

# CS574

# Computer Security

San Diego State University

Spring 2008

Lecture #11

# Today's Structure

- Administrivia
- Questions
- Lecture

# Administrivia

- Lecture #10 feedback
- Lecture #10 outline posted
- Assignment #1 update

**Questions?**

# Lecture

# Three Concepts

- Identification
- Authentication
- Authorization

# Why Authenticate?

- Accountability
- Access Control (authorization)
- Resource Control

# How To Authenticate

- Something you know
- Something you have
- Something about you

**Something You Know**

# Something You Know

- PIN
- SSN / Credit Card Numbers
- Passwords
- Challenge / Response Algorithms

# Passwords

- “Pass, friend.”
- Static / unchanging.
- Reusable / reused.
- Often poorly chosen.
- Sometimes shared or accidentally revealed.

# Password Storage

- Plaintext
- Trivially enciphered
- Encrypted
- Hashed
- Desirable features of password storage.

# Password Files

- The `/etc/passwd` scheme
- Attacks against `/etc/passwd`
- Fixes:
  - “salt”
  - “shadow” password file (least privilege)

# Password Threats

- Theft
  - steal from the system
  - obtain from the user
  - “sniff” from the network
- Guess
  - based on knowing the person
  - knowing how users tend to select passwords
  - random / exhaustive search

# Password Theft

- Password files (e.g., /etc/passwd)
- Configuration files
- Database tables
- Files in \$HOME
- Log files

# Password Theft

- Backpacks, purses, wallets.
- Sticky notes on monitor/keyboards/etc.
- Whiteboards

# Password Theft

- Unencrypted protocols like telnet, ftp, http
- Man-in-the-Middle attacks
- Social Engineering attacks
- Using “network sniffers” like tcpdump, wireshark
- A common intrusion activity

# Revealing Passwords

(from bash.org)

#244321 +(23119)- [X]

<Cthon98> hey, if you type in your pw, it will show as stars

<Cthon98> \*\*\*\*\* see!

<AzureDiamond> hunter2

<AzureDiamond> doesnt look like stars to me

<Cthon98> <AzureDiamond> \*\*\*\*\*

<Cthon98> thats what I see

<AzureDiamond> oh, really?

<Cthon98> Absolutely

<AzureDiamond> you can go hunter2 my hunter2-ing hunter2

<AzureDiamond> haha, does that look funny to you?

<Cthon98> lol, yes. See, when YOU type hunter2, it shows to us as \*\*\*\*\*

<AzureDiamond> thats neat, I didnt know IRC did that

<Cthon98> yep, no matter how many times you type hunter2, it will show to us as \*\*\*\*\*

<AzureDiamond> awesome!

<AzureDiamond> wait, how do you know my pw?

<Cthon98> er, I just copy pasted YOUR \*\*\*\*\*'s and it appears to YOU as hunter2 cause its your pw

<AzureDiamond> oh, ok.

# Guessing Passwords

- Repeated Login Attempts
  - slow, obvious, and logged
  - easy enough to control against
- Offline Attacks
  - steal (encrypted) password file
  - automated attack:
    - brute-force attack
    - dictionary attack
  - tools like ‘crack’ and ‘john the ripper’ simplify this process

# John the Ripper Demonstration

- Time or interest?
- Create some easy passwords.
  - also reverse, number/letter substitution
- Run John the Ripper against 'em.

# John/Crack Attacks

- Dictionary attack.
- Likely passwords.
- Common substitutions.
- Applies patterns (reverse, doubling, etc.).

# LANMAN Password Problem

- Older versions of Microsoft Windows OSes.
- In addition to the “normal” password, the system also stored each user’s password in the LANMAN format.
- LANMAN format:
  - Uppercase only password.
  - Pad out to 14 characters.
  - Hash 7 bytes separately, then concatenate.

# Password Selection

- Want to make guessing the password difficult.
- Avoid names, dictionary words, patterns.
- Longer is generally better.
- Should use upper and lower case, digits, and special characters.
- Use different passwords for different domains.

# Password Selection

- Techniques
  - Stem + variant approach.
  - Initials from favorite quote.
- Passphrases.

