Introduction

CS642: Computer Security

Professor Swift

http://www.cs.wisc.edu/~swift/

swift at cs dot wisc dot edu

Computer security:
understanding and improving the behavior of computing technologies in the presence of adversaries

Target/victim computing systems

Attackers

Security engineers

Computer systems:

- Operating systems
- Networks / Internet
- Web
- Software applications
- Cell Phones
- Internet-of-Things
- …

We will not even attempt to be exhaustive

Who am I?

- UW CS professor for 12 years
- Developer at Microsoft on Windows NT / Windows Cairo security team for 8 years
  - Authorization
  - Authentication
- Researcher on cloud security

My origins - Commodore CBM 2001 in 1981

1 MHz, 8-bit processor, 16 KB ram, audio cassette tape for storage (50 byte/second, 100 KB capacity)

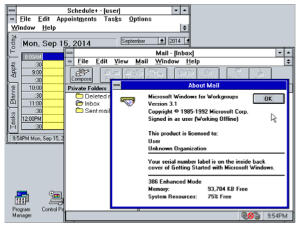
High school

- Computers the size of a refrigerator

16 bit, 3.6 MHz, 2 MB memory, 36 MB disk, 30 simultaneous users

College

- Intern at Microsoft



Schedule+ - [user]
File Edit Appointments Tasks Options Window Help
Mon, Sep 15, 2014
Mail - [Inbox]
File Edit View Mail Window Help

About Mail

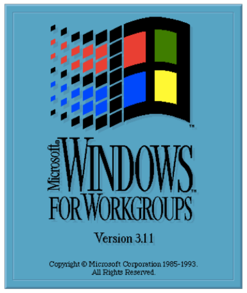
Microsoft Windows for Workgroups
Version 3.1
Copyright © 1985-1992 Microsoft Corp.
Signed in as user (Working Offline)

This product is licensed to:
User
Unknown Organization

Your serial number label is on the inside back cover of Getting Started with Microsoft Windows.

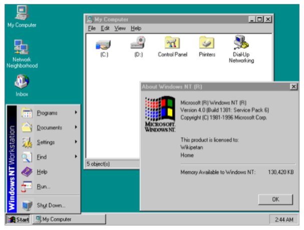
386 Enhanced Mode
Memory: 93,704 KB Free
System Resources: 75% Free

9:54PM Mon, Sep 15, 2
9:54PM



After college: Microsoft

- Windows, Windows Windows!



My Computer
File Edit View Help
(C:) (D:) Control Panel Printers Dial-Up Networking
5 object(s)

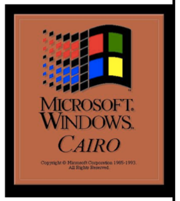
About Windows NT (R)

Microsoft (R) Windows NT (R)
Version 4.0 (Build 1381: Service Pack 6)
Copyright (C) 1981-1996 Microsoft Corp.

This product is licensed to:
Wikipetan
Home

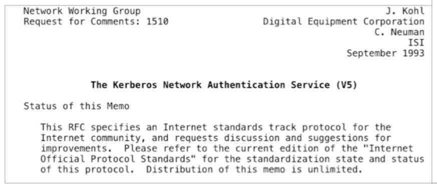
Memory Available to Windows NT: 130,420 KB

Windows NT Workstation
Programs
Documents
Settings
Find
Help
Run...
Shut Down...
Start
2:44 AM



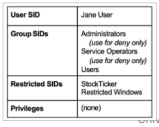
My jobs: security

- Implement Kerberos

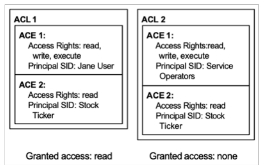


```
Network Working Group                                        J. Kohl
Request for Comments: 1510            Digital Equipment Corporation
                                                           C. Neuman
                                                                 ISI
                                                      September 1993


           The Kerberos Network Authentication Service (V5)

Status of this Memo

   This RFC specifies an Internet standards track protocol for the
   Internet community, and requests discussion and suggestions for
   improvements.  Please refer to the current edition of the "Internet
   Official Protocol Standards" for the standardization state and status
   of this protocol.  Distribution of this memo is unlimited.
```

