Password Cracking

by sofia filippatou

Why do we still use passwords today ?

- **Basic Concept:** Most people understand the basic concept of a password
- **Widespread Use:** Passwords are used across a vast array of platforms and services
- **Accessibility:** Passwords do not require specialized equipment or technology, making them accessible to a wide range of users

TIME IT TAKES A HACKER TO BRUTE FORCE YOUR PASSWORD IN 2024

How did we make this? Learn at hivesystems.com/password

Number of Characters	Numbers Only	Lowercase Letters	Upper and Lowercase Letters	Numbers, Upper and Lowercase Letters	Numbers, Upper and Lowercase Letters, Symbols
4	Instantly	Instantly	3 secs	6 secs	9 secs
5	Instantly	4 secs	2 mins	6 mins	10 mins
6	Instantly	2 mins	2 hours	6 hours	12 hours
7	4 secs	50 mins	4 days	2 weeks	1 month
8	37 secs	22 hours	8 months	3 years	7 years
9	6 mins	3 weeks	33 years	161 years	479 years
10	1 hour	2 years	1k years	9k years	33k years
11	10 hours	44 years	89k years	618k years	2m years
12	4 days	1k years	4m years	38m years	164m years
13	1 month	29k years	241m years	2bn years	11bn years
14	1 year	766k years	12bn years	147bn years	805bn years
15	12 years	19m years	652bn years	9tn years	56tn years
16	119 years	517m years	33tn years	566tn years	3qd years
17	1k years	13bn years	1qd years	35qd years	276qd years
18	11k years	350bn years	91qd years	2qn years	19qn years

HIVE SYSTEMS

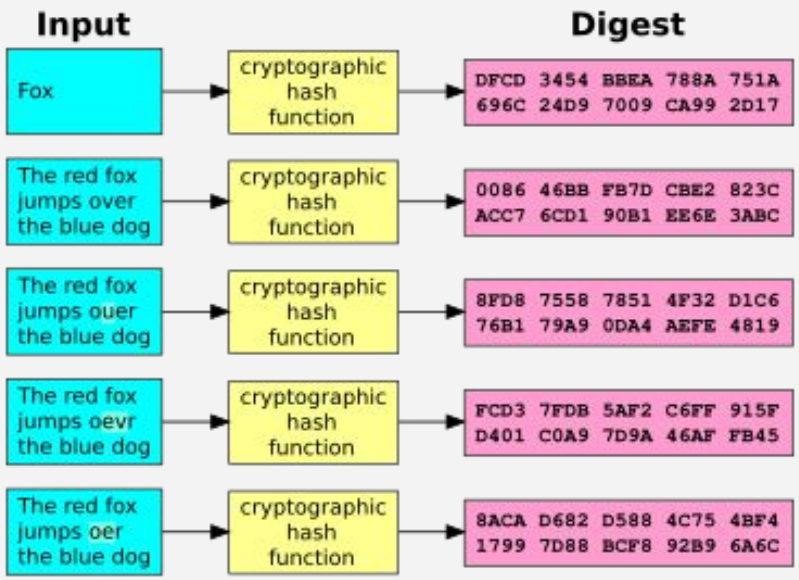
› Hardware: 12 x RTX 4090 | Password hash: bcrypt

Example

Username	Password
Alice	MySecurePassword123
Bob	MySecurePassword123
Charlie	Password123
Dana	MySecurePassword123
Eve	UniquePassword456

What do you find wrong with this way of storing passwords ?

(Cryptographic) Hash Functions


Input
Digest
Fox
cryptographic hash function
DFCD 3454 BBEA 788A 751A 696C 24D9 7009 CA99 2D17
The red fox jumps over the blue dog
cryptographic hash function
0086 46BB FB7D CBE2 823C ACC7 6CD1 90B1 EE6E 3ABC
The red fox jumps ouer the blue dog
cryptographic hash function
8FD8 7558 7851 4F32 D1C6 76B1 79A9 0DA4 AEFE 4819
The red fox jumps oevr the blue dog
cryptographic hash function
FCD3 7FDB 5AF2 C6FF 915F D401 C0A9 7D9A 46AF FB45
The red fox jumps oer the blue dog
cryptographic hash function
8ACA D682 D588 4C75 4BF4 1799 7D88 BCF8 92B9 6A6C

What are the **properties** of hash functions ?

