

IT 4823

Information Security Concepts and Administration

Storing and Using Passwords



Notice: This session is
being recorded.



Copyright © 2016 by Bob Brown



About this Lecture

- Most of the material in this lecture came from a series of articles at the ArsTechnica.com web site.
- Other sources are credited on the appropriate slides.
- And some of it came right out of my head.

How Crackers Crack Passwords

- Guessing.
- Attacking the password file.
- Subverting the password reset mechanism.
- “Spear phishing.”

Guessing

- Common passwords
 - 4.7% of users have the password *password*
 - 14% have a password from the top 10 passwords
 - 40% have a password from the top 100 passwords
 - 79% have a password from the top 500 passwords
 - 91% have a password from the top 1000 passwords
- Knowledge of target
 - Birth date, phone number, license plate number
 - Spouse name, kids, dogs, work, etc.

<https://xato.net/passwords/more-top-worst-passwords/>

Can You Prevent Guessing?

- Disable account after n tries?
- Insert a long interval (seconds) between tries?
- Increase interval exponentially:
 - After first try: one second.
 - After second, two second
 - After third, four seconds, then eight seconds, etc.
- Filter and disallow those common passwords, maybe.

Attacking the Password File

- Bad guys obtain a copy of the password file, usually really a database.
- Often, this is through SQL injection.
- Depending on how the passwords were stored, they extract the passwords.
- Depending on how user IDs are formed, those passwords may give access to (many) other systems.

Forming User IDs

- Often organizations use email address as user ID because it's unique.
- If the victim of a password crack has used the same password for his email account, it's toast!
- If the victim of a password crack has used the same email address and password for any other service, it's toast.
- If an email account is compromised, it can often be used to subvert password reset mechanisms on other services.

Storing Passwords

- Usually the user ID is there, too, because it's the database key.
- Three ways to store passwords:
 - Plain text (evil!)
 - Hashed
 - Good hashes
 - Bad hashes
 - Hashed and salted.

Plain Text Storage

- An attacker who can obtain a copy of the password table has all the user IDs and passwords with no further work.
- Yes, people really do store passwords this way, amazing though it may be.

Hashed Password Files

- When an account is set up, the password is passed through a cryptographic hash function, and the result is stored.
- The password given with a login is run through the same function and compared with the database. If the hashes are the same, the login succeeds.
- Password: **love**
Hash: **b5c0b187fe309af0f4d35982**

Good and Bad Hash Functions

- Hash functions are used for two purposes:
 - Digital signatures, file fingerprints, etc.
 - Password storage
- For the first purpose, a fast algorithm is wanted.
- For passwords, attackers try to exploit the hash function itself, so a slow (computationally expensive) function is wanted.
- Good hash functions: bcrypt, PBKDF2
<https://crackstation.net/hashing-security.htm>

If the Password is Only Hashed

- The attacker takes a dictionary of several thousand words, runs them through the hash function.
- Assume Adam and Barbara both choose the password “love”
- They’ll **both** have the hash code:
b5c0b187fe309af0f4d35982
- ... and it’ll be in the attacker’s hashed dictionary! The passwords are toast.

The Value of Salting

- A “salt” is a random value added to a password *before* hashing.
- Adam: Password: *love* Salt: 12345
Hash: 4758ca3df2856cbf538f28c85d50
- Barbara: Password: *love* Salt: 67890
Hash: b7b6af2e6f9df077f15e0d3a6f65
- The hashes are different, and the attacker’s pre-computed dictionary is useless.
- **Salting stops pre-computation attacks.**

About Salts

- Salts must be *random*. Using the same salt for every password is equivalent to having no salt at all.
- Salts must be *long*. An eight bit salt has only 256 combinations. The attacker makes 256 hashed dictionaries, and uses the salt value to pick one. (The salt value must be stored in the password database.)
- With a long salt, passwords must be attacked one at a time, not the whole file at once.

