Cracking 936 Million Passwords

Jeff Deifik
jeff@deifik.com

About Me

- MS in Cybersecurity, CISSP, C|CISO
- Software for first e-commerce system (from 1985-1995)
- Software for the first orbiting radio telescope satellite
- Software for the most advanced pulse oximeter
- Cybersecurity for government satellite ground control, balancing sound cybersecurity with cost and schedule. Currently employed at The Aerospace Corp.
- Interest in the intersection of cybersecurity and software development began with white hat password cracking over 30 years ago.

Cracking 936,504,299 Passwords

- Dump from Have I Been Pwned
- Good news – they are NTLM format
- Bad news – 936,000,000
- This requires a Big Data approach and lots of RAM
- Started with 128gb and went to 256gb
  - Generally needs server grade hardware for lots of RAM
- Limited RAM means I could only run a few threads at the beginning
- I have found 92.0%

Tools

- John The Ripper
  - Infrequent official releases, Many unofficial releases
  - Poor Graphical Processor Unit (GPU) windows support
  - Easy to make custom rules
  - Good mailing list support
- HashCat
  - 6.2.6 latest release Sep 2022 ☹
  - Great GPU acceleration
  - Primitive rule syntax
  - Dictionary attacks takes a lot of memory

Wordlists

- Some very high quality
- Most stuffed full of junk and require editing
  - Very long lines, often thousands of characters long
  - Non ASCII letters
  - Separators that are not newlines
  - Since they are big, specialized tools are needed
- Rockyou2021 is a bit big, but very high quality

My Custom Password Tools

- short -s 40 foo > foo.40 short -l 41 foo > foo.41
  - splits foo into 2 files, 40 chars and shorter, and 41 chars and longer
- msort -l foo > foo.l - Sorts foo by line length
- ascii-lines -p foo > foo.p
  - only outputs lines of foo compromised solely of printable ascii characters
- multi-merge foo.1 foo.2 foo.3 > foo.123
  - merges any number of sorted files into a big sorted file
- sample -10000 foo > foo.10k
  - outputs one line every 10000 lines, for sampling foo
- line_len foo - Prints line length counts
- count foo - Character frequency count
- pw_stats – Shows stasicists on passwords
- pw_unhex – Removes hex encoding from found passwords

Standard Wordlist Tools

- gnu sort
  - You generally want to process sorted wordlists
  - Works with files bigger than RAM using tmp files
- uniq
  - Remove duplicate words
- comm
  - Removes duplicate words in different files
- emacs
  - The one true editor, regular expressions, can process gigabyte files

Relative Hashing Speed

- NTLM Speed 41,825.0 MH/s ☺
- md5 Speed 24,943.1 MH/s
- LM Speed 18,382.7 MH/s
- descrypt Speed 906.7 MH/s
- SHA1 Speed 788.2 MH/s
- scrypt Speed 435.1 kH/s
- WPA2 Speed 396.8 kH/s
- bcrypt Speed 13094 H/s
  https://gist.github.com/epixoip/a83d38f412b4737e99bbef804a270c40

Salt

- 1979 - Unix 12 bits, 4,096 different salts

  https://spqr.eecs.umich.edu/courses/cs660sp11/papers/10.1.1.128.1635.pdf
- 1980’s - Unix 48 bits, 281,474,976,710,656
- 1996 - bcrypt 128 bits, 3.4 x 10^38 salts
- Argon2 128 bits, 3.4 x 10^38 salts
- Descrypt uses 12 bits of salt
- LM and NTLN doesn’t use salt ☺

How to Crack

- Dictionaries - very efficient
- Brute force attack - very powerful, but slow and doesn't scale
  - 8 chars upper lower number - 18.340,105,584,896 @229 days on 3060ti
  - 12 char upper lower number - 3,226,266,762,397,899,821,056 @12,791,288 years
  - 16 char upper lower number - 47,672,401,706,823,533,450,263,330,816 years
- Rule based attack

Rainbow Tables

- Doesn’t play nice with salt
- Very very fast ☺
- Works with LM, NTLM, MD5, etc.
- Defcon data duplication village – 6tb drives
  - freerainbowtables.com GSM A51 and MD5 hash tables
  - more rainbowtables, lanman, mysqlsha1, ntlm, and some word lists
- Best used with a small number of hashes

Starting to Crack - Using Rainbow Crack

- I tried Rainbow Crack 1.8
- Used NTLM loweralpha-space 9 char rainbow table
  - 43 gigabytes
- Unable to get it working
  - Complex process to convert downloaded tables to rainbow table
  - Unable to crack a known hash
  - **Uses 160kbytes per hash ☹**
  - **Therefore on 936m passwords, 150,000 gigabytes RAM required ☹**
  - Contacted project rainbow crack Sep 26 – no response

