A cheat-sheet for password crackers

22 Dec 2014 - m3g9tr0n

Bash, Password Cracking, Hashcat, Security

In this article I am going to share some bash scripting commands and regular expressions which I find useful in password cracking. Most of the time, we find hashes to crack via shared pastes websites (the most popular of them being Pastebin.)

Isolating the hashes by hand can be a time consuming process; for that reason we are going to use regular expressions to make our life easier!

Extract md5 hashes

```
# egrep -oE '(^|[^a-fA-F0-9])[a-fA-F0-9]{32}([^a-fA-F0-9]|$)' *.txt | egrep -o '[a-fA-F0-9]{32}' > md5-hashes.txt
```

An alternative could be with sed

```
# sed -rn 's/.*[^a-fA-F0-9]([a-fA-F0-9]{32})[^a-fA-F0-9].*/1/p' *.txt > md5-hashes
```

> **Note:** The above regexes can be used for SHA1, SHA256 and other unsalted hashes represented in hex. The only thing you have to do is change the '{32}' to the corresponding length for your desired hash-type.

Extract valid MySQL-Old hashes

```
# grep -e "[0-7][0-9a-f]{7}[0-7][0-9a-f]{7}" *.txt > mysql-old-hashes.txt
```

Extract blowfish hashes

```
# grep -e "$2a\$\08\$(.){75}" *.txt > blowfish-hashes.txt
```

Extract Joomla hashes

```
# egrep -o "([0-9a-zA-Z]{32}):(w{16,32})" *.txt > joomla.txt
```

Extract VBulletin hashes

```
# egrep -o "([0-9a-zA-Z]{32}):(S{3,32})" *.txt > vbulletin.txt
```

Extraxt phpBB3-MD5

```
# egrep -o '$H$S{31}' *.txt > phpBB3-md5.txt
```

Extract Wordpress-MD5

```
# egrep -o '$P$S{31}' *.txt > wordpress-md5.txt
```

Extract Drupal 7

```
# egrep -o '$S$S{52}' *.txt > drupal-7.txt
```

Extract old Unix-md5

```
# egrep -o '$1$w{8}S{22}' *.txt > md5-unix-old.txt
```

Extract md5-apr1

```
# egrep -o '$apr1$w{8}S{22}' *.txt > md5-apr1.txt
```

Extract sha512crypt, SHA512(Unix)

```
# egrep -o '$6$w{8}S{86}' *.txt > sha512crypt.txt
```

Extract e-mails from text files

```
# grep -E -o "\b[a-zA-Z0-9.#?$*_-]+@[a-zA-Z0-9.#?$*_-]+.[a-zA-Z0-9.-]+\b" *.txt > e-mails.txt
```

Extract HTTP URLs from text files

```
# grep http | grep -shoP 'http.*?[" >]' *.txt > http-urls.txt
```

For extracting HTTPS, FTP and other URL format use `# grep -E '(((https|ftp|gopher)|mailto)[.:][^ >"  ]*|www.[-a-z0-9.]+)[^ .,;  >">):]' *.txt > urls.txt`

> **Note:** if grep returns "Binary file (standard input) matches" use the following approaches `# tr '[\000-\011\013-\037177-377]' '.' < *.log | grep -E "Your_Regex"` OR `# cat -v *.log | egrep -o "Your_Regex"`

Extract Floating point numbers

```
# grep -E -o "^[-+]?[0-9]*.?[0-9]+([eE][-+]?[0-9]+)?$" *.txt > floats.txt
```

Extract credit card data

Visa `# grep -E -o "4[0-9]{3}[ -]?[0-9]{4}[ -]?[0-9]{4}[ -]?[0-9]{4}" *.txt > visa.txt`

MasterCard `# grep -E -o "5[0-9]{3}[ -]?[0-9]{4}[ -]?[0-9]{4}[ -]?[0-9]{4}" *.txt > mastercard.txt`

American Express `# grep -E -o "\b3[47][0-9]{13}\b" *.txt > american-express.txt`

Diners Club `# grep -E -o "\b3(?:0[0-5]|[68][0-9])[0-9]{11}\b" *.txt > diners.txt`

Discover `# grep -E -o "6011[ -]?[0-9]{4}[ -]?[0-9]{4}[ -]?[0-9]{4}" *.txt > discover.txt`

JCB `# grep -E -o "\b(?:2131|1800|35d{3})d{11}\b" *.txt > jcb.txt`

AMEX `# grep -E -o "3[47][0-9]{2}[ -]?[0-9]{6}[ -]?[0-9]{5}" *.txt > amex.txt`

Extract Social Security Number (SSN)

```
# grep -E -o "[0-9]{3}[ -]?[0-9]{2}[ -]?[0-9]{4}" *.txt > ssn.txt
```

Extract Indiana Driver License Number

```
# grep -E -o "[0-9]{4}[ -]?[0-9]{2}[ -]?[0-9]{4}" *.txt > indiana-dln.txt
```

Extract US Passport Cards

```
# grep -E -o "C0[0-9]{7}" *.txt > us-pass-card.txt
```

Extract US Passport Number

```
# grep -E -o "[23][0-9]{8}" *.txt > us-pass-num.txt
```

Extract US Phone Numberss

```
# grep -Po 'd{3}[s-_]?d{3}[s-_]?d{4}' *.txt > us-phones.txt
```

Extract ISBN Numbers

```
# egrep -a -o "\bISBN(?:-1[03])?:? (?=[0-9X]{10}$|(?=(?:[0-9]+[- ]){3})[- 0-9X]{13}$|97[89][0-9]{10}$|(?=(?:[0-9]+[- ]){4})[- 0-9]{17}$)(?:97[89][- ]?)?[0-9]{1,5}[- ]?[0-9]+[- ]?[0-9]+[- ]?[0-9X]\b" *.txt > isbn.txt
```

