QUANTUM SECURITY

NoHolidayChurchGenius: Password Security with 2020 Vision

Antonio Radich

Experience

2017 - Present
Quantum Security
Security Consultant

- Compsci/Infosys Major
  University of Auckland

- OSCP

Antonio Radich

- Been at Quantum for 2 years
- Performing penetration tests on high value targets
- Enjoys developing video games

Quantum Security

- Wellington based security consultants
- Wide range of security services
- Clients in government, telecommunications and financial

Why talk about passwords?

- They stop people accessing your systems
- The technology and maths behind security are pretty solid
  - People are the weak point again
- Users will always choose convenience over security
  - Because people a lazy
- They are a failed concept
  - Need to remember far more now

Agenda

01 A 20/20 look at today

What's the current landscape and how did we get here?

02 Users are too smart

How are users responding to this environment?

03 Attacking smart users

Why everything they do is counter productive.

04 A 2020 look at the future

Enabling users.

A 20/20 look at today

What's the current landscape and how did we get here?

Trends of today

- What is the most common password of 2018*?
  - 123456
- What is the second most common password of 2018?
  - 123456789
- Where do we get all this data from?
  - Data breaches!
- 501,636,842 passwords, free to download!

*https://www.troyhunt.com/86-of-passwords-are-terrible-and-other-statistics/

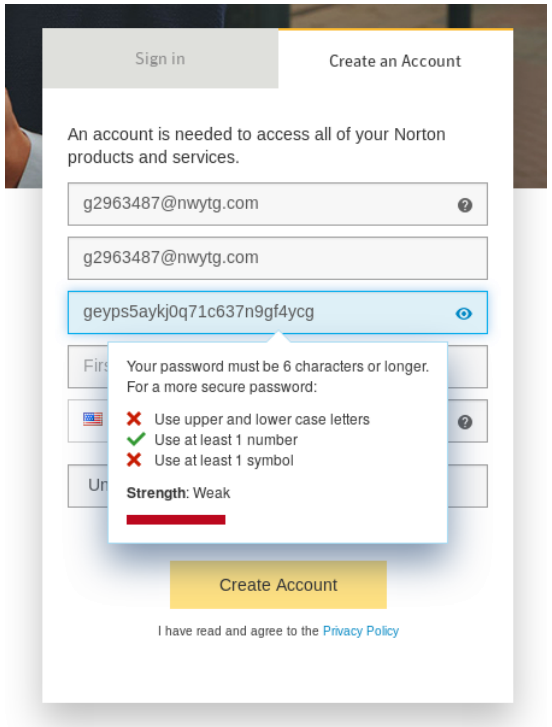
How did we get here

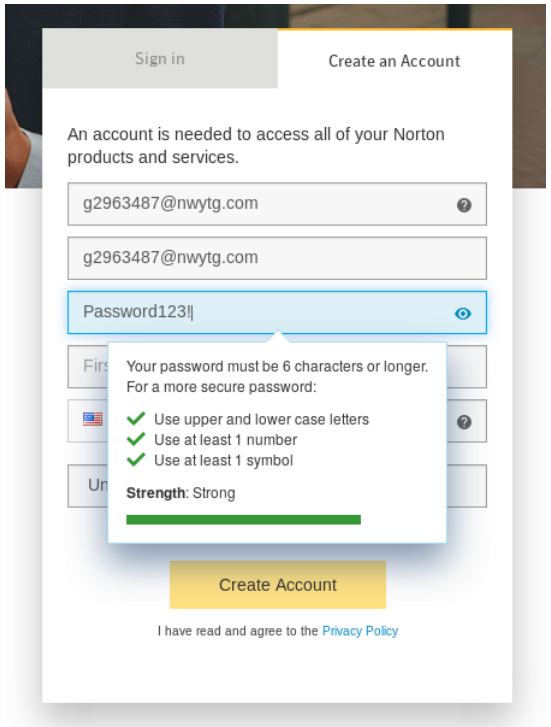
- The password policies out there are the issue
- Decided at NIST in 2003
  - Immediately adopted everywhere for some reason
- The decision:
  - Use a combination of alphanumeric characters with complexity
  - Some minimum length
  - Change it regularly
- NZISM is currently pretty much this

So what's the problem?