# Password Safety

- Frequent question: “How do I keep track of all my passwords safely?”
- Memorize and use frequently.
- Use a program to encrypt passwords.
- Write ‘em down and keep that paper safe.

# Challenge - Response

- What you know is an algorithm.
- Different Challenge → Different Response.
- Done right, it should be difficult to steal.
- May not scale well.

# Challenge - Response Sign/Countersign

- Traditional approach
- How it works
  - choose some set of words or phrases
  - first person challenges with first word
  - second person response with next word
- Can have several 'authenticators'

# Challenge - Response Finger/Number

- Used in at least WWII, Korea, and Vietnam
- How it works
  - choose a secret number in range 2 - 10
  - first person holds up a # of fingers
  - second person response with that plus the secret number
- Advantage: language independent and quiet

# Challenge - Response Secret Word

- How it works
  - choose a secret word or phrase
  - first person challenges with the first letter
  - second person responds with next letter
  - each person adds one more letter
- Goal is to spell the word without getting shot.

# Challenge - Response One Time Passwords

- Similar approach as a one-time-pad.
- AKA single-use passwords.
- Defends against replay attack.
- A variety of schemes and tools are out there (e.g., PAM OTP modules).
- See RFC 2289

**Something You Have**

# Something You Have

- Credit/Debit Card
- Keys
- Badge/Driver's License/SS Card
- Passport

# Something You Have

- Dumb Cards
- Smart Cards
- Authentication Tokens

# Dumb Cards

- Mag-stripe / swipe cards
- Identity + some sort of static “password”
- Saves you time (hopefully)
- Helps avoid shoulder-surfing attacks

# Smart Cards

- In addition to / replacement for dumb cards.
- Three categories:
  - optical memory
  - integrated circuit memory
  - integrated circuit processor
- Typical uses

# Tokens

- Handheld or key-fob “tokens”
- e.g., RSA SecurID, Secure Computing SafeWord
- Mechanism
  - Challenge-response calculator
  - Time-synchronized
  - Sequence-synchronized

# Token Challenge- Response Example

- User attempts to log in.
- Host issues a challenge.
- User enters PIN into token widget.
- User enters challenge string into token.
- Token calculates a response, displays a result.
- User types in the result as a “password”.

# Token Threats

- Theft (without additional authentication, the thief might find it trivial to impersonate you).
- May fail (hardware breaks).
- May be duplicated.
- May suffer from clock drift.

**Something About You**

# Something About You

- Where you are.
- What you do.
- Biometrics.

# Location Based Authentication

- Only allowed to log in via specific terminals or client stations.
- Sometimes, location alone is sufficient.
- Proximity (e.g., RFID) cards.
- Generally requires good physical security.

# Time Based Authentication

- Only allowed access a certain times.
- Ought to be used with another factor.
- Reasonable for corporate networks with limited-function terminals.

# What You Do

- Handwriting / signature
- Keystroke / typing patterns
- Voiceprints

# Biometrics

- Use something inherent about the user.
- Fingerprints most common approach.
- Retinal scans quite unique, but difficult.
- Iris scans not quite unique.
- *Interesting* privacy issues.
- Technology keeps getting better.

# Biometrics Risks

- Digitized data can be treated as a plaintext password that cannot be easily changed.
- Biometric changes are often unintentional.
- Link between device and system is a weak point.
- Often mandates constant supervision to ensure that the system is used properly.
- Sampling issues and false positive/negatives.
- Safety concerns.

# Authentication

# Multi-Factor Authentication

- Use more than one authentication mechanism.
- Two-factor examples:
  - swipe card + PIN
  - password + location
  - etc. etc.
- Helps avoid “stolen token” problems.

# Man in the Middle Authentication Attacks

- Console logins are easy to spoof.
- GUI logins aren't much harder.
- Login controls:
  - Get attention of OS
  - Verify Last Login

# Lecture References

- Pfleeger, *Security in Computing*, 3rd & 4th edition
- Bishop, *Introduction to Computer Security*

**End Lecture**

# Reading

- Chapter 4

**Finis**