# Hashing algorithms & password cracking

Jack Walton

November 4, 2019

Newcastle University

# Table of contents

# Hash functions

# Definition

A hash function $H$ maps from data of arbitrary size (the input) to data of fixed size (the hash)

Hash functions are designed to be "one-way" (easy to compute, hard to invert)

Toy hash function:

$$y = H(x) = \lfloor 5x \bmod 10 \rfloor,\ x \in \mathbb{R} \text{ and } y \in \{0, 1, \ldots, 9\}$$

# Toy function

Toy hash function:

$$y = H(x) = \lfloor 5x \bmod 10 \rfloor,\ x \in \mathbb{R} \text{ and } y \in \{0, 1, \ldots, 9\}$$

| $x$ | $H(x)$ |
|---|---|
| 3.14 | 5 |
| 2.72 | 3 |
| 1.41 | 7 |

**Table 1:** Input and output values

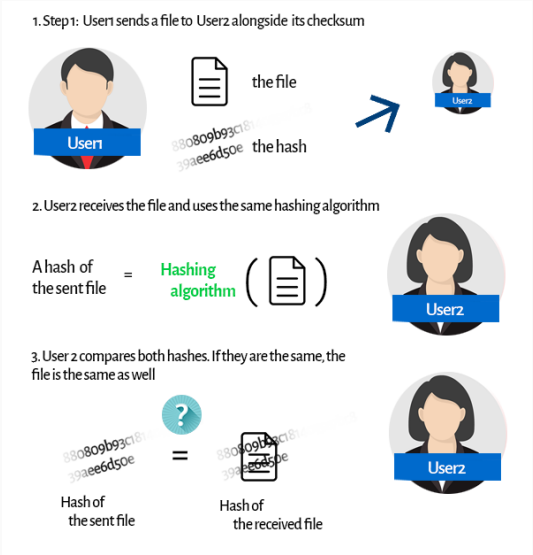
# Toy usage

- Alice & Bob are working on a homework problem
- They want to check they got the same result
- However, they do not want to reveal their answers to one-another
- Solution?

# Toy usage

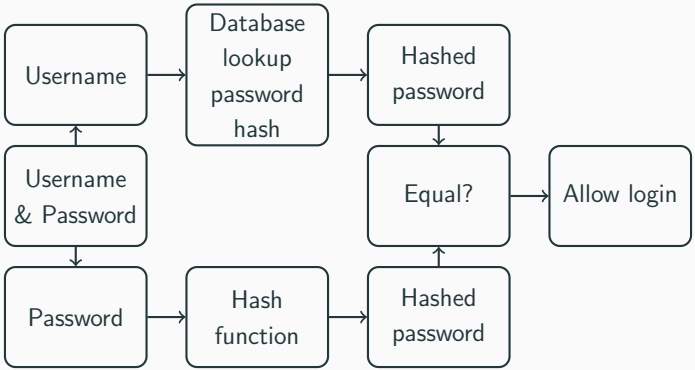
- Alice & Bob are working on a homework problem
- They want to check they got the same result
- However, they do not want to reveal their answers to one-another
- Solution? Hash and compare

# IRL usage: message authentication



# IRL usage: password verification

- Hashes are used to store passwords online
- Omits need for developers to store passwords in plaintext



# Desirable properties

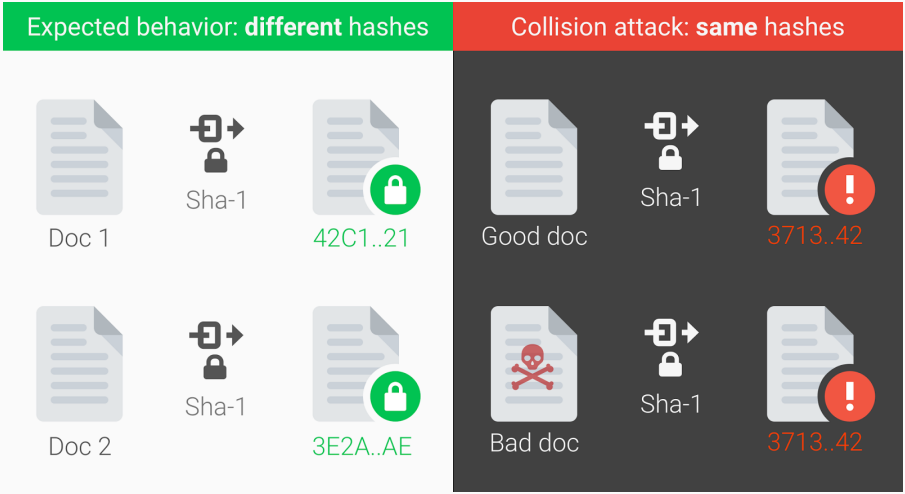
To use hash functions in the wild, we desire them to be:

1. Deterministic
2. Quick to compute given any input
3. One-way
4. Very sensitive to input
5. Infeasible to find collisions