- People are gaming these systems for convenience

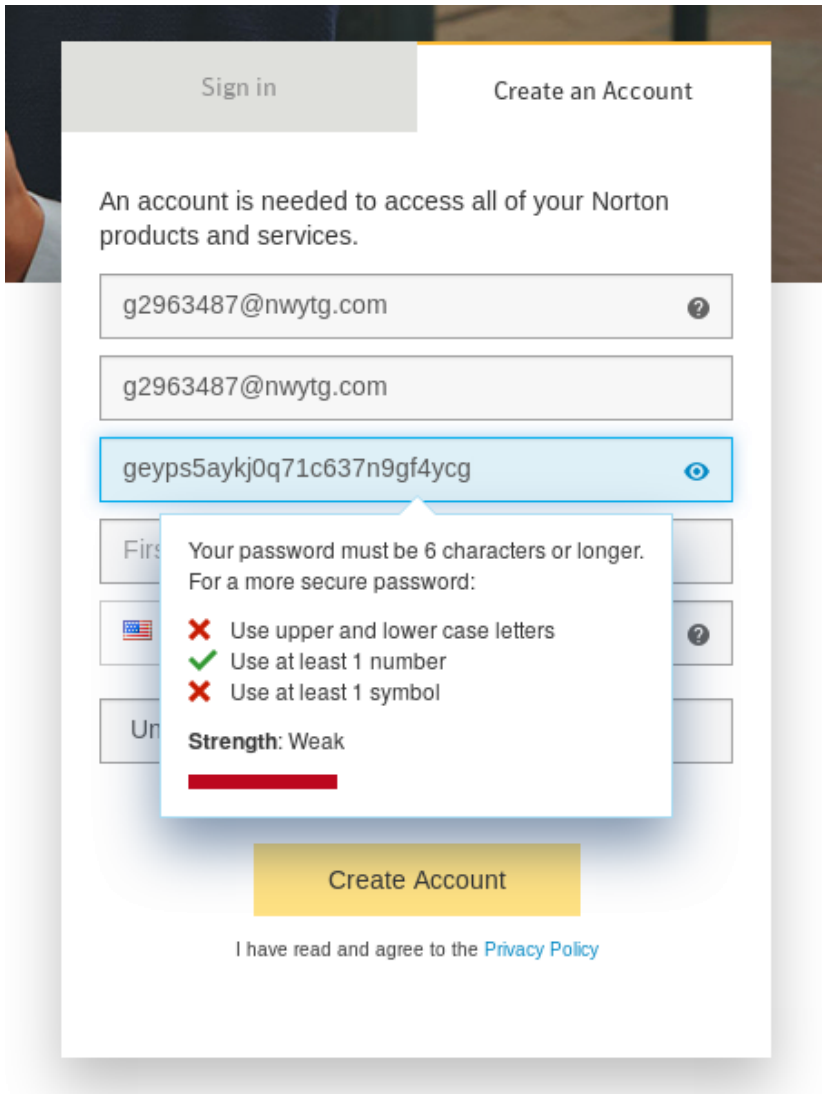
- Leading to 'strength checkers' making users make poor decisions