Choosing a Salt

- Use a CSPRNG: cryptographically secure pseudo random number generator.
 - Passes the “next bit” test.
 - Withstands state compromise extensions.
- Most random number generators are not good enough.
- Rule of thumb: Use a salt size that’s the same size as the hash function output. So, for a hash function with 256 bit output, use a 256 bit hash.

Keyed Hashes

- If the hash function includes plain text password, salt, and a secret key, it will be harder to crack.
- The salt is stored in the password table; the secret key must not be.
- If the attacker gains control of “the system,” the secret key is toast...
- ... but most password table breaches are the result of SQL injection attacks.

Subverting the Password Reset

- If an attacker can gain control of a victim's email, it is often trivial to reset passwords for other accounts.
- Sometimes it is possible to subvert the password reset process using social engineering, *i.e.* deception.

http://www.slate.com/articles/technology/technology/2012/08/mat_honan_the_four_things_you_need_to_do_right_now_to_avoid_getting_hacked_.html

Spear Phishing

- “Spear phishing” is a phishing attack against a single individual or a small group.
- Like other phishing attacks, it starts with an email message.
- The message appears to come from someone like the victim’s boss...
- ... and contains an attachment like, “next_year_budget_cut.xlsx” which installs a keylogger.

<http://www.pcworld.com/article/2047628/spear-phishing-led-to-dns-attack-against-the-new-york-times-others.html>

Eavesdropping and Replay

- If an attacker can eavesdrop on a login transaction, the attacker can capture and later “replay” the password.
- Password transmission (from client to server) *must* be encrypted with TLS or other mechanism.

Anatomy of an Attack

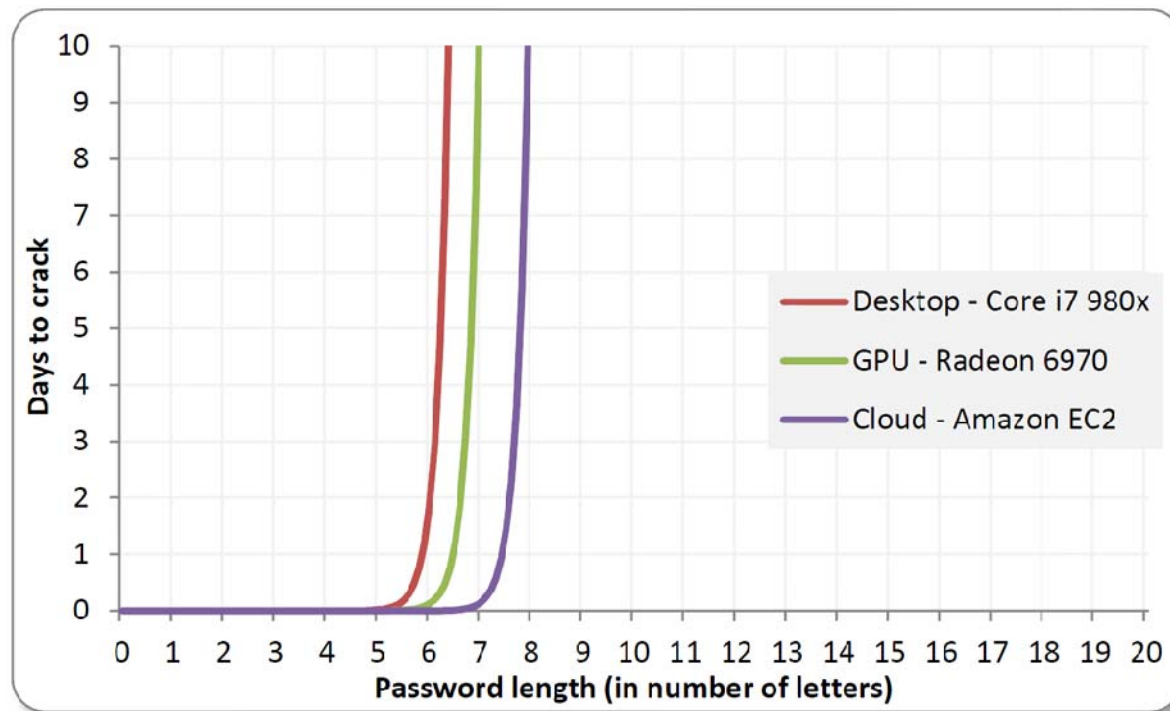
- Assume an adversary has acquired a copy of a password table including user ID, hashed password, and salt. (An attack is *much* easier if there's no salt.)
- A keyed hash was not used. (A keyed hash with a long key may be impossible to crack.)