Starting to Crack - Using Rainbow Crack

- I tried rcracki_mt (0.7.0) (works with rti2 files)
  - It actually works, unlike rainbow crack
- Used NTLM loweralpha-space 9 char rainbow table
  - 35 gigabytes

Takes 6 seconds per file (SATA SSD), 84 files (504 sec per hash)

- 900m passwords will take 16,000 years ☹
- Good for cracking a few passwords, bad for millions

Starting to crack - Using Hashcat

- My hashcat machines has 16gb of ram.
- When I ran hashcat on 1m passwords:

hashcat.exe -m 1000 ..\pwned_pw_pruned_ntlm.rawest.1m ..\dictionaries\rock.dic (3.9mbyte dictionary)

Host memory required for this attack: 667 MB

Therefore on 936m passwords, 624 gigabytes RAM required ☹

Starting to crack - Using JTR rules

- Started using JTR / default dictionary & rules
- Using –fork option consumes a lot of ram – typically 30gb per fork
- Upgraded from 128gb to 256gb
- Running 6 forks currently
- Found 487,193,352 passwords in 12 days
- Lots more work to do

More JTR

- JTR default dictionary and rules
  - Found 154m passwords
- JTR incremental attack (which never finishes)
  - Total found 325m passwords
- JTR using rockyou2021 wordlist
  - Found 156m passwords (already found with incremental ☹)
- Got total 256gb ram
- JTR –fork=6 default wordlist & rules
  - Found 15m more passwords

More JTR

- JTR --fork=7 apply rules twice
  - Found 36m more passwords
- JTR –fork=8 apply rules to rockyou2021
  - Found 265m passwords in less than an hour ☺
- JTR --fork=18 rules on rockyou2021
  - Now we can use more threads, as only 140m unfound passwords
  - found @11m passwords in 3 days

More JTR

- JTR brute force lower/number up to len=9
  - Brute force all lower/number up to 9 len
  - found @3.6m passwords in @4 days
- JTR rules using 811m found passwords as dictionary
  - Found 16m passwords in @7 days
  - Will take years to finish ☹
- JTR apply rules twice on 811m found passwords
  - Will take years to finish ☹

More JTR – control characters

- john.exe --fork=10 --format=NT --verbosity=2 --no-log --wordlist=\pw-crack\dictionaries\rockyou2021.dic --rules=rep_control_1 \pw-crack\pwn_ntlm.129m.rawest
  - Replace a control char into rockyou2021
- john.exe --fork=22 --format=NT --verbosity=2 --no-log --wordlist=\pw-crack\dictionaries\rockyou2021.dic --rules=ins_control_1 \pw-crack\pwn_ntlm.115m.rawest
  - Insert a control char into rockyou2021
  - Found 8m (@105k tabs, @7.9m cr)

More JTR – control char rules

From solar designer:

```
# Overstrike any one character
[List.Rules:rep_control_1]
# Trivial
# o[0-9A-Z][\x7f\x80\x01-\x1f]
# Optimized
->\r[1-9A-ZZ] >\p[0-9A-Z] o\0[\x7f\x80\x01-\x1f] Q

# Insert any one character
[List.Rules:ins_control_1]
# Trivial
# i[0-9A-Z][\x7f\x80\x01-\x1f]
# Optimized
->\r[2-9A-ZZZ] >\p1[0-9A-Z] i\0[\x7f\x80\x01-\x1f]
```

Hashcat

- Brute force attack
  - lower, upper, number, special — len 7 3.7 days
  - lower, upper, number — len 8 @10 days
    - 6m passwords
  - lower, number — len 9 5.3 days
    - 1.9m passwords
  - upper, number — len 9 5.3 days
    - 1.1m passwords found
  - Lower, number — len 10 – 180 days
    - 2.46m passwords found

Password Statistics on 870.0m - Length

Length:

1: 0.0 % (275)	2: 0.0 % (10220)	3: 0.0 % (203268)
4: 0.1 % (1,164 k)	5: 0.7 % (6,108 k)	6: 5.5 % (47,769 k)
7: 8.9 % (77,405 k)	8: 27.6 % (239,920 k)	9: 14.3 % (124,538 k)
10: 16.3 % (141,422 k)	11: 8.3 % (72,074 k)	12: 5.9 % (50,928 k)
13: 3.7 % (32,434 k)	14: 2.6 % (22,281 k)	15: 2.7 % (23,397 k)
16: 1.6 % (13,588 k)	17: 0.6 % (5,192 k)	18: 0.5 % (4,240 k)
19: 0.3 % (2,753 k)	20: 0.2 % (2,161 k)	21: 0.1 % (723 k)
22: 0.1 % (597406)	23: 0.0 % (340760)	24: 0.0 % (336319)
25: 0.0 % (183047)	26: 0.0 % (187938)	27: 0.0 % (114836)
28: 0.0 % (33)	29: 0.0 % (1)	30+: 0.0 % (104)