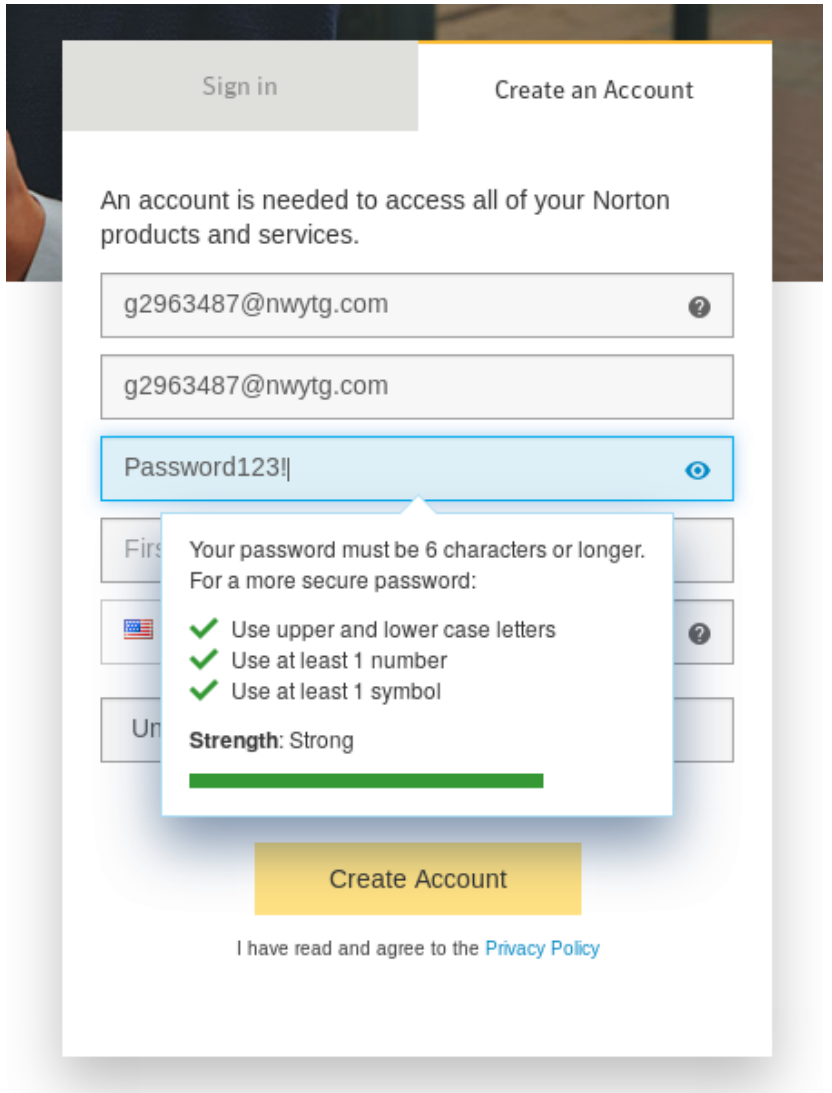




https://twitter.com/AaronToponce/status/988761412464857088/photo/1

So what's the problem?





https://twitter.com/AaronToponce/status/988761412464857088/photo/1

Why were those requirements chosen?

- Looked great! On paper

- Why were those requirements chosen?
  - More complexity = better?

Entropy

- Not in a physics sense, an information context
- Essentially a measure of randomness
  - *How many random choices were made to make this password?*
- Relevant when talking about cracking/brute forcing