- Implement access control



User SID	Jane User
Group SIDs	Administrators *(use for deny only)* Service Operators *(use for deny only)* Users
Restricted SIDs	StockTicker Restricted Windows
Privileges	(none)

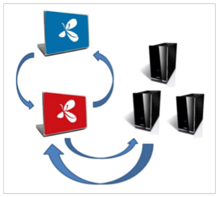


ACL 1
ACE 1: Access Rights: read, write, execute; Principal SID: Jane User
ACE 2: Access Rights: read; Principal SID: Stock Ticker
Granted access: read

ACL 2
ACE 1: Access Rights:read, write, execute; Principal SID: Service Operators
ACE 2: Access Rights: read; Principal SID: Stock Ticker
Granted access: none

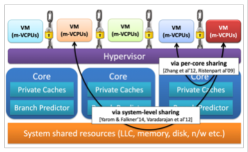
Weird things that happened

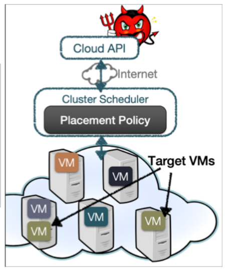
- When someone found a way to hack in to windows, we had to fix it
  - Jujitsu attack: sending network packets from a machine back to it could break passwords
  - L0pht crack: look at network packets and break passwords
- Often had to work weekends to fix problems right away
- Claims to fame:
  - NTLM zero-bit encryption
  - Gave LM hash key to Jeremy Allison/SAMBA

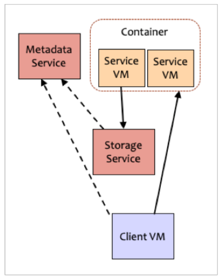


Recent Work

- Cloud side channels
- Attestation


VM (m-VCPUs)
VM (m-VCPUs)
VM (m-VCPUs)
VM (m-VCPUs)
VM (m-VCPUs)
Hypervisor
via per-core sharing [Zhang et al'12, Ristenpart al'09]
Core
Private Caches
Branch Predictor
Core
Private Caches
Core
Private Caches
Predictor
via system-level sharing [Yarom & Falkner'14, Varadarajan et al'12]
System shared resources (LLC, memory, disk, n/w etc.)





Security goals

- Confidentiality
  - data not leaked
  - encryption, access controls
- Integrity
  - data not modified
  - message integrity checks, access controls
- Authenticity
  - data comes from who we think it does
  - digital signatures, passwords
- Availability
  - services operating when needed
  - redundancy

Adversaries:

- script kiddies
- Criminals
- "hacktivists"
- Dissidents (if you are an oppressive regime)
- Nation states
- …

Attack mechanisms

- Rogue application
  - Malware in app store
- Over the network
  - Network packets
  - Emails
  - Websites
- Inside employee
  - Rogue: Snowden
  - Social engineering
- In the network
  - AT&T
- Over the air
  - Audio channels, Tempest
- Physical devices:
  - Stuxnet
- Physical access
  - Xerox copiers in Russia
  - NSA & Supercomputers

Anatomy of an example attack in 2011

http://arstechnica.com/tech-policy/news/2011/02/anonymous-speaks-the-inside-story-of-the-hbgary-hack.ars/1

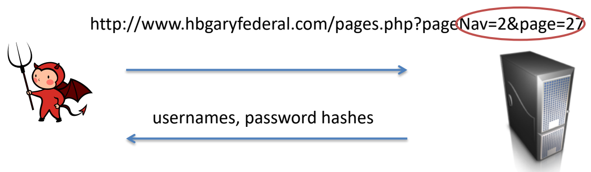
Anonymous vs HBGary

rootkit.com

hbgaryfederal.com

Ran by Greg Hoglund,
owner of HBGary / HBGary Federal

Anonymous vs HBGary


http://www.hbgaryfederal.com/pages.php?pageNav=2&page=27
usernames, password hashes

hbgaryfederal.com

Runs a CMS

SQL injection attack

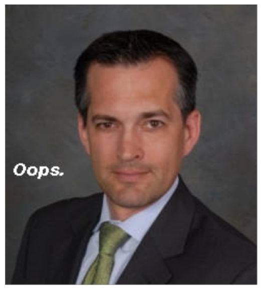
h = Hash(pw)