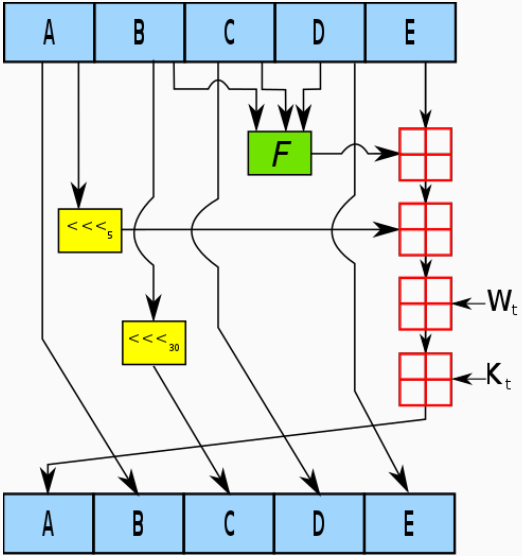
# Hashing in the wild: SHA-1

- Designed by the NSA and published in 1995
- Produces 160-bit hash (typically rendered as a hexadecimal number)
- Not considered secure against well-funded opponents (since 2005)
- In 2017 Google performed a collision attack on SHA-1

# IRL usage: message authentication



# Hashing in the wild: SHA-1



# Hashing in the wild: SHA-1

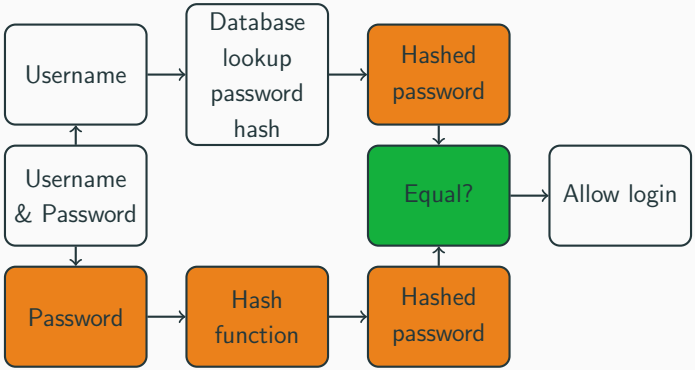
SHA1("The quick brown fox jumps over the lazy dog") output:
2fd4e1c67a2d28fced849ee1bb76e7391b93eb12

SHA1("The quick brown fox jumps over the lazy cog") output:
de9f2c7fd25e1b3afad3e85a0bd17d9b100db4b3

# Password cracking

# Hashcat

- Hashcat advertises as "World's fastest password cracker"
- Cracks passwords from leaked lists of hashed passwords
- Cracked passwords & emails used to attempt access to other services



# Demonstrations

- We will use the GPU equipped machine “Langkawi” (thanks NPP) to run hashcat
- Langkawi is equipped with a NVIDIA Tesla K40c graphics card, with 12gb onboard RAM
- Attempt to crack md5 hashed passwords released from LulzSec’s 2011 hack of EA’s Battlefield Heroes game

# Brute force demo

```
$ hashcat -m 0 -a 3 -O bfield.hash
```

# Brute force attack

| | Password length | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 6 | | | | 7 | | | | 8 | | | | 9 | | | |
| | y | d | h | m | y | d | h | m | y | d | h | m | y | d | h | m |
| 26 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 22 |
| 36 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 11 | 0 | 0 | 7 | 3 |
| 62 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 14 | 0 | 0 | 15 | 9 | 0 | 39 | 4 | 4 |
| 95 | 0 | 0 | 0 | 3 | 0 | 0 | 4 | 50 | 0 | 19 | 4 | 42 | 4 | 363 | 15 | 19 |

**Table 2:** Worst case scenario times to crack passwords hashed with md5 on Langkawi

# Dictionary attack

- Time to crack “P@55word” is 19 days. But surely this is a weak password?
- Instead of brute force we should try words we know people have used as passwords — so called dictionary attack
- Dictionary attacks make use of ‘word-lists’: lists of leaked passwords

# RockYou list

- 'RockYou' was a company which developed widgets for MySpace.
- Hackers used a 10-year-old SQL vulnerability to get RockYou user's passwords
- RockYou used an unencrypted database to store plaintext passwords (d'oh)
- List of these plaintext passwords is easily obtainable online. Known as 'RockYou list'

# Dictionary demo

```
$ ./hashcat -a 0 -m 0 -O bfield.hash rockyou.txt
```

# Rule based attack

- One of the most complicated attack modes
- Used to manipulate and transform passwords in word-lists (like the RockYou list)
- Rule-based attack like a programming language for password candidate generation

- Why not stick to regular expressions? Too slow.
- Typically have to generate 1 billion+ password candidates in less than 10 ms

# Rule based attack

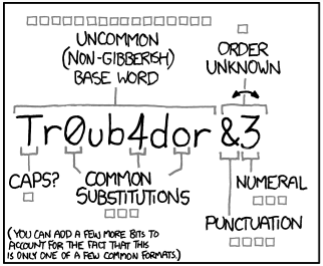
```
$ ./hashcat -a 0 -m 0 -O bfield.hash rockyou.txt -r rules/dive.rule
```

# Setting secure passwords

# Password security

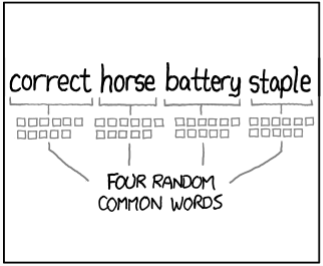
- All your passwords are bad and you should feel bad (probably)
- But how should we set secure ones?