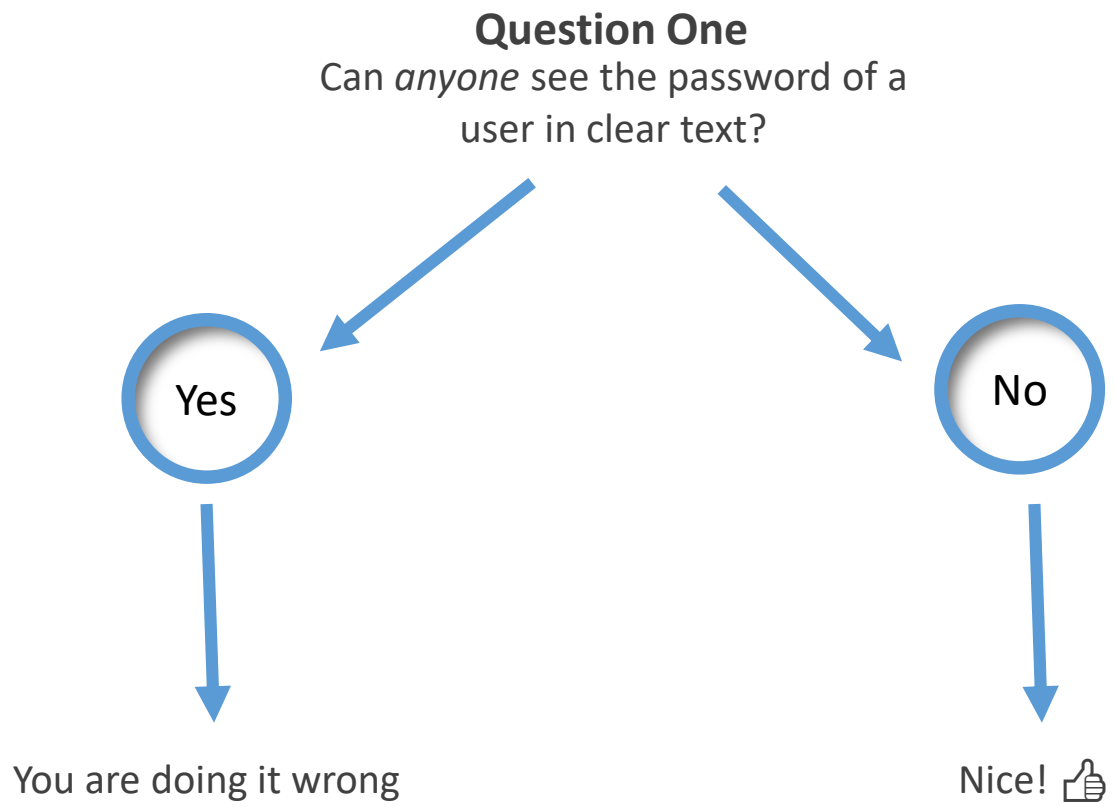
Why is Entropy relevant?

- High entropy = longer to crack
- You cannot calculate entropy of a given password
- It is a measure of randomness in the selection process
- Most data breaches will leak hashes, not passwords

What's a hash?

- A hash is a Digital Signature
  - Supercoolpassword123 -> `2567b46c3ea2a697ff3d9737cef39a36`
  - supercoolpassword123 -> `0fa24a38948fab9985d15b3292ba7bd8`
- Both had very similar inputs but very different outputs
  - Irreversible
- This is the correct way to store a password
  - With some additions

QUICK TEST: Am I storing passwords correctly?



Why is Entropy relevant?

- A simple way to calculate entropy in bits is:

$$\log_2(number\ of\ options^{length})$$

Why is Entropy relevant?

$$\log_2(number\ of\ options^{length})$$

- For example:
  - A random alphanumeric password of length 12
  - 52 alphabet characters of *both cases* and 10 numbers (62 options)

$$\log_2(62^{12}) = 71 \text{ bits}$$

- So what?

How did we get here

- Why were those requirements chosen?
- More bits = more complexity = better?