Given h, recover pw by brute force attack
if pw is "simple" enough

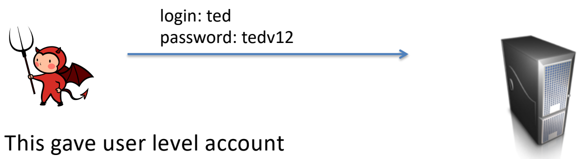
Aaron Barr's (CEO of HBGary) and Ted Vera (COO) had passwords only 6 digits, lower case letters and numbers

JohntheRipper easily inverts hashes of such passwords

http://www.openwall.com/john/



Anonymous vs HBGary



This gave user level account

hbgaryfederal.com

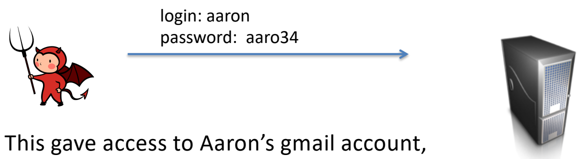
Exploit a privilege escalation vulnerability
in the glibc linker on Linux

Runs a CMS

http://seclists.org/fulldisclosure/2010/Oct/257

Now have root access on hbgaryfederal.com (and more?)
Delete gigabytes of data, grab emails, take down phone system

Anonymous vs HBGary



This gave access to Aaron's gmail account,
since he used same password here

google apps

Aaron was administrator for companies' email
on google apps

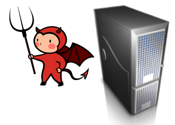
Runs a CMS

Read Greg Hoglund's emails

Anonymous vs HBGary

From: Greg
To: Jussi
Subject: need to ssh into rootkit
im in europe and need to ssh into the server. can you drop open up firewall and allow ssh through port 59022 or something vague?
and is our root password still 88j4bb3rw0cky88 or did we change to 88Scr3am3r88 ?
thanks

"social engineering"



rootkit.com

Recap:

- SQL injection
- Password cracking
- Privilege escalation via setuid program
- Social engineering

Themes in this course

- Understanding threats
- Security evaluations (thinking like an attacker)
- Defensive technologies
- Advancing our technical skills
  - x86 assembly, low-level programming
  - networking
  - cryptography
  - web security

Topic areas

- Low-level software security
- Processor security
- Network security
- Web
- Cryptography
- What else?

We will learn how systems break

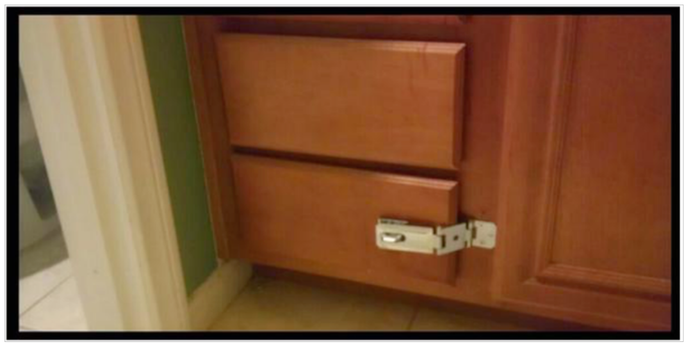
Security currently is an arms race between attack and defense

Security engineers must understand attack vectors in order to improve systems' security

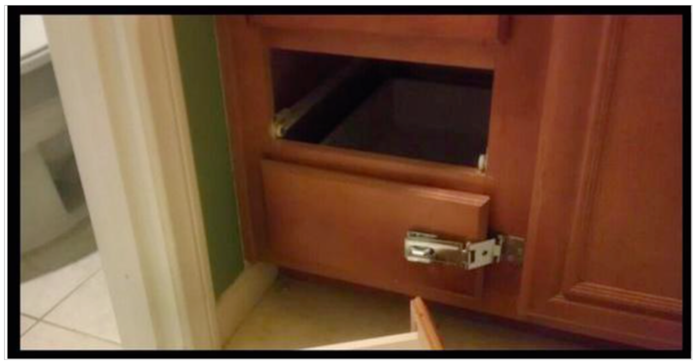
Security Mindset

- Thinking critically about designs, challenging assumptions
- Being curious, thinking like an attacker
- "That new product X sounds awesome, I can't wait to use it!" versus "That new product X sounds cool, but I wonder what would happen if someone did Y with it…"
- Why it's important
  - Technology changes, so learning to think like a security person is more important than learning specifics of today
  - Will help you design better systems/solutions

What do you see?



