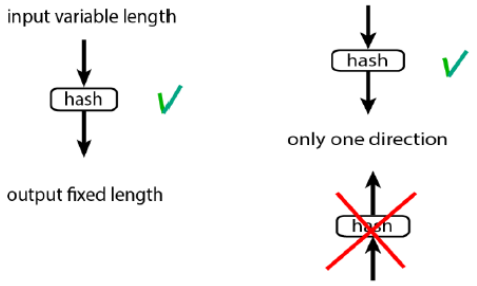
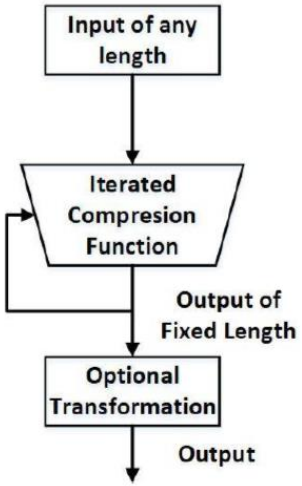
MD5 Rainbow tables

Hash function	Digest length	Secure?
MD2	128 bits	No
MD4	128 bits	No
MD5	128 bits	No
SHA-1	160 bits	No
SHA-256	256 bits	Yes



Hashes

- Hash is a function that :
  - Accepts input of any length
  - Produces output of fixed length
  - Is deterministic
- Additional properties
  - H(x) is relatively easy to compute
  - Pre-image resistance (1-way function)
    - For a given y is computationally infeasible to find x, such that H(x)=y
  - Collision-free
    - It is computationally infeasible to find such x and x', where x <> x', that H(x) = H(x')



Hashes

- Input is split into blocks of fixed length
- Each block is then passed to the compress function along with output from last iteration
- Repeats until all blocks are processed
- Outputs hash value of fixed length

Name	Surname	Login	Password
Edwin	Franco	edwinf	8257e9022bb55b5a8d80427e1d434712
Yeaz	Jaddoo	yeazj	bdc87b9c894da5168059e00ebffb9077

Hashes – Salting

- Prevention of guessing attack
- Stored without salting

Name	Surname	Login	Password
Yeaz	Jaddoo	yeazj	bdc87b9c894da5168059e00ebffb9077
Edwin	Franco	edwinf	bdc87b9c894da5168059e00ebffb9077

- Salt randomizes the hash

Name	Surname	Login	Password	Salt
Yeaz	Jaddoo	yeazj	2e3a83257957bffd0dbaea89d61684dc	1
Edwin	Franco	edwinf	7e37cdf7329e00a28b613cc817589c37	e

Hashes – Not a solution for everything!

Is our password safe if it is hashed on the server?

There is still plenty of possibilities how to steal your password:

- Online guessing attack–Attacker simply tries to guess the password
- Social engineering/phishing – Unwittingly reveals the password to the attacker
- Eavesdropping – Attacker intercepts the password from the network traffic
- Malware – Captures the passwords and sends it to the attacker (keyloggers)

Defenses

- Use encrypted connection to server
- Use 2FA
- Deploy security software like AV

MD5 (Message Digest)

First published in 1992

Digest Length (Output) – 128bit

Block size – 512bit

Still popular

- **Eatigo** breach (Oct 2018) – 2.8M accounts, unsalted MD5 hashes
- **Creative** breach (May 2018) – 500k accounts, salted MD5 hashes
- **123RF** breach (Mar 2020) – 8M accounts, salted MD5 hashes

Suffers from extensive vulnerabilities

Can be cracked by **bruteforce attack**, or with use of **Rainbow Tables**

Attacks on Hashes

Brute Force

- The most simple and straight forward attack
- Calculates a hash on every attempt => Slow

Look-up Table

- Table of precomputed hashes
- Fastest, but requires a lot of storage resources

Rainbow Table

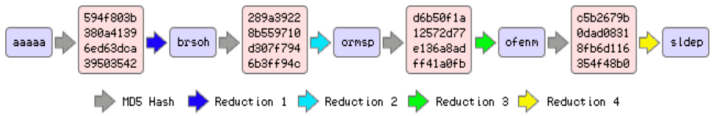
- Trade-off between time and space complexity
- Difficult if salting is used

