

CS530

Introduction to Security Systems

Bill Cheng

<http://merlot.usc.edu/cs530-s10>

Copyright © William C. Cheng



1

What Is Security

- What are you trying to secure?
 - ▬ system
 - ▬ network
 - ▬ data
- How to evaluate
 - ▬ can be difficult
 - ▬ what are the costs?
 - hardware & software
 - administration/management
 - ▬ balance costs to protect with costs of compromise
 - ▬ balance costs to compromise with benefit to attacker
- Security vs. Risk Management
 - ▬ (cont...)

Copyright © William C. Cheng



2

What Is Security (Cont...)

- Security vs. Risk Management (cont...)
 - ▬ prevent successful attacks vs. mitigate the consequences
 - ▬ an example of Risk Management: banks
 - difficult to defend against losses from robbery, credit card fraud, identify theft
 - solution: charge fees, understand costs, buy insurance
- It's not all technical

Copyright © William C. Cheng



3

What Do We Want From Security

- Protection
 - ▬ enforced by hardware
 - virtual memory system
 - user/kernel modes, rings 0-3, etc.
 - no stepping around, no I/O accesses
 - ▬ depends on trusted kernel
- Authentication
 - ▬ determining identity of principal
 - a principal can be a process or a user
 - can use an access matrix to specify what subjects can access what objects
- Integrity
 - ▬ (cont...)

Copyright © William C. Cheng



4

What Do We Want From Security (Cont...)

- Integrity
 - ▬ authenticity of document
 - ▬ that it hasn't changes
- Confidentiality
 - ▬ that inappropriate information is not disclosed
- Availability
 - ▬ that the system continues to operate
 - ▬ that the system and data is reachable and readable
- Enforcement of policies
 - ▬ privacy
 - ▬ accountability and audit
 - ▬ payment

Copyright © William C. Cheng



5

What Makes Up Security

- Basic services:
 - ▬ Authentication
 - ▬ Authorization
 - ▬ Accounting (e.g., quota)
 - ▬ Audit
 - ▬ Assurance (e.g., software engineering, virus checkers)
 - ▬ Payment
 - ▬ Protection
 - ▬ Policy
 - rules about who can do what, at what cost
 - generally hard to define for an organization
 - ▬ Privacy (policy about individual)
 - ▬ Confidentiality (about data)

Copyright © William C. Cheng



6

Security Weaknesses & Why We Are Not Secure

- ↳ Buggy code
 - buffer overrun
 - never use `strcpy()`, use `strncpy()` and `memcpy()`
 - always check return code of library functions and system calls
- ↳ Protocols design failures
 - unspecified patterns
 - holes in the spec?
- ↳ Weak crypto
 - it is usually a good idea to use well understood ones
- ↳ "Social engineering"
 - (cont...)

Copyright © William C. Cheng



Security Weaknesses (Cont...)

- ↳ "Social engineering"
 - failure in people?
 - plenty of bad people out there (and inside)
- ↳ Misconfiguration
 - systems should be shipped in **secure mode** (not *open mode*)
 - unfortunately, this is usually against what vendors want
- ↳ Incorrect policy specification
- ↳ Stolen keys or identities
 - weak key management
 - single sign-on feature (put password on local disk)
- ↳ Denial of service
 - hard to defend against

Copyright © William C. Cheng



Security Mechanisms

- ↳ Encryption
 - scrambling of data for confidentiality and integrity
- ↳ Checksums
- ↳ Key management
 - e.g., Kerberos, X.509
- ↳ Authentication
 - e.g., Kerberos, X.509
- ↳ Authorization
 - ACL (access control list)
- ↳ Accounting
- ↳ Firewalls

Copyright © William C. Cheng



Security Mechanisms (Cont...)

- ↳ VPNs
 - interconnecting private nets over the Internet
- ↳ Intrusion detection and response
 - audit
 - push back authorization & firewall
- ↳ Development tools
- ↳ Virus scanners
- ↳ Policy managers
- ↳ Trusted hardware

Copyright © William C. Cheng



Today's Security Deployment

- ↳ Most of the deployment of security services today handles the easy stuff, implementing security at a single point in the network, or at a single layer in the protocol stack:
 - firewalls, VPN's
 - IP Sec
 - SSL
- ↳ Unfortunately, security isn't that easy. It **must** be better integrated with the application
 - at the level at which it must ultimately be specified, security policies pertain to application level objects, and identify application level entities (users)

Copyright © William C. Cheng