What do you see?



Security: not just for PCs

"The price of greatness is responsibility"

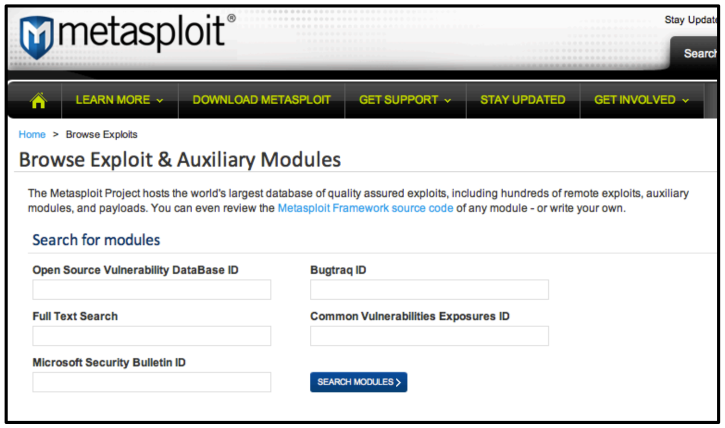
Winston Churchill

Black hat:
cracker, a criminal

Grey hat:
sometimes criminal, or at least "bending the law"

White hat:
ethical hacker, working within legal framework to perform security evaluations

Being a malicious script kiddie is easy … and stupid



Browse Exploit & Auxiliary Modules

The Metasploit Project hosts the world's largest database of quality assured exploits, including hundreds of remote exploits, auxiliary modules, and payloads. You can even review the Metasploit Framework source code of any module - or write your own.

Search for modules

Open Source Vulnerability DataBase ID

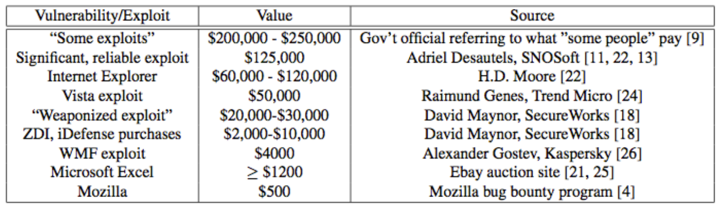
Bugtraq ID

Full Text Search

Common Vulnerabilities Exposures ID

Microsoft Security Bulletin ID

Reverse engineering and Zero days



Vulnerability/Exploit	Value	Source
"Some exploits"	\$200,000 - \$250,000	Gov't official referring to what "some people" pay [9]
Significant, reliable exploit	\$125,000	Adriel Desautels, SNOSoft [11, 22, 13]
Internet Explorer	\$60,000 - \$120,000	H.D. Moore [22]
Vista exploit	\$50,000	Raimund Genes, Trend Micro [24]
"Weaponized exploit"	\$20,000-\$30,000	David Maynor, SecureWorks [18]
ZDI, iDefense purchases	\$2,000-\$10,000	David Maynor, SecureWorks [18]
WMF exploit	\$4000	Alexander Gostev, Kaspersky [26]
Microsoft Excel	$\geq$ \$1200	Ebay auction site [21, 25]
Mozilla	\$500	Mozilla bug bounty program [4]

Table 1: Estimates on exploit values.

