wörtern aus Wörterbuch

```
$ john --wordlist="wordlist.txt" --mask='?w?d?d?s' — z.B. password → password12!
```

### Umgang mit Salts & speziellen Hashes

### Umgang mit Salts & speziellen Hashes

Beispiel für einen verketteten Hash mit Salt

```
sha1($s.md5($p)), $s =Salt, $p = Passwort
```

Lösung: Dynamische Formate

Entweder eigene oder vorgefertigte Subformate

Auflistung der Subformate

```
$ john --list=subformats
```

Nutzung eines eigenen dynamischen Formats

```
--format=dynamic="sha1($s.md5($p))"
```

Format des gespeicherten Hashwerts

```
[userID:][dynamic_<Subformat_Nr>]$base_16_hash[$salt]
```

### Weitere nützliche Optionen & Befehle

Mehrere CPU-Kerne verwenden

```
--fork=<Anzahl>
```

Session einen Namen geben

```
--session=<Name>
```

Unterbrochene Session wiederherstellen

```
--restore[=<Name>]
```

Gecrackte Hashes mit Passwörtern ausgeben

```
cat ~/.john/john.pot
```

Status einer Session ausgeben

```
--status[=<Name>]
```

### Die Option --format

Unterstützte Hashtypen

z.B. Raw-SHA256, md5crypt, bcrypt

Verfügbare Formate auslesen

```
$ john -list= formats
```

### Hashes aus verschlüsselten Dateien erhalten

### Nützliche Links

Kompakter "Einstiegsguide" mit Übungen <https://www.stationx.net/how-to-use--john-the-ripper/>

Ausführliche themenspezifische Guides <https://miloserdov.org/?tag=john-the-ripper>

Dokumentation <https://github.com/openwall/john/tree/bleeding-jumbo/doc>

CUPP: Tool zur Erstellung maßgeschneiderter Wörterlisten <https://github.com/Mebus/cupp>

### Der Modus --incremental

Unterstützte Zeichensätze

z.B. digits, upper, alnum, utf8

Verfügbare Zeichensätze auslesen

```
$ john -list= inc -modes
```

Beispiel für einen verketteten Hash mit Salt

```
sha1($s.md5($p)), $s = Salt, $p =  
Passwort
```

Lösung: Dynamische Formate

Entweder eigene oder vorgefertigte  
Subformate

Auflistung der Subformate

```
$ john --list=subformats
```

Nutzung eines eigenen dynamischen  
Formats

```
--format=dynamic="sha1($s.md5($p))"
```

Format des gespeicherten Hashwerts

```
[userID:][${dynamic_<Subformat_Nr>}]$b-  
ase_16_hash[$salt]
```

Programme, um JtR-geeignete Hashes aus Dateien zu extrahieren

```
xxx2john
```

Extrahieren des Hashwerts der Zip-Datei secret.zip

```
$ zip2john secret.zip > secret.hash
```

Auflistung aller Skripte

```
$ locate *2john
```

Nutzer- und Passwortinfos unter Linux kombinieren

```
$ unshadow /etc/passwd /etc/shadow > output.  
db
```

C

By **leon1**

[cheatography.com/leon1/](https://cheatography.com/leon1/)

Not published yet.

Last updated 19th October, 2024.

Page 2 of 2.

Sponsored by **CrosswordCheats.com**

Learn to solve cryptic crosswords!

<http://crosswordcheats.com>