Password Statistics on 870.0m - Characters

all lower: 18.9 % (164,525 k)
all upper: 0.7 % (6,002 k)
all digit: 8.1 % (70,238 k)
all special: 0.0 % (41325)
all lower digit: 45.3 % (393,994 k)
all lower upper: 2.4 % (20,937 k)
all lower upper digit: 11.5 % (100,450 k)
all lower special: 2.5 % (21,995 k)
all upper digit: 2.7 % (23,631 k)
all digit special: 0.4 % (3,631 k)
all lower upper special: 0.4 % (3,704 k)
all lower digit special: 4.3 % (37,256 k)
all lower upper digit special: 2.3 % (19,851 k)
Has control char: 0.0 % (271 k)
Has 8 bit asciil: 0.0 % (130 k)

Password Statistics on 870.0m – String Classes

String Classes:

All alpha: 22.0 % (191,465 k)

Alphas + Numbers: 35.1 % (305,622 k)

Numbers + Alphas: 7.0 % (6,0765 k)

Alphas + Specials: 0.7 % (5,767 k)

Alphas + Numbers + Alphas: 6.9 % (60,269 k)

Numbers + Alphas + Numbers: 2.3 % (20,149 k)

Alphas + Specials + Alphas: 1.8 % (15,430 k)

Control chars in passwords

nul [0]=751	soh [1]=1098	stx [2]=722	etx [3]=1003
eot [4]=1555	enq [5]=786	ack [6]=601	bel [7]=637
bs [8]=783	ht [9]=134 k	lf [10]=8	vt [11]=545
ff [12]=804	cr [13]=29,280 k	so [14]=1234	si [15]=571
dle [16]=574	dc1 [17]=494	dc2 [18]=612	dc3 [19]=604
dc4 [20]=254	nak [21]=261	syn [22]=669	etb [23]=815
can [24]=606	em [25]=691	sub [26]=786	esc [27]=616
fs [28]=638	gs [29]=530	rs [30]=526	us [31]=742
del [127]=1046			

Defense

- Don’t use NTLM
- 2 factor authentication
  - What you have - Titan security key, yubikey, smartcard
  - What you are - Fingerprint, Face ID
- Use cryptographically strong random passwords
- Use a password manager
  - keepass, 1password, bitwarden
- I wrote a password generator, here is some output:

password is K)dE;pN%(]R~H6L-11!R bits 129
password is GAw->8k?+Qou#(*#L:Z0 bits 129
password is YmytLWazQ[g{0R@}l2ha bits 129
password is _a^W9h8[J~jsO)*6ahaQ bits 129
password is [q;)y_):BTJAfHZU)7.* bits 129

Other Stuff

- You will want to undervolt / underclock your GPU to save power
  - MSI Afterburner works well, windows specific

https://www.openwall.com/presentations/OffensiveCon2024-Password-Cracking/

https://jakewnuk.com/static/BsidesCaymanIslands2023%20-%20Leveling%20Up%20Password%20Attacks%20with%20Breach%20Data.pdf

Dictionaries

47,085,595 linked.dic
72,382,568 SkullSecurityComp.dic
93,559,564 10-million-passwords.dic
94,461,698 ignis-10M.dic
139,749,969 10-million-user-pass.dic
139,921,988 rockyou.dic
362,881,958 hk_hlm_founds.dic
382,000,913 collection_1_5_v3.dic
1,075,899,306 superpass_fixed.dic
1,305,699,616 facebook-lastnames.dic.l33t
1,643,295,189 kac.dic
2,266,396,047 Super_mega_dic.dic
2,277,681,952 exploit.in.dic
3,107,889,706 thedefinitvepasswordlist_complete_.dic
4,276,546,161 HashesOrg.dic
5,403,987,782 hibp_515_found.dic
11,432,450,014 b0n3z.dic
13,675,962,135 hashesorg2019.dic
13,832,356,359 crackstation_fixed.dic
17,264,739,583 Md5decrypt-awesome-wordlist.dic
17,539,451,065 collection_1_5_v1.dic
17,868,066,068 DCHTPassv1.0.dic
18,166,067,612 naxxatoe-dict-total-new-unsorted.dic
18,624,885,828 HYPER-WORDLIST-DIC.dic
21,102,866,314 b0n3z-sorted-wordlist.dic
37,241,758,679 weakpass_2a.dic
41,514,529,952 collection_1_5_v2.dic
98,378,212,907 rockyou2021.dic
123,968,583,755 WordlistBySheez_v8.dic