$$\log_2(62^{12}) = 71 \text{ bits}$$

62 possible characters
12 characters long

$$\log_2(52^{12}) = 68 \text{ bits}$$

52 possible characters
12 characters long

- More characters has made it more complex!
  - 8 times to be precise

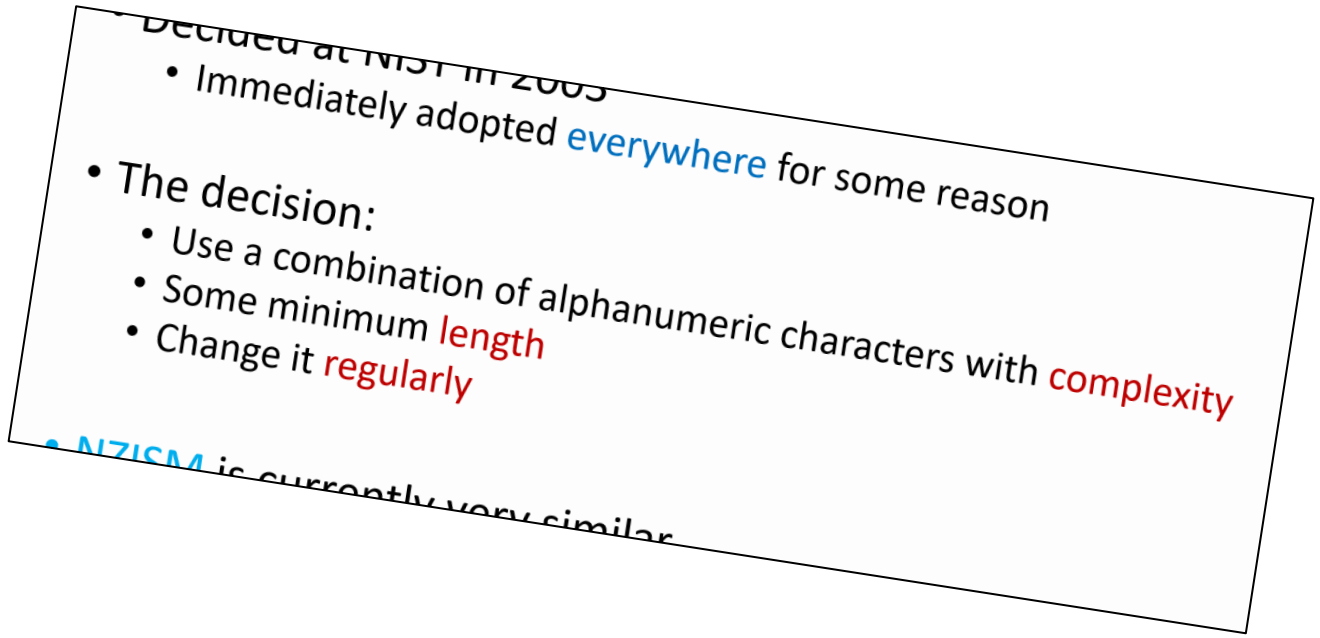
Users are too smart

How are users responding to this environment?

What are users doing?

- Users are gaming the system for convenience

- Lets take a look at those requirements again…



What are users doing?

- We see this time and time again in breaches
- Service desks using simple patterns
  - Winter2019
- Password reuse!
  - Using the same password as you did for LinkedIn in 2012?
  - Ohno…
- Is your administrator password the same as your standard user?

Reduced Randomness

- Example time!

- Lets say the password requirements are…
  - 11 minimum length
  - Any letter and numbers allowed

$$\log_2(number\ of\ options^{length})$$

Calculations

$$\log_2(number\ of\ options^{length})$$

$$\log_2(94^{11}) = 65 \text{ bits}$$

94 possible characters
11 characters long

Time to crack 65 bits

Hashes per Second	Time
10,000	116,988,483.5 years
1,000,000	1,169,884.8 years
1,000,000,000	1,169.9 years
100,000,000,000	11.7 years
1,000,000,000,000	1.2 years
100,000,000,000,000	4.3 days

Example Passwords:

- `y5)@Y$Jq5F_`
- `N&5(4^4kDqU`
- `fwJ6[u9BH'/`
- `Password12!`

Reduced Randomness

94 possible characters
11 characters long

Time to crack 65 bits

Hashes per Second	Time
10,000	116,988,483.5 years
1,000,000	1,169,884.8 years
1,000,000,000	1,169.9 years
100,000,000,000	11.7 years
1,000,000,000,000	1.2 years
100,000,000,000,000	4.3 days

Attacking smart users

Why everything they do is counter productive.

Series of events

1. Password policy dictate complex requirements
2. Users try to meet it predictably
3. Attackable patterns emerge

Reduced Randomness

- Everyone uses a capital to start
- Everyone uses a ! at the end
  - This reduces it to about 63 bits

$$\log_2(94^9) + \log_2(26^1) = 63 \text{ bits}$$

Time to crack 65 bits

Hashes per Second	Time
10,000	116,988,483.5 years
1,000,000	1,169,884.8 years
1,000,000,000	1,169.9 years
100,000,000,000	11.7 years
1,000,000,000,000	1.2 years
100,000,000,000,000	4.3 days