Step 1: Brute Force

- There are only 95 possible one-character passwords with a standard keyboard. Try 'em all, using a GPU to compute hashes,
- There are only 95^2 or 9,025 two-character passwords. Try them, too.
- Use brute force up to six characters:
 $95^6 + 95^5 + 95^4 + 95^3 + 95^2 + 95 =$
742,912,017,120
- For a single password, or the whole file if no salt is used, this will take under three minutes.

Why Not Continue: the Exponential Wall

95^7 is about 6.8×10^{13} , a large number.



Step 2: Key-Limited Brute Force

- Try passwords of length 7 or 8 assuming a key space of lowercase letters only.
- Try again with only uppercase letters. About 41 seconds for each try.
- Try numbers only up to 12 digits: 3:21.

Step 3: Word Lists

- According to Mark Burnett, one could crack 98.8% of all passwords using a list of 10,000 words.
- Remove all “words” shorter than 7, letters only of length 7 or 8, and numbers up to 12. (These have already been tested, and need not be tried again.)

Step 4: Hybrid Attack

- Words (from list) plus single digit or special character. (Hybrid of word list and brute force.)
- Try adding two digits or specials, then 3.
- Add digits (only) of length 4. (*brown1946* falls to that one!)

Step 5: Analysis and Intuition

- By now, a large percentage of passwords have been cracked.
- Different users of the same site tend to use similar password patterns, for whatever reason.
- So, analysis of cracked passwords may reveal patterns, *e.g.* capital at the beginning, symbols at the end, all lowercase in the middle:

Qbesancon321

Are Passwords Dead?

- Not necessarily
- Four kinds of attacks
 - Online guessing
 - Compromised password files
 - Compromised password reset mechanisms
 - “Spear phishing”
- Each of these attacks is defensible.
- Password security can be bolstered:
 - Long pass phrases
 - Two-factor authentication

Helping Customers Form Good Passwords

- Do not reuse passwords. *Ever. At all.*
- The longer the better, and never less than eight. Remember the exponential wall.
- Use passphrases:
“My maternal uncle’s name is Bill.”
- Lie:
“My maternal uncle’s name is George.”
- Yoda-Speak:
“George my maternal uncle’s name is.”

Diceware

- Truly random selection.
- Common words (like “correct horse battery staple” but don’t use that one!)
- Length *may* prevent brute force cracking.
- Will not prevent testing permutations of the Diceware words. (So maybe make your own word lists?)
- Here are some dice with which to experiment.

Password Managers

- The only practical way to have unique passwords is a password manager.
- The user of a password manager has only to remember a master password, which should not be used for anything else!
- Some provide for cloud storage. (Pros and cons to that.)
- Brown likes Password Safe. (No cloud storage.)

Multi-Factor Authentication

- The three **factors** are:
 - Something you know
 - Something you have
 - Something you are
- Using two or all three factors improves security.
 - Loss of a token is less bad if a password is also required.
 - A guessed or compromised password is less bad if a fingerprint is also required. *Etc.*

Hardware Assists

- Can provide for two-factor authentication.
- The only practical (from a cost standpoint, for individuals) one I know about is the YubiKey, about \$50.
- There are password managers with YubiKey integrated.