# xkcd

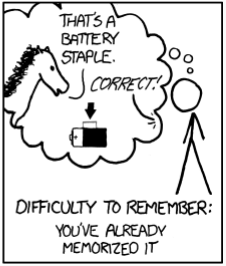

UNCOMMON (NON-GIBBERISH) BASE WORD
ORDER UNKNOWN
Tr0ub4dor &3
CAPS?
COMMON SUBSTITUTIONS
NUMERAL
PUNCTUATION
(YOU CAN ADD A FEW MORE BITS TO ACCOUNT FOR THE FACT THAT THIS IS ONLY ONE OF A FEW COMMON FORMATS.)


~28 BITS OF ENTROPY
2^28 = 3 DAYS AT 1000 GUESSES/SEC
(PLAUSIBLE ATTACK ON A WEAK REMOTE WEB SERVICE. YES, CRACKING A STOLEN HASH IS FASTER, BUT IT'S NOT WHAT THE AVERAGE USER SHOULD WORRY ABOUT.)
DIFFICULTY TO GUESS:
EASY


WAS IT TROMBONE? NO, TROUBADOR. AND ONE OF THE Os WAS A ZERO?
AND THERE WAS SOME SYMBOL...
DIFFICULTY TO REMEMBER:
HARD


correct horse battery staple
FOUR RANDOM COMMON WORDS


~44 BITS OF ENTROPY
2^44 = 550 YEARS AT 1000 GUESSES/SEC
DIFFICULTY TO GUESS:
HARD


THAT'S A BATTERY STAPLE.
CORRECT!
DIFFICULTY TO REMEMBER:
YOU'VE ALREADY MEMORIZED IT

# Passphrase generation

- Should move away from the concept of passwords to passphrases
- There are many passphrase generation techniques (DiceWare, PAO method, Schneier's Method, etc.)
- Recommend approach similar to xkcd. Additionally use uncommon words!
- Lists online of most common English words
- Don't use words or phrases that are meaningful to you

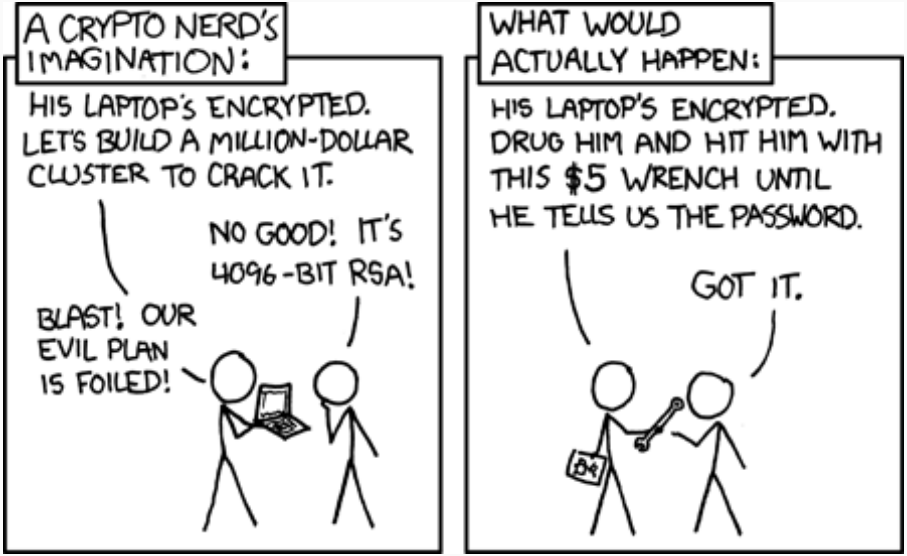
# Password managers

- Never reuse passwords
- Password managers provide an easy way to acheive this
- LastPass, 1password, KeePass, KeePassX
- With password managers the emphasis is on setting a secure master password

# Strong bois

- neon meat dream of an octafish
- murmuration cacophany
- phizzwizzwards quogwinkle

# Expectations vs. reality



# Take home points

1. Hash functions have uses in encryption and message authentication
2. Hashed passwords can be cracked using specialist software
3. Password managers help improve security


That's all Folks!
TM
Fritz Freleng
205
500
A WARNER BROS. CARTOON
WARNER BROS. INC. 1955