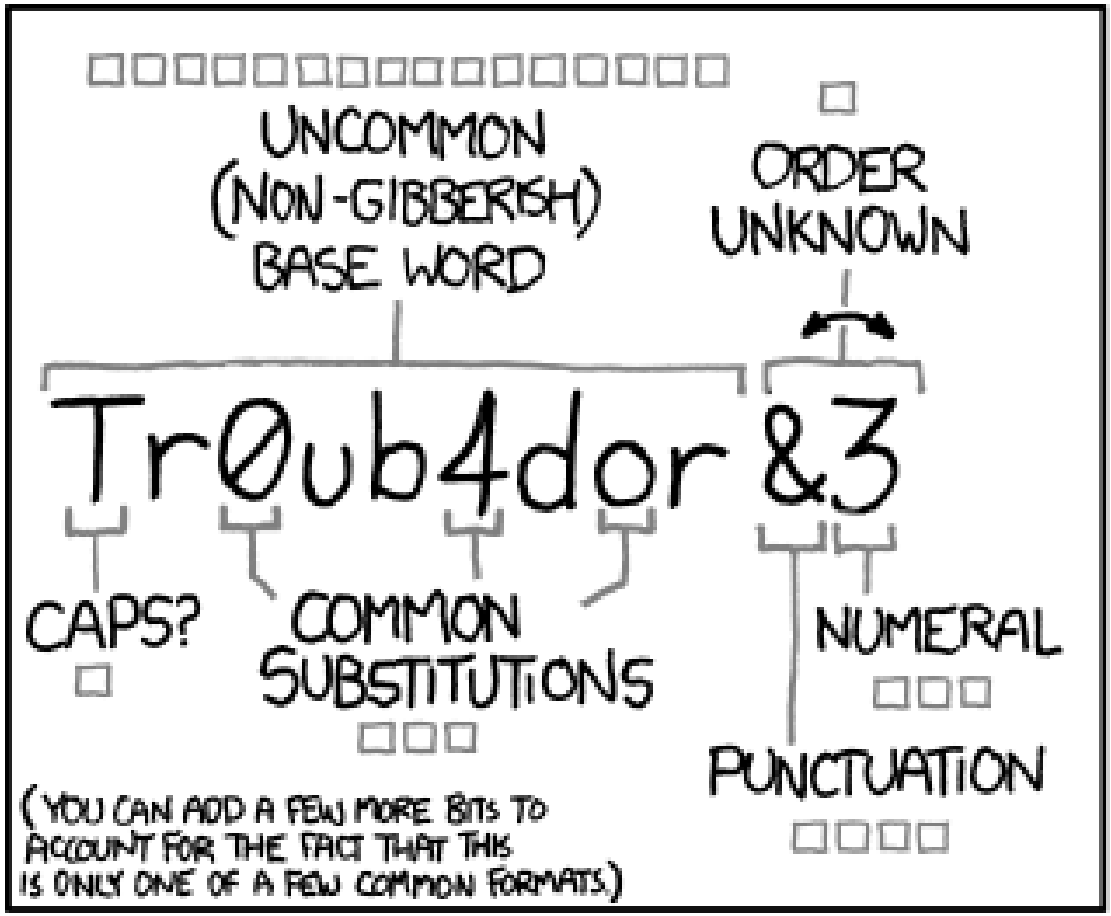
vs

Time to crack 63 bits

Hashes per Second	Time
10,000	29,247,120.9 years
1,000,000	292,471.2 years
1,000,000,000	292.5 years
100,000,000,000	2.9 years
1,000,000,000,000	3.5 months
100,000,000,000,000	1 day

Reduced Randomness

- Each of these assumptions by an attacker reduce entropy



Time to crack 65 bits (earlier example)

Hashes per Second	Time
10,000	116,988,483.5 years
1,000,000	1,169,884.8 years
1,000,000,000	1,169.9 years
100,000,000,000	11.7 years
1,000,000,000,000	1.2 years
100,000,000,000,000	4.3 days

Time to crack 28 bits (Tr0ub4dor&3)

Hashes per Second	Time
10,000	7.4 hours
1,000,000	4.4 minutes
1,000,000,000	0.26 seconds
100,000,000,000	0 seconds
1,000,000,000,000	0 seconds
100,000,000,000,000	0 seconds

Password Attacks

- Dictionary attacks are even more effective in a given time against real people

- This involves attempting a list of words or common passwords
  - These lists are publicly known

- Remember that 100 billion hashes per second?

