The Password is Dead, Long Live the Password!

David C Frier, CISSP, CISM, CRISC, CCSK
Rochester Security Summit
October 6, 2015
What, him again?

* About me...
* David C Frier, CISSP, CISM, other stuff
* InfoSec Manager for Xerox’s Infrastructure
  * ...but I speak only for myself, not for Xerox!
* Been doing Information Security for 10 years
* IT of one sort or another for 37 years
* An avid player of poker and Ingress
* $FIRST.$LAST@{xerox|gmail}.com
What we Talk About when we Talk About Passwords

- What?
- Why?
- Usage
- Storage
- Complexity
- Volatility
- 2\textsuperscript{nd} Factor Considerations
- Future Possibilities
What are Passwords?

- A password is a string of characters or words
  - Known by a person...
  - Kept a secret...
  - That authenticates the person as being authorized to gain access to something.

- Passwords are ancient
  - “Halt, who goes there?”

- First Computer Password (probably)
  - CTSS by MIT – 1961
Why Passwords?

- **Cheap to implement**
  - Software-only
  - Probably built-in to whatever OS they were already using.
  - No training costs or learning curve for users (mostly)

- **Lowest Common Denominator**
  - If diverse systems with diverse schemes were being integrated...
  - What authentication mode did they all support?
Mainly: Authentication
Authentication is based on one or more factor from the list:
- something you know - that’s a password
- something you have
- who/what you are

Also: Encryption key
* It’s important for this to be disclosed and obvious
Password storage ranges from bad to not-so-bad

**User:**
- Memorized
- Paper & pencil
- Spreadsheet, etc.
- Encrypted Vault

**Server:**
- Plaintext
- Reversible Encryption
- Hashed
- Salted & Hashed
A quick digression

* Sometimes a system has a maximum password length

* Not referring to limitations of input forms, which may be as small as 255 characters...

* But if passwords are being hashed
  * Hash output is a fixed length, independent of input
  * Storage requirements are not a factor

* Storage limitations call the hashing into question
## Attacks and Defenses

### Attacks
- Rainbow Tables
- Cracking
- Dictionary
- Brute Force
  - *both flavors*

### Defenses
- Hashing & Salting
- Complexity
- Response Delays
- Lockouts
The Wrench Attack

A crypto nerd's imagination:
His laptop's encrypted.
Let's build a million-dollar cluster to crack it.

No good! It's 4096-bit RSA!
Blast! Our evil plan is foiled!

What would actually happen:
His laptop's encrypted.
Drug him and hit him with this $5 wrench until he tells us the password.

Got it.
Complexity correlates to **Shannon Entropy**

i.e. the expected amount of information contained in the message (string, words, phrase). For passwords, we usually count this in “bits”, which is **roughly** the $\log_2$ of the range of possible values. (The actual math behind this is **well** beyond the scope of this presentation, not to mention this presenter.)

**How to make passwords complex**

* Length
* Diversity of Character Set
* Nonexistence in Dictionaries
Why passwords need to be complex

- More resistant to attacks
  - Brute force -- Resist guessing
  - Dictionary -- Resist intelligent guessing
  - Rainbow tables -- Resist pre-computation of hashes

- Additional entropy for their usage as encryption keys
Complexity & Memorability Trade-Offs

Tr0ub4dor & 3

- Uncommon (non-gibberish) base word
- Order unknown
- 2^28 = 3 days at 1000 guesses/sec
- Difficulty to guess: Easy
- Difficulty to remember: Hard

Correct horse battery staple

- Four random common words
- 2^44 = 530 years at 1000 guesses/sec
- Difficulty to guess: Hard
- Difficulty to remember: You've already memorized it

Through 20 years of effort, we've successfully trained everyone to use passwords that are hard for humans to remember, but easy for computers to guess.
Complexity

Examples of making passwords complex

* pAs5w0rd – 36 bits  nope! Too short, also probably vulnerable to a well-engineered dictionary attack.

* CorrectHorseBatteryStaple -- 44 bits  not horrible.
  * If you want a tool for making these word-salad passwords, contact me for my “WordBase” spreadsheet, or try Diceware

* $jCYPGuFpM*QScf$9fwHn#82kqJWRW*3TT5j2Nek -- 215 bits  probably overkill.
Volatility

Changing Passwords:
* If a password is...
  * only ever used by one person
  * only ever transmitted encrypted
  * Salted-Hashed on the server, never stored as-is
* then it probably almost never needs changing

* On the other hand, consider the Cyber-Ark model for passwords that have to be passed around
Time for a 2\textsuperscript{nd} Factor

* Something you Know -- *covered*

* **Something you Have**
  * Examples:
    * Google Authenticator ([TOTP - RFC6238](https://tools.ietf.org/html/rfc6238))
    * Grid cards
    * RSA Tokens

* **Something you Are**
  * Biometrics are shiny but probably less practical
  * Can’t be changed
  * Error rates are still too high
Two-factor renders guessing and cracking attacks almost useless

Getting infrastructure in place to support a consistent usable 2FA is the challenge
  * Even Grid Cards require changes to all apps or to SSO
  * TOTP Assumes users all have smartphones
Some Future Possibilities

* **FIDO - Open Authentication Specification**
  * Public/Private Key Pairs per Site
  * User signs a challenge and website validates
  * Can be managed with apps as well as hardware

* **YubiKey**
  * Supports FIDO U2F Challenge/Response (Neo)
  * Proprietary OTP
  * Static passwords, also

* **Common challenges to all these**
  * The inertia of the “installed base”
  * Cost of issuing new tech to all users
Conclusions & Recommendations

- Emphasize Complexity over Volatility
- Salt & Hash Properly
- Overcome the 2FA challenge
- Buy my snake oil... *(j/k)*