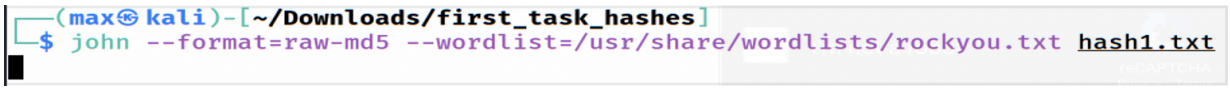
John the Ripper (in Kali)

hashid -to narrow down type of hash

CLI for john

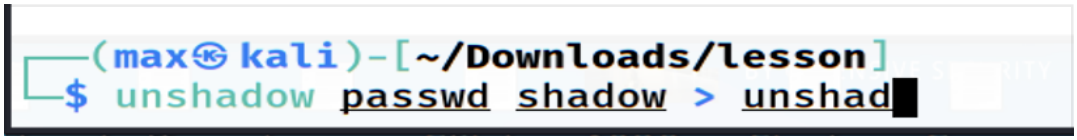


PASSWD, SHADOW CRACK IN LINUX:

1. create file with contents of /etc/passwd file, nam it passwd

2. create file with contents of /etc/shadow file, name it shadow

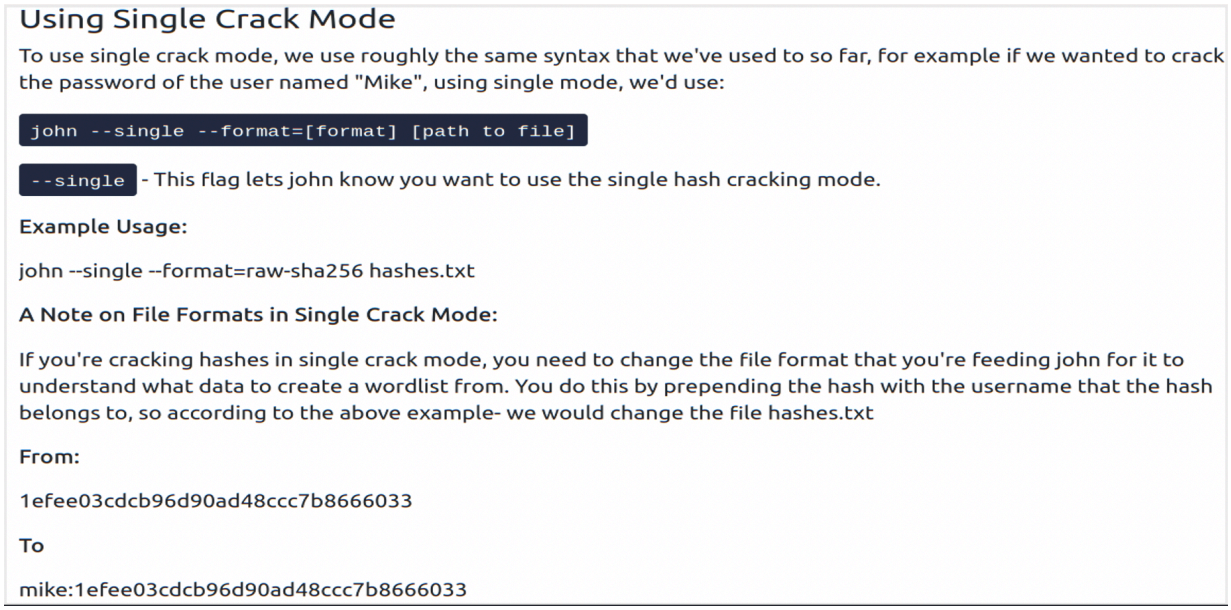
use unshadow CLI to put into usable form for john



CLI for cracking unshadowed files in john

```
┌──(max㉿kali)-[~/Downloads/lesson]
└─$ ls -la
total 20
drwxr-xr-x  2 max max 4096 May 14 11:25 .
drwxr-xr-x 11 max max 4096 May 14 11:17 ..
-rw-r--r--  1 max max   28 May 14 11:19 passwd
-rw-r--r--  1 max max  124 May 14 11:22 shadow
-rw-r--r--  1 max max  133 May 14 11:25 unshad
```

SINGLE MODE:



Using Single Crack Mode

To use single crack mode, we use roughly the same syntax that we've used to so far, for example if we wanted to crack the password of the user named "Mike", using single mode, we'd use:

```
john --single --format=[format] [path to file]
```

`--single` - This flag lets john know you want to use the single hash cracking mode.

Example Usage:

john --single --format=raw-sha256 hashes.txt

A Note on File Formats in Single Crack Mode:

If you're cracking hashes in single crack mode, you need to change the file format that you're feeding john for it to understand what data to create a wordlist from. You do this by prepending the hash with the username that the hash belongs to, so according to the above example- we would change the file hashes.txt

From:

1efee03cdcb96d90ad48ccc7b8666033

To

mike:1efee03cdcb96d90ad48ccc7b8666033

john —single —format=md5 hash.txt

CRACKING A ZIP FILE:(encryted)

Zip2John

Similarly to the unshadow tool that we used previously, we're going to be using the zip2john tool to convert the zip file into a hash format that John is able to understand, and hopefully crack. The basic usage is like this:

```
zip2john [options] [zip file] > [output file]
```

`[options]` - Allows you to pass specific checksum options to zip2john, this shouldn't often be necessary

`[zip file]` - The path to the zip file you wish to get the hash of

`>` - This is the output director, we're using this to send the output from this file to the...