Time to crack 65 bits

Hashes per Second	Time
10,000	116,988,483.5 years
1,000,000	1,169,884.8 years
1,000,000,000	1,169.9 years
100,000,000,000	11.7 years
1,000,000,000,000	1.2 years
100,000,000,000,000	4.3 days

vs

Time to try 1 billion hashes

Hashes per Second	Time
10,000	100,000 seconds
1,000,000	1,000 seconds
1,000,000,000	1 seconds
100,000,000,000	0.01 seconds
1,000,000,000,000	0.001 seconds
100,000,000,000,000	0.00001 seconds

Catered password attacks

- Dictionary attacks can also be catered to an organisaiton
- CeWL can create custom wordlists
- From individual site breaches we can see more patterns emerge
  - Apply these elsewhere?

1. cashcrate123
2. CashCrate
3. mycashcrate
4. cashcreate
5. cashcrate.com
6. etarchsac

*https://www.troyhunt.com/86-of-passwords-are-terrible-and-other-statistics/

A 2020 look at the future

Enabling users.

Length is best

- Length adds entropy much faster than complexity

$$\log_2(94^{10}) = 65 \text{ bits}$$

94 possible characters
10 characters long

Time to crack 65 bits

Hashes per Second	Time
10,000	116,988,483.5 years
1,000,000	1,169,884.8 years
1,000,000,000	1,169.9 years
100,000,000,000	11.7 years
1,000,000,000,000	1.2 years
100,000,000,000,000	4.3 days

$$\log_2(52^{12}) = 68 \text{ bits}$$

52 possible characters
12 characters long

Time to crack 68 bits

Hashes per Second	Time
10,000	935,907,868 years
1,000,000	9,359,079 years
1,000,000,000	9,359 years
100,000,000,000	94 years
1,000,000,000,000	9.4 years
100,000,000,000,000	1.1 months

Passphrases and Password Managers

- How do remember longer passwords?

- Two best options

Passphrases

Password Managers

- Does anyone remember this talks title?

Passphrases and Password Managers

- NoChurchHolidayGenius

$$\log_2(62{,}000^4) = 64 \text{ bits}$$

- NoChurchHolidayGenius vs &-2VBGcY!(

- Which is easier to remember?
  - Don't use NoChurchHolidayGenius!

Passphrases and Password Managers

- Password managers are a secure place to store many passwords
- Sounds risky?
- The password manager can generate monstrously complex passwords for you
  - It cant forget them.
  - Never brute forced. Never guessed.