1. **NO collisions:** Computationally infeasible to find two different inputs that produce the same hash output

2. **One-way:** NOT reversible, NO decrypt

3. **Deterministic:** Same input <-> Same output

4. **Uniform distribution:** The outputs are random, unpredictable

Username	Password	SHA1
Alice	MySecurePassword123	e46bdba4607b7ee7cb6d1eff9105ba17118e4a3c
Bob	MySecurePassword123	e46bdba4607b7ee7cb6d1eff9105ba17118e4a3c
Charlie	Password123	b2e98ad6f6eb8508dd6a14cfa704bad7f05f6fb1
Dana	MySecurePassword123	e46bdba4607b7ee7cb6d1eff9105ba17118e4a3c
Eve	UniquePassword456	1ea138ca05845750bd58c8c64894ae7c93b4487b

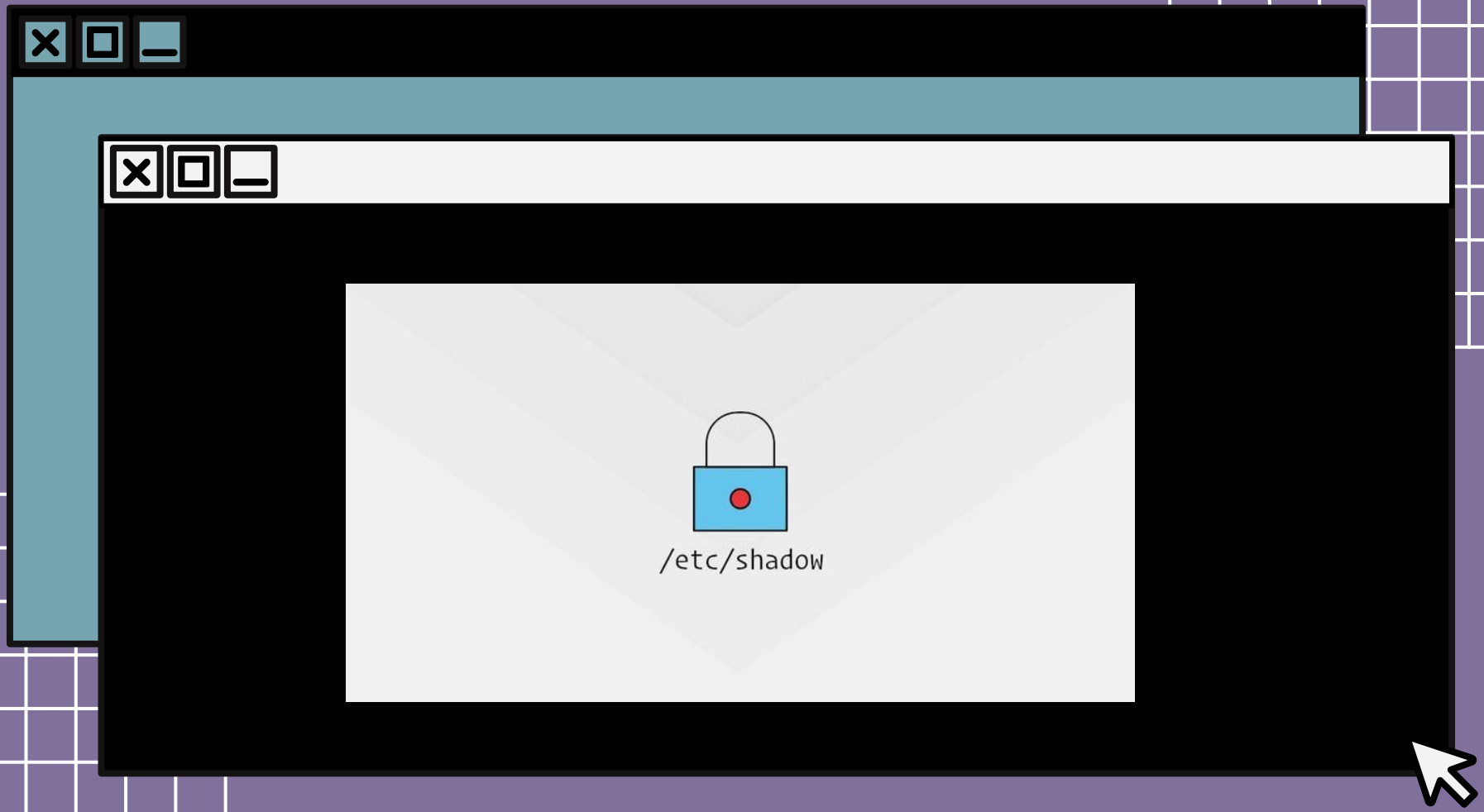
What is salting ?

What is salting ?

Salting is essentially the addition of random data before it is put through a hash function, and they are most commonly used with passwords.