The Legitimate Vulnerability Market. Inside the Secretive World of 0-day Exploit Sales by Charlie Miller

The law and ethics

- Abuse of security vulnerabilities
  - is against University of Wisconsin policies.
    I will report anyone who "crosses the line" to the relevant university authorities
    http://www.cio.wisc.edu/policies.aspx
  - runs afoul of various laws.
- Abuse of security vulnerabilities is unethical
  - Think about what you're doing and the price it has on yourself, the victims, and society in general

Rules of thumb

- When in doubt … don't.
  - Come ask me
- You must have explicit (written) permission from a system owner before performing any penetration testing
  - Homework assignments will generally be on your own system
  - We will give explicit permission to hand us exploits for us to test

Responsible disclosure

- **Full disclosure** means revealing everything about a vulnerability including an example exploit
- **Responsible disclosure** (generally) refers to ensuring potential victims are aware of vulnerabilities before going public

Administrative stuff

- http://pages.cs.wisc.edu/~cs642-1
- Will use email list for announcements
- Piazza for discussion, bonus information
- Canvas for posting grades
- Homework assignments (50%)
- Midterm (20%)
- Final (20%)
- Participation

Homeworks

- Some problem sets will allow teams of up to 2
- Collaboration policy:
  - no collaboration with people outside team
  - using the web for general information is encouraged
  - Googling for answers to questions is not
  - Cheating will be reported to university authorities
- Need access to virtualization software: VirtualBox: https://www.virtualbox.org/

Details

- Exams: 2 Midterms
- Participation:
  - Ask questions in class
  - Come to office hours
  - Present an attack (up to 5 minutes)
    - Stuxnet
    - Melissa
    - Equifax
    - Snowden leak tools

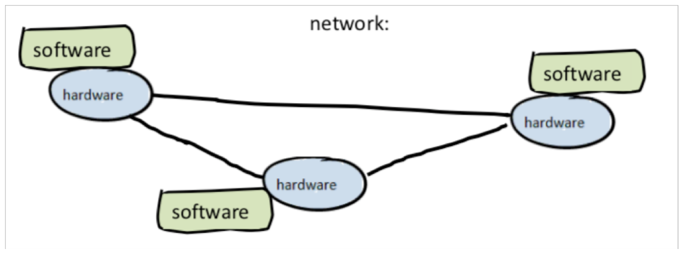
Participation

- Speak up in class
- No need to read all papers for a lecture in detail, but:
  - Be aware of topic areas
  - Read in depth selectively later

A warm up: security principles

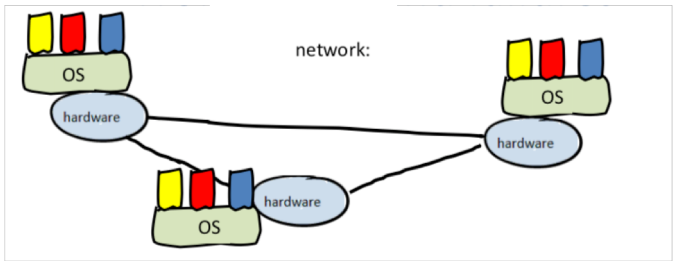
Saltzer and Schroeder.
The protection of information in computer systems.
Proceedings of the IEEE, 1975

Errors, bugs, failures



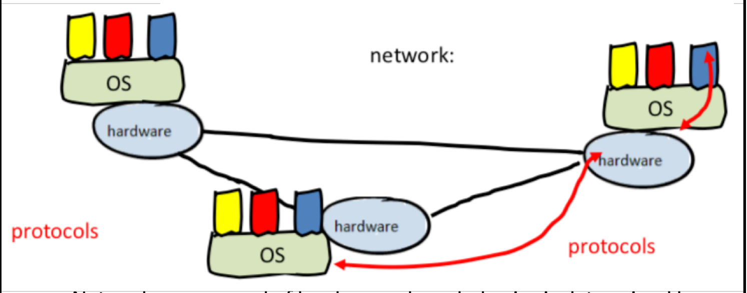
Networks: composed of hardware whose behavior is determined by software (roughly...)

