Isolation and Flow of Information

CMMRS 2019 (1/3)

Saarbrucken, Germany
August 2019

Fred B. Schneider
Samuel B Eckert Professor of Computer Science

Department of Computer Science
Cornell University
Ithaca, New York 14853
U.S.A.

Cornell CIS
Computer Science

Why study security?

- Society increasingly depends on having networked systems that are trustworthy.
- Technically interesting:
  - trace properties → hyperproperties
  - resist an unknown adversary

A 3 lecture snapshot …

I. Overview: Terminology and metaphors

II. Isolation: New view on a classic idea

III. Information flow: Visit the frontier

Lecture I: Goals

Framework for thinking about computer security.

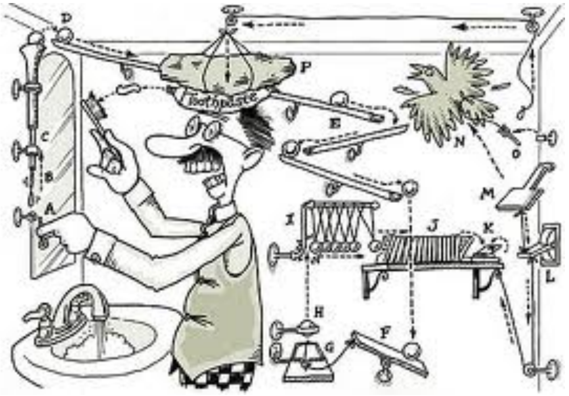
- Introduce vocabulary used by practitioners.
- Understand principles that underpin computer security.
- Discuss interface between technical and policy.

Trustworthy Systems?

A **trustworthy** system will

- do what is expected
- not do the unexpected

despite attacks and failures, and offers assurance about this claim.



Example:

- Do more: reveal secrets
- Do less: fail to store or retrieve information

What to protect?

Attacks compromise:

- **Secrecy (confidentiality)** causing improper disclosure of information.

  … But what constitutes a secret, anyway?

- **Integrity** causing improper alteration of information or use of resources.

- **Availability** causing service outages.

Protect against what?

Terminology:

- **vulnerability**: Weakness that can be exploited to cause damage.
- **attack**: Method of exploiting a vulnerability.
- **threat**: Motivated capable adversary who will mount attacks.

All systems have vulnerabilities.

Understand the threats and defend against attacks they can mount.

All assumptions are vulnerabilities.

Cyber threats

- Operator/user blunders.
- Hackers driven by intellectual challenge (or boredom).
- Insiders: employees or customers seeking revenge.
- Criminals seeking financial gain.
- Organized crime seeking gain or hiding criminal activities.
- Organized terrorist groups or nation states trying to influence national policy.
- Foreign agents seeking information for economic, political, or military purposes.
- Tactical countermeasures intended to disrupt military capability.
- Large organized terrorist groups or nation-states intent on overthrowing the government.

Cyber threats: Classification

Class I: Execute existing attacks against known vulnerabilities.

Class II: Analyze system, find new vulnerabilities, develop new attacks.

Class III: Create new vulnerabilities (e.g., compromise the supply chain).

Cyber threats: Classification

Access based:

- Physical access
- Software access
- User access

Capability based:

- Computational (probabilistic polynomial time TM)

Security in the “real world”

Use locks to block attacks: “Prevention”

- Locks must not be annoying, or they won’t be used.
- All locks aren’t the same. They are:
    - Scaled for what they are protecting.
    - Scaled for their environment.
- Police and courts are central---not the locks!

  Expect security breaches.

    - Tracking down the “bad guys” is what’s central.
    - Locks reduce temptation and reduce workload on police and courts.

“Real world” (con’t)

- People only pay for security that they think they need.

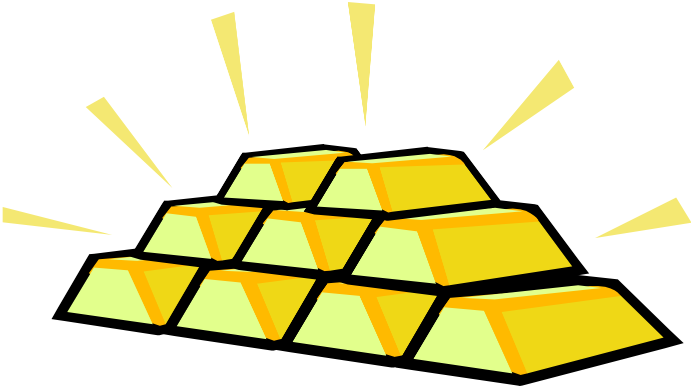
  … need is based on personal experience & others experiences.

