

Lecture Notes – Computer Security

Author: Dr. Aaron Striegel
University of Notre Dame striegel@nd.edu

Topic: OS Issues / Trust

Controlling Threats

- Software Development
 - Modularity
 - Smaller, easier to analyze – think SW Eng
 - Cohesion
 - Logical purpose vs. mumbo jumbo
 - Coupling
 - Dependence on one another
- Software Engineering

Fault Tolerance & Security

- Making a system fault tolerant
 - Passive (re-active) vs. pro-active
 - Redundancy C++, Java
 - How redundant?
 - Byzantine general problem – bad code is smart/malicious

Proving Correctness

- Algorithms/etc.
- Translate to logical implications -> discuss next week

Operating System -> Use of Programs

- Trusted Software
 - MS Palladium
 - Test / signed in trusted environment
- Mutual Suspicion
- Confinement
 - Digital sandbox for untrusted code
- Access / Auditing
 - Monitoring / detection of problems

Operating Systems

- Separation
 - Keep user's objects separate from one another
 - Solaris box example
 - Login to e-mail, AFS, etc.
 - Definitions
 - Physical DoD

- Temporal
- Logical
- Cryptographic
- Level of separation
 - No protection Need physical/temporal
 - Isolate Unware of others
 - Share all/share none
 - Share via access
 - Share via capabilities
 - Limit use of object
- Key: What granularity of control?

Memory Address

Simple embedded system

- Two programs running

- Sensor – watching the data

- Wireless x-mitter – send/receive data requests/etc.

?: What prevents the x-mitter program from overwriting sensor data?

- Think architecture / etc.

- Asm example

- Draw memory map

Fence

- Pre-defined memory address

Fence Register

- Allow variable locations

Relocation

- Goes through OS before going to memory

- HW relocation -> page table from architecture

Base / Bounds register

- Variable size program space

- Context switch

- Multiple stacks / etc.

Protect some but not all

- Everyone can print

- Not everyone can see User 3156's data

Tagged architecture

- Access control – each chunk of memory

- Good / bad ?

Segmentation

- Code / data segments

- NAME / OFFSET

- OS translates all references

- Problem: Segment size

Integrity Same concept of X and I
I has the property of integrity with respect to X if all members of X trust I

Availability X can access I
What is access?

Other thoughts

Information flow Leakage of rights, covert channels
Dynamic changes (time sensitive)

Mechanisms vs. policy

Book example CS students -> work on a computer
Policy: Do not copy

1st student copies homework of second student that did not read protect their files

Who breached security?

2nd student No -> Too trusting but that is not a breach
1st student Yes -> Policy against copying

Mechanism Enforces policy
Mechanism Set to add read protection

Key: Be explicit with the policy, not implicit

Book example Original UNIX

Prevent accidental damage of other user's files
No deletion

Implied: Do not delete or corrupt other files, any file not protected may be read

Consider university/business setting

Security Policies

Military policy Confidentiality
Commercial policy Integrity

Trust

Confidentiality
Integrity

Green Eggs and Ham can be top secret
Should this program be trusted?

Trust

Critical to understanding security

Explicit trust
Implicit trust

I know I trust object B
I know I trust object B and object B trusts method C
so I implicitly trust object C

Discussion: Voting precinct

Discussion: DRM

Key: Trust analysis

Understand what you are trusting and what it trusts so you understand the strengths and weaknesses of the system