Errors, bugs, failures



- Applications run on operating systems

Errors, bugs, failures



- Networks: composed of hardware whose behavior is determined by software (roughly...)
- Applications run on operating systems
- interoperate through protocols

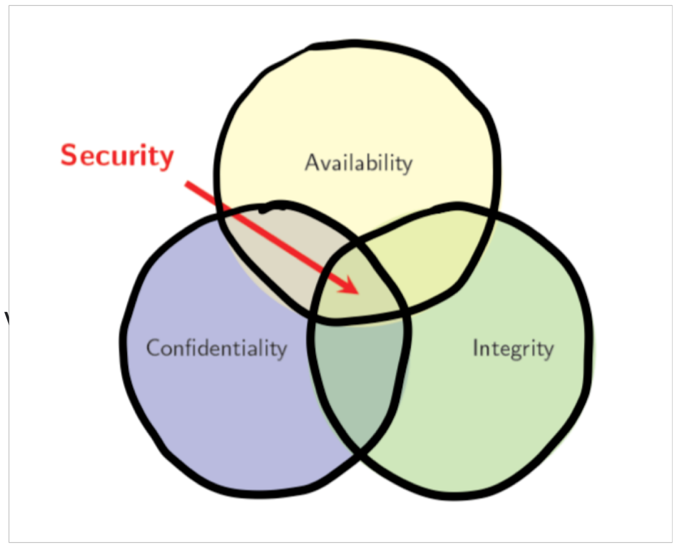
Hardware, Software, Protocols

- Designed by humans
- Not perfect!

A human error may introduce a bug (or fault)
The IEEE Standard Glossary of Software Engineering Terminology defines "fault" as "an incorrect step, process, or data definition in a computer program"
When a fault gets triggered, it might generate a failure...

Security Bugs, Errors, Failures

- A security error is made by a human
- Aa consequence, a security bug is introduced
  - A security bug is also called a “vulnerability”
  - When the bug is triggered (or “exploited”) it generates a security failure
  - The security of a system is compromised...



The root of it All

- Trust: when should you trust things?
  - In the real world:
    - When you know someone personally
    - When someone you know vouches for it
    - When it looks official or is in an official place
  - On the Internet:
    - When it comes from a believable web site?
    - When it is digitally signed?
- Case study: Unix

Trusting Trust

Example: Thompson’s Turing Award

- How to build an undetectable Trojan Horse that lives forever!
  1. Modify login to accept login “Ken” without a password and to grant root permission. This code would be obvious, if left in the source code, so, replace it with a trigger (some identifiable, but innocuous comment).

In login

```
if (strcmp(user, “Ken”))  != 0) {
   … check password here …
}
```

```
/* Check for valid password. */
 … check password here …
```

  2. Modify the compiler so that when it compiles login and sees the trigger, it adds the Ken-checking code. Since this is also obvious; replace that.

In compiler

```
if (strcmp(comment,
     “Check for valid password.”)
    == 0) {
        … Add Ken-check code ...
```

```
/*
 * Is this a comment?
 * If so, ignore.
 */
```

Trusting Trust (2)

4. Now, modify the compiler AGAIN to recognize the second trigger ("Is tl a comment?"). Add the code that recognizes the comment, but replace with the code that checks for the comment in login.
5. Compile this (second) hacked version of the compiler.
6. Now, remove the source code from the compiler.
7. You are left with an executable that will always generate a buggy comp as long as a particular comment in the compiler source code doesn't g away.
8. What is even more remarkable is that this bug can persist, even if you the compiler to a new backend! (It's all in the front end parsing.)

How do you know this sort of bug isn't in your compiler today!?

Security Design Principles

Security Design Principles

- Saltzer &Schroeder, 1975, as part of Multics

    1) Economy of mechanism
    2) Fail-safe defaults
    3) Complete mediation
    4) Open design
    5) Separation of privilege
    6) Least privilege
    7) Least common mechanism
    8) Psychological acceptability