- People avoid annoying locks by buying insurance.
  - Risk avoidance versus risk management.
  - Externalities is a wrinkle!
- Security is holistic…
  - Security is only as strong as the weakest link.
  - Making any link stronger than the weakest link doesn’t much improve security.

Locks in cyberspace

Computer Security "Gold Standard"

- Authentication
- Authorization
- Audit



(N.B. Au is the chemical symbol for Gold.)

Security Mechanism Design

- Central concerns:
  - What does the mechanism do?
  - Why believe it works? Under what assumptions?
- Some principles:
  - Functionality Principles.
  - Assurance Principles.

Approaches to Assurance

What basis to *trust* C:

- Axiomatic: Accepted on faith.
  - producer, certification of producer, …
- Analytic: Accepted based on analysis:
  - testing, type-checking, verification.
- Synthetic: Accepted based on construction.
  - mechanism design principles (to come)…

Mechanism Design:
Assurance Principles: Economy

Economy of Mechanism: Use small and simple mechanisms where possible.

Consequences:

- Fewer errors in implementation because simpler.
- Easier to analyze for yourself.

Mechanism Design:
Assurance Principles: Open Design

Open Design: Security of a mechanism should not depend on an attacker's ignorance of the design.

A. Kerkhoffs Principle (1883): The security of a cryptosystem must not depend on keeping the algorithm secret.

No security by obscurity.

Consequences:

- Increased assurance if many critics.
- Reduced cost of recovering from key compromise.

Mechanism Design:
Assurance Principles: Open Design?

Open Design is controversial.

With open design:

- Attackers job is easier because design is available.
- Analysis tends to concentrate on certain “main code”. Vulnerabilities off the beaten path remain.
- Flaws are not always revealed.

Open source:

- Economic model
- Limited access to newest tools?

Mechanism Design:
Functionality Principles

What should the mechanism do?

Best to distinguish **policy** from **mechanism**.

Desire mechanisms that implement many policies.

Mechanism Design:
Functionality Principles

Principle of Least Privilege: "Every program and every user of the system should operate using the least set of privileges necessary to complete the job."

J.H. Saltzer and M.D. Schroeder,
The Protection of Information in Computer Systems.
Proc. Of the IEE 63, 9 (Sept 1975),
pp 1278-1308.

Consequences:

- Limits Damage that can result from attack or error.
- Limits number of programs that can be compromised to effect an attack.
- Helps with debugging.

Example: super-user versus admin privileges.

Mechanism Design: Functionality Principles

Corollary of Principle of Least Privilege:

Complete Mediation: Every access to every object is checked.

Some implicit assumptions:

- Some interface is being monitored.
- Mediation mechanism cannot be compromised.

Mechanism Design:
Functionality Principles

Corollary of Principle of Least Privilege:

Failsafe Defaults: Access decisions are based on the explicit presence of permissions rather than their absence of explicit prohibitions.

Safe way to tolerate administrative oversight.

Mechanism Design: Functionality Principles

Corollary of Principle of Least Privilege:

Separation of Privilege: Each “lock” should require a separate “key”.

Consequences:

- Allows fine-grained control and therefore supports PoLP with higher fidelity.
- Can be a sys admin nightmare.

Where to deploy mechansim

Axiom: Every system has vulnerabilities!

Consequently…

- Employ multiple lines of defense.

  … this is just Separation of Privilege!

- Employ diversity of mechanism.

  … diverse mechanisms are unlikely to share vulnerabilities.

For additional information

Introduction Fred B. Schneider. Untitled draft textbook: http://www.cs.cornell.edu/fbs/publications/chptr.Intro.pdf

Not only a technical problem …

- Existing technical solutions not deployed due to:
  - Increased expense and delay for developers.
  - Less convenience for users.

- New technical solutions are needed, too.
  - Conversations required between technical and policy communities.

To Invest in Trustworthiness …

The obvious recipe:

- Decide to invest.
  - *How much? Expected return?*
- Explore regulatory or other policy mechanisms.
  - *Which ones work? Side effects and trade-offs?*
- Package to create incentives.
  - *Best to embrace a* ***doctrine****.*

Deciding to Invest: Who?

Who pays?

- consumers
  - price, delay, functionality, convenience, values
- government
  - tax credits or grants
- investors
  - profits

How should costs be allocated across sectors?

Deciding to Invest: What?

Security goals?

- Who is the adversary?
  - Uses known attacks
  - Invents new attacks
  - Creates new vulnerabilities.
- What policies must be enforced?
  - Known vs unknown interfaces
  - Known vs unknown specification

Deciding to Invest: When?

Investment must be recurring.

- Bugs must be patched
- Deployment environment changes
  - developer assumptions invalidated
  - unanticipated uses must be supported.

Business model for recurring investment?

- Sale vs license
- Low-cost and disposable vs maintainable