`[output file]` - This is the file that will store the output from

Example Usage

zip2john zipfile.zip > zip_hash.txt

Cracking

We're then able to take the file we output from zip2john in our example use case called "zip_hash.txt" and, as we did with unshadow, feed it directly into John as we have made the input specifically for it.

```
john --wordlist=/usr/share/wordlists/rockyou.txt zip_hash.txt
```

CRACKING A RAR FILE:

Rar2John

Almost identical to the zip2john tool that we just used, we're going to use the rar2john tool to convert the rar file into a hash format that John is able to understand. The basic syntax is as follows:

```
rar2john [rar file] > [output file]
```

`rar2john` - Invokes the rar2john tool

`[rar file]` - The path to the rar file you wish to get the hash of

`>` - This is the output director, we're using this to send the output from this file to the...

`[output file]` - This is the file that will store the output from

Example Usage

rar2john rarfile.rar > rar_hash.txt

Cracking

Once again, we're then able to take the file we output from rar2john in our example use case called "rar_hash.txt" and, as we did with zip2john we can feed it directly into John..

```
john --wordlist=/usr/share/wordlists/rockyou.txt rar_hash.txt
```

extracting a password protected RAR file with ‘password’ as password

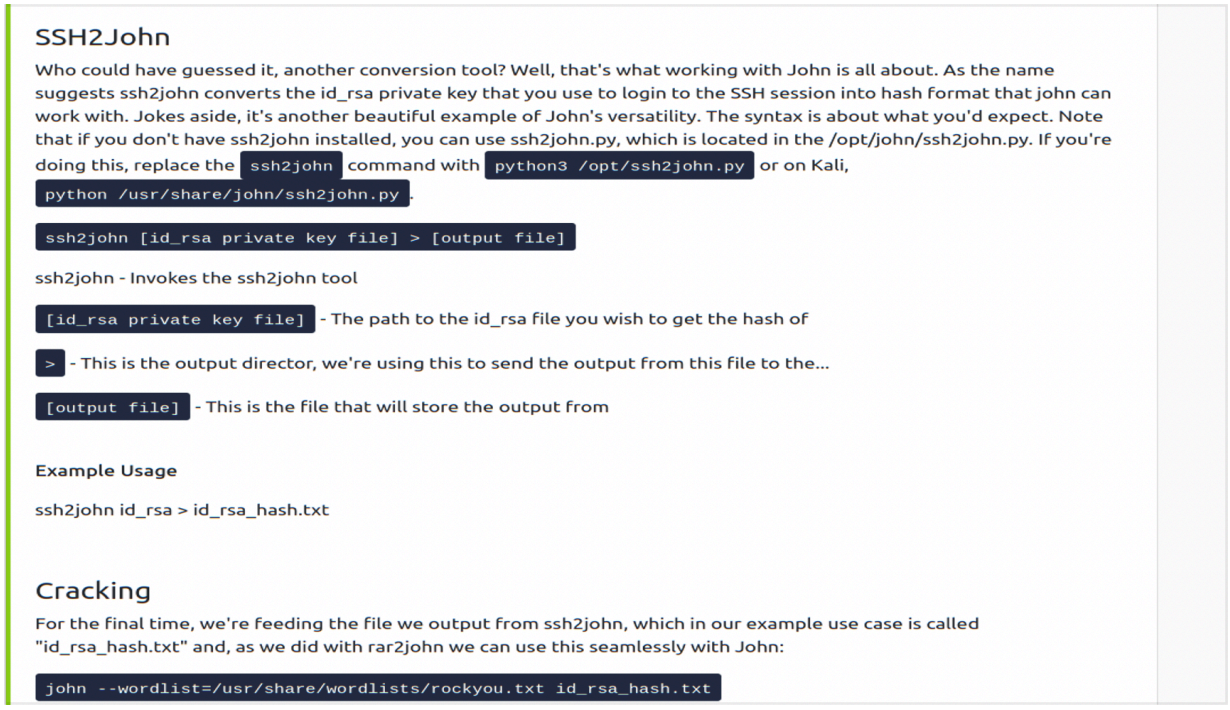
```
┌──(max㉿kali)-[~/Downloads]
└─$ unrar x -ppassword rar                                    7 ×

UNRAR 6.00 freeware      Copyright (c) 1993-2020 Alexander
Roshal

Extracting from rar

Extracting  flag.txt                                  OK
All OK
```

CRACKING SSH:



SSH2John

Who could have guessed it, another conversion tool? Well, that's what working with John is all about. As the name suggests ssh2john converts the id_rsa private key that you use to login to the SSH session into hash format that john can work with. Jokes aside, it's another beautiful example of John's versatility. The syntax is about what you'd expect. Note that if you don't have ssh2john installed, you can use ssh2john.py, which is located in the /opt/john/ssh2john.py. If you're doing this, replace the `ssh2john` command with `python3 /opt/ssh2john.py` or on Kali, `python /usr/share/john/ssh2john.py`.

```
ssh2john [id_rsa private key file] > [output file]
```

ssh2john - Invokes the ssh2john tool

`[id_rsa private key file]` - The path to the id_rsa file you wish to get the hash of

`>` - This is the output director, we're using this to send the output from this file to the...

`[output file]` - This is the file that will store the output from

Example Usage

ssh2john id_rsa > id_rsa_hash.txt

Cracking

For the final time, we're feeding the file we output from ssh2john, which in our example use case is called "id_rsa_hash.txt" and, as we did with rar2john we can use this seamlessly with John:

```
john --wordlist=/usr/share/wordlists/rockyou.txt id_rsa_hash.txt
```

1. convert id_rsa file to john readable (via python)

```
┌──(max㉿kali)-[~/Downloads]
└─$ sudo python /usr/share/john/ssh2john.py id_rsa > idjohn
```

2. crack idjohn file with john

```
┌──(max㉿kali)-[~/Downloads]
└─$ john --wordlist=/usr/share/wordlists/rockyou.txt idjohn
```