WordList Manipulation

Remove the space character with sed

```
# sed -i 's/ //g' file.txt
```
OR
```
# egrep -v "^[[:space:]]*$" file.txt
```

Remove the last space character with sed

```
# sed -i s/.$// file.txt
```

Sorting Wordlists by Length

```
# awk '{print length, $0}' rockyou.txt | sort -n | cut -d " " -f2- > rockyou_length-list.txt
```

Convert uppercase to lowercase and the opposite

```
# tr [A-Z] [a-z] < file.txt > lower-case.txt
# tr [a-z] [A-Z] < file.txt > upper-case.txt
```

Remove blank lines with sed

`# sed -i '/^$/d' List.txt`

Remove defined character with sed

`# sed -i "s/'//" file.txt`

Delete a string with sed

`# echo 'This is a foo test' | sed -e 's/<foo>//g'`

Replace characters with tr

`# tr '@' '#' < emails.txt` OR `# sed 's/@/#' file.txt`

Print specific columns with awk

`# awk -F "," '{print $3}' infile.csv > outfile.csv` OR `# cut -d "," -f 3 infile.csv > outfile.csv`

> **Note:** if you want to isolate all columns after column 3 use `# cut -d "," -f 3- infile.csv > outfile.csv`

Generate Random Passwords with urandom

```
# tr -dc 'a-zA-Z0-9._!@#$%^&*()' < /dev/urandom | fold -w 8 | head -n 500000 > wordlist.txt
# tr -dc 'a-zA-Z0-9-_!@#$%^&*()_+{}|:<>?=' < /dev/urandom | fold -w 12 | head -n 4
# base64 /dev/urandom | tr -d '[^:alnum:]' | cut -c1-10 | head -2
# tr -dc 'a-zA-Z0-9' < /dev/urandom | fold -w 10 | head -n 4
# tr -dc 'a-zA-Z0-9-_!@#$%^&*()_+{}|:<>?=' < /dev/urandom | fold -w 12 | head -n 4 | grep -i '[!@#$%^&*()_+{}|:<>?
```

```
=]'
# tr -dc '[:print:]' < /dev/urandom | fold -w 10| head -n 10
# tr -cd '[:alnum:]' < /dev/urandom | fold -w30 | head -n2
```

Remove Parenthesis with tr

```
# tr -d '()' < in_file > out_file
```

Generate wordlists from your file-names

```
# ls -A | sed 's/regexp/&
/g'
```

Process text files when cat is unable to handle strange characters

```
# sed 's/([[:alnum:]]*)[[:space:]]*(.)(..*)/12/' *.txt
```

Generate length based wordlists with awk

```
# awk 'length == 10' file.txt > 10-length.txt
```

Merge two different txt files

```
# paste -d' ' file1.txt file2.txt > new-file.txt
```

Faster sorting

```
# export alias sort='sort --parallel=<number_of_cpu_cores> -S <amount_of_memory>G ' && export LC_ALL
='C' && cat file.txt | sort -u > new-file.txt
```

Mac to unix

```
# tr '\015' '\012' < in_file > out_file
```

Dos to Unix

```
# dos2unix file.txt
```

Unix to Dos

```
# unix2dos file.txt
```

Remove from one file what is in another file

```
# grep -F -v -f file1.txt -w file2.txt > file3.txt
```

Isolate specific line numbers with sed

```
# sed -n '1,100p' test.file > file.out
```

Create Wordlists from PDF files

```
# pdftotext file.pdf file.txt
```

Find the line number of a string inside a file

```
# awk '{ print NR, $0 }' file.txt | grep "string-to-grep"
```

Faster filtering with the silver searcher

https://github.com/ggreer/the_silver_searcher

For faster searching, use all the above grep regular expressions with the command ag. The following is a proof of concept of its speed:

```
# time ack-grep -o "\b[a-zA-Z0-9.#?$*_-]+@[a-zA-Z0-9.#?$*_-]+.[a-zA-Z0-9.-]+\b" *.txt > /dev/null
real    1m2.447s
user    1m2.297s
sys 0m0.645s

# time egrep -o "\b[a-zA-Z0-9.#?$*_-]+@[a-zA-Z0-9.#?$*_-]+.[a-zA-Z0-9.-]+\b" *.txt > /dev/null
real    0m30.484s
user    0m30.292s
sys 0m0.310s

# time ag -o "\b[a-zA-Z0-9.#?$*_-]+@[a-zA-Z0-9.#?$*_-]+.[a-zA-Z0-9.-]+\b" *.txt > /dev/null
real    0m4.908s
user    0m4.820s
sys 0m0.277s
```

Useful Use of Cat

Contrary to what many veteran unix users may believe, this happens to be one of the rare opportunities where using cat can actually make your searches *faster*. The SilverSearcher utility is (at the time of this writing) not quite as efficient as cat when it comes to reading from file handles. Therefore, you can pipe output from **cat** into **ag** to see nearly a 2x real time performance gain:

```
$ time ag -o '(^|[^a-fA-F0-9])[a-fA-F0-9]{32}([^a-fA-F0-9]|$)' *.txt | ag -o '[a-fA-F0-9]{32}' > /dev/null

real    0m10.851s
user    0m13.069s
sys 0m0.092s

$ time cat *.txt | ag -o '(^|[^a-fA-F0-9])[a-fA-F0-9]{32}([^a-fA-F0-9]|$)' | ag -o '[a-fA-F0-9]{32}' > /dev/null

real    0m6.689s
user    0m7.881s
sys 0m0.424s
```

About the Author

m3g9tr0n is a security analyst and a member of Team Hashcat.