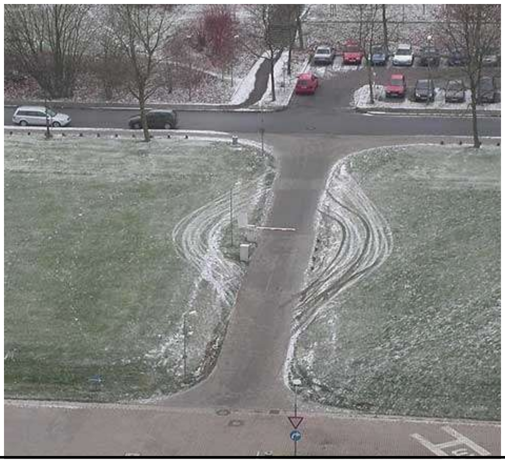
Economy of mechanism

Fail-safe defaults

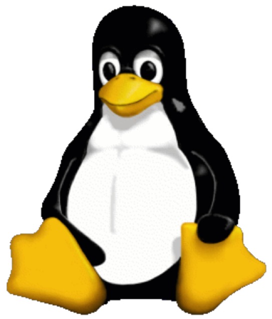
```
isAdmin = true;
try {
    codeWhichMayFail();
    isAdmin = isUserInRole( “Administrator” );
}
catch (Exception ex) {
    log.write( ex.toString() );
}
```

(Example from https://www.owasp.org/index.php/Secure_Coding_Principles)

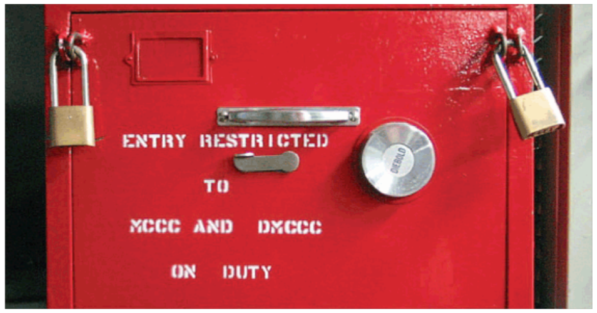
Complete mediation



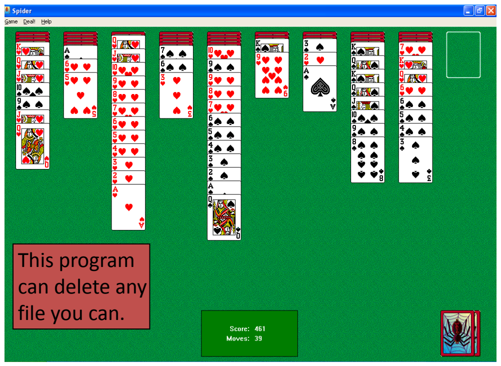
Open design
(avoid “security by obscurity”)



Separation of privilege



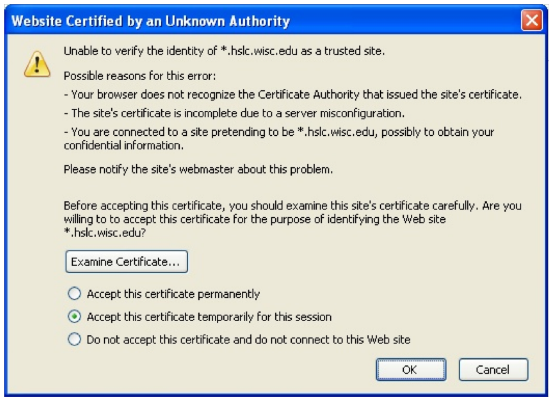
Least privilege



(Courtesy of UCB CS161 slides)

Least common mechanism
(isolation)

Psychological acceptability
(consider human factors)



Website Certified by an Unknown Authority

Unable to verify the identity of *.hslc.wisc.edu as a trusted site.

Possible reasons for this error:
- Your browser does not recognize the Certificate Authority that issued the site's certificate.
- The site's certificate is incomplete due to a server misconfiguration.
- You are connected to a site pretending to be *.hslc.wisc.edu, possibly to obtain your confidential information.

Please notify the site's webmaster about this problem.

Before accepting this certificate, you should examine this site's certificate carefully. Are you willing to to accept this certificate for the purpose of identifying the Web site *.hslc.wisc.edu?

Examine Certificate...

- Accept this certificate permanently
- [✓ Accept this certificate temporarily for this session]
- Do not accept this certificate and do not connect to this Web site

OK Cancel

Principles from 1970's

- Do you think they are relevant today?
- A bit… abstract
- Recur over and over again