Other Two-Factor Identification

There are two-factor mechanisms other than carrying a hardware “key”

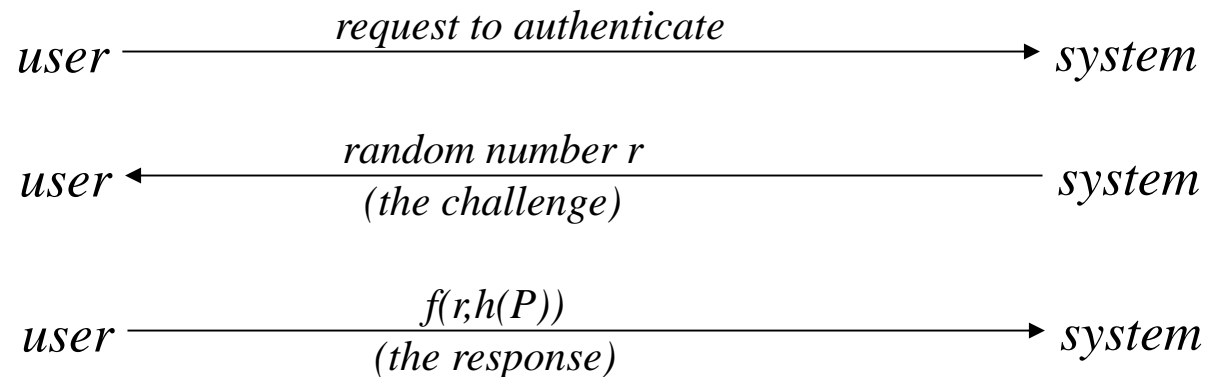
- Google’s SMS messages (also Microsoft, Facebook, and maybe Apple.)
- Twitter’s public key enabled app.

Remote User Authentication

- Authentication over a network is more complex
 - problems of eavesdropping, replay
- Mitigate these with a challenge-response scheme
 - user sends identity
 - host responds with random number
 - user computes $f(r, h(P))$ and sends back
 - host compares value from user with own computed value, if match user authenticated
- This protects against a number of attacks

Challenge-Response

User, system share a secret such as a cryptographic key, *e.g.* the password, P or its hash, $h(p)$. The function f is a cryptographic function. That the functions f and h are assumed known is Kerckhoffs' Principle.



Authentication Security Issues

- Client attacks
- Host attacks
- Eavesdropping
- Replay
- Trojan horse
- Denial-of-service (attacks availability; the rest attempt to attack confidentiality and integrity.)

Some Password Tips

- Change default passwords
- Help users deal with all their passwords
- Understand limitations of user generated passwords
- Understand limitations of machine generated passwords
- Prioritise Administrators and Remote user accounts
- Use account lockouts and protective monitoring
- Don't store passwords as plain text

https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/458857/Password_guidance_-_simplifying_your_approach.pdf

Anonymity on the 'Net

- Recipients can determine origin of incoming packet
Sometimes not desirable
- Anonymizer: a site that hides origins of connection
Usually a proxy server
 - User connects to anonymizer, tells it destination
 - Anonymizer makes connection, sends traffic in both directions
- Destination host sees only anonymizer

Anonymity Itself

Some purposes for anonymity:

- Removes personalities from debate
- With appropriate choice of pseudonym, shapes course of debate by implication
- Prevents retaliation

Are these benefits or drawbacks?

Depends on society, and who is involved

Privacy

- Anonymity protects privacy by obstructing amalgamation of individual records
- Important, because amalgamation poses three risks:
 - Incorrect conclusions from misinterpreted data
 - Harm from erroneous information
 - Not being let alone (abrogation of freedom from surveillance)
- Also hinders monitoring to deter or prevent crime
- Conclusion: anonymity can be used for good or ill
- Right to remain anonymous entails responsibility to use that right wisely

Pics of J-Law and Others ?!?

- Taken with an iPhone.
- Automagically backed up to iCloud, possibly without the user even being aware.
- User ID is email address.
- Until after the “hack,” Apple allowed unlimited login attempts to “Find my iPhone.”
- Start with that, and add a list of 1,000 common passwords. Security was toast!
- Apple denies a “breach.”

Thought for the Day

“There is no ‘cloud,’ it's simply a bunch of hard drives you don't own being operated by a bunch of people you don't know.”

– *NinjaNerd56 on Ars Technica.*

Questions