After Salting

User ID	Username	Password	SHA-1
AAA	Alice	MySecurePassword123	2368bd16a9b6a473e52234d0109eced13de0ae99
AAB	Bob	MySecurePassword123	094bfcc7f4ad32073f8c98dde4b52b66619f7b2f
ABB	Charlie	Password123	85befacd1bc957a551c24ba56eeef702c0e2fb12
BBB	Dana	MySecurePassword123	d0e2308b17de1c920ac48f571daf39190ff76181
ABA	Eve	UniquePassword456	52bae796d90f08503259a860ba57f0282e76e9be


/etc/shadow

sudo cat /etc/shadow

It is a file on Unix and Linux systems that stores user account information, including hashed passwords.

Source: https://linuxize.com/post/etc-shadow-file/

idsalt$encrypted

```
iodine:!:19953::::::
miredo:!:19953::::::
statd:!:19953::::::
redis:!:19953::::::
postgres:!:19953::::::
mosquitto:!:19953::::::
inetsim:!:19953::::::
_gvm:!:19953::::::
kali:$y$j9T$zY1oKFxJlTgP2WcJhzbNl1$xhkUmB8R9fzETc/1kgL/nOPcWFTvhn17clxXCgyFjpC:19953:0:99999:7:::
s:$y$j9T$pFd3FVBO5cJ4/c.Pq.zD50$8kkKv99/Bsxl37LUbuGnyFNdvC0W0JTTPyAgxJXaEw5:20060:0:99999:7:::
```

Hash-identifier: https://hashes.com/en/tools/hash_identifier

```
mark:$6$.n.:17736:0:99999:7:::
[--] [----] [---] - [---] ----
|      |      |   |   |   |||+-----------> 9. Unused
|      |      |   |   |   ||+------------> 8. Expiration date
|      |      |   |   |   |+-------------> 7. Inactivity period
|      |      |   |   |   +--------------> 6. Warning period
|      |      |   |   +------------------> 5. Maximum password age
|      |      |   +----------------------> 4. Minimum password age
|      |      +--------------------------> 3. Last password change
|      +----------------------------------> 2. Encrypted Password
+-------------------------------------------> 1. Username
```

y (or 7) is yescrypt

Source: https://www.baeldung.com/linux/shadow-passwords

Password cracking using John

Example

1. Open terminal
2. Download:
3. (Do `sudo apt-get install john`)
4. Do `unshadow passwd-copy.txt shadow-copy.txt > crack.txt`
5. Remove everything from the file except for the users
6. Do `john crack.txt`
7. What do you see ?

```
┌──(kali㉿kali)-[~]
└─$ unshadow passwd-copy shadow-copy > crack.txt

┌──(kali㉿kali)-[~]
└─$ john crack.txt
Using default input encoding: UTF-8
Loaded 1 password hash (HMAC-SHA256 [password is key, SHA256 256/256 AVX2 8x])
Will run 2 OpenMP threads
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Warning: Only 2 candidates buffered for the current salt, minimum 16 needed for performance.
Warning: Only 5 candidates buffered for the current salt, minimum 16 needed for performance.
Almost done: Processing the remaining buffered candidate passwords, if any.
Proceeding with wordlist:/usr/share/john/password.lst
Proceeding with incremental:ASCII
```

More things to crack..

- Windows Password Hashes
- PDF Files
- ZIP and RAR Archives

Wordlists

Wordlists

A wordlist is a text file that contains a list of words or phrases, typically one per line. These words can be anything from common passwords, dictionary words, usernames, or even phrases that people might use in their passwords.

Password cracking using

```
hashcat -m 0 -a 0 hash.txt /usr/share/wordlists/rockyou.txt.tar.gz -O
```

- **hashcat:** Main command to run the program (on Windows, use hashcat.exe)
- **-m 0:** Hash type (In this case, 0 represents MD5)
- **-a 0:** Attack mode (In this case, 0 stands for "straight" mode, a dictionary attack)
- **hash.txt:** This file contains the hash of hashes you're trying to crack
- **/usr/share/wordlists/rockyou.txt.tar.gz:** This is the path to your dictionary or list of potential passwords
- **-O**: This flag enables optimized kernels. It instructs Hashcat to use optimized code paths, potentially improving performance on some systems.

tip: if pass was found once, you can show it again by adding --show to the end of the command

Source: https://www.stationx.net/how-to-use-hashcat/

Final Thoughts

Final Thoughts

- Password cracking is a critical aspect of cybersecurity that involves the process of recovering passwords from data that has been stored in a hashed or encrypted format.
- However, every password can be cracked given sufficient time and computational power.

Today we'll use

- https://tryhackme.com/r/room/crackthehash
- https://tryhackme.com/r/room/crackthehashlevel2


THANKS FOR YOUR
"ATTENTION"