Rainbow Table

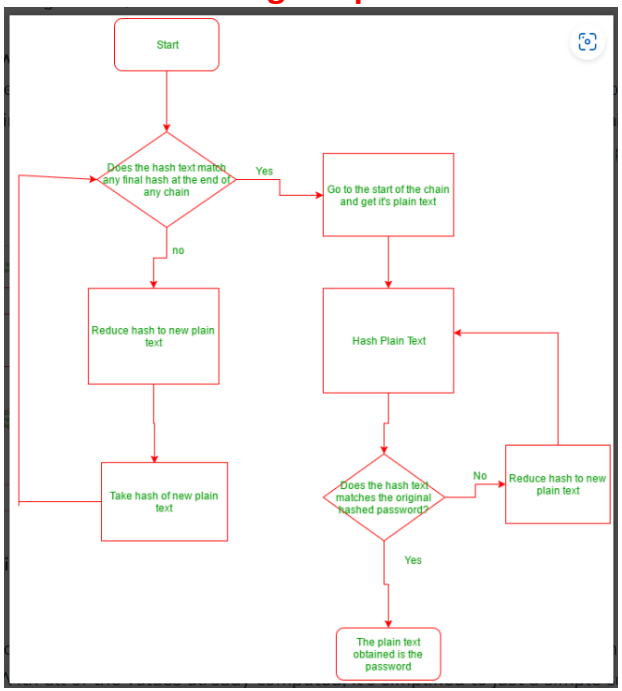
- Invented by Phillippe Oechslin in 2003
- M. Hellman had an initial idea in 1980
- Precomputed table for reversing hash functions
- Usually used in recovering a password up to a certain length consisting of a limited set of characters
- Uses reduction functions R_x that maps hash values back into values in a finite set of passwords P
- Reduction function is NOT actually an inverse of the hash function
- Table contains precomputed hash chains (e.g., length of chain n = 2)
  - Table stores only start and end points of chain

How Rainbow tables Work

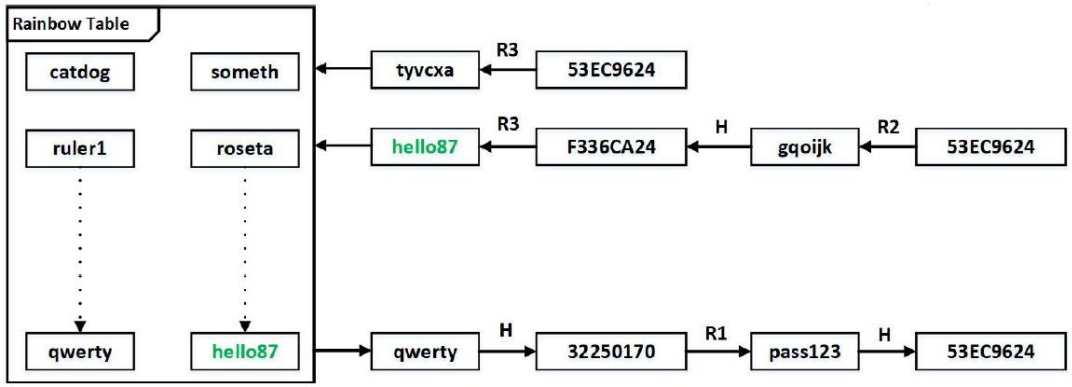
- Create a table of the most common password, **12345678**, using MD5 hash function on first 8 characters
  - hashMD5(12345678) = 25d55ad283aa400af464c76d713c07ad
- Reduce the hash by taking only the first 8 characters. Then, we re-hash it.
  - hashMD5(25d55ad2) = 5c41c6b3958e798662d8853ece970f70
- This is repeated until enough hashes in output chain.
- This represents one chain, which starts from the first plain text and ends at the last hash.
- After obtaining enough chains, we store them in a table.
- A rainbow table consists of many chains of alternating hashes and passwords.
- Rainbow tables use a different reduction function for each step of the chain.
  - This approach completely eliminates loops because a reduction function is never reused in the same chain.



Cracking the password



Rainbow Table



https://joshduck.com/blog/2008/02/09/rainbow-tables/

- Input hash is 53EC9624
- Hash is step by step reduced by chain 1,2…,n reduction functions, while the result is compared to all end points in the table (as only start and end is stored)
- If match found (hello87), then the end point is reduced by chain of reduction functions until plaintext is found (pass123)