$$\log_2(94^{30}) = 196 \text{ bits}$$

Time to crack 196 bits

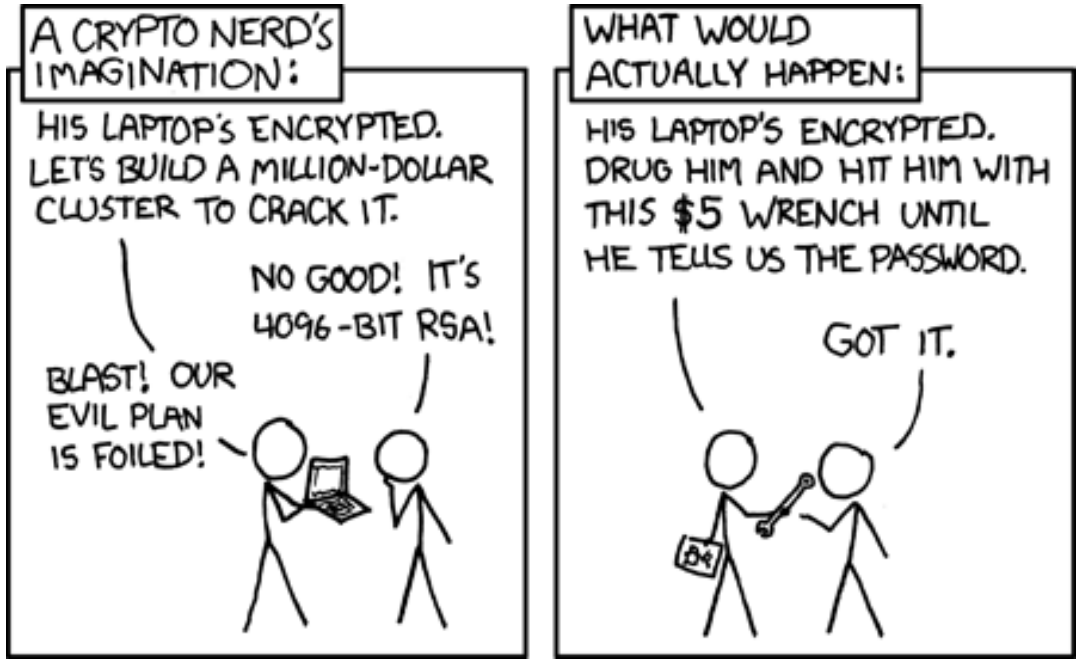
Hashes per Second	Time
10,000	∞ years
1,000,000	3.18×10^45 years
1,000,000,000	3.18×10^42 years
100,000,000,000	3.18×10^40 years
1,000,000,000,000	3.18×10^39 years
100,000,000,000,000	3.18×10^37 years

thirty-one duodecillion

Passphrases and Password Managers

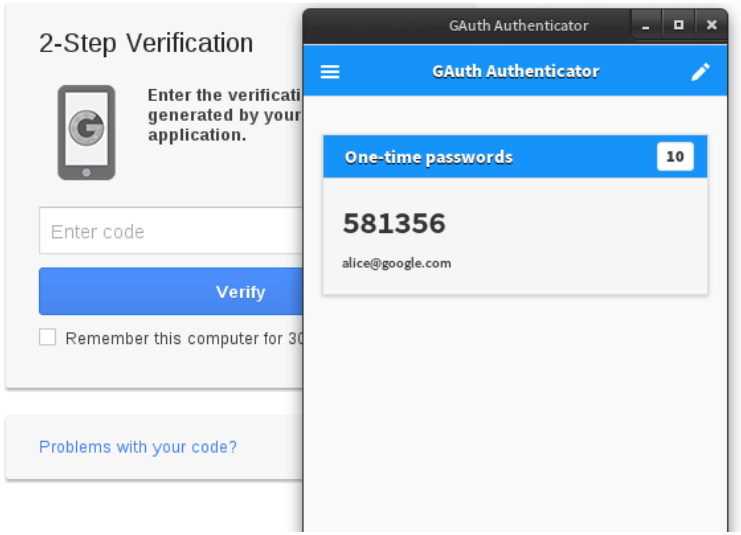
- Extra benefit! Prevent reuse!

- Also become resistant to Rubber-hose cryptanalysis



Multi-Factor

- Password leaked? No problem!
  - mostly…
- Soft or hard tokens are fine
- Defense in depth!
  - Don’t rely on 2FA during a breach



- Highly recommended for any administrative actions!

*Google image searches “2fa tokens”

Password Expiry

- What about expiry on passwords?
- If known to be compromised, do it!
  - Assume breach?
- Defense in depth again!
  - Cycle critical passwords
- A good password should remain a good password

Other quick advice

- Allow passwords as long as possible
- Allow pasting into fields
- Use better strength measurement
- No password history limit
- Brute force protection
- Store them securely!!

Conclusion

- Use longer passwords
- Use a password manager (protect it with a passphrase)
- Use multi-factor

- What new technology is emerging?
  - Biometrics

- Discussions are healthy

Any questions?

Antonio Radich

antonio@quantumsecurity.co.nz
https://www.quantumsecurity.co.nz/