Software and Security Engineering

Lecture 1

Alastair R. Beresford

arb33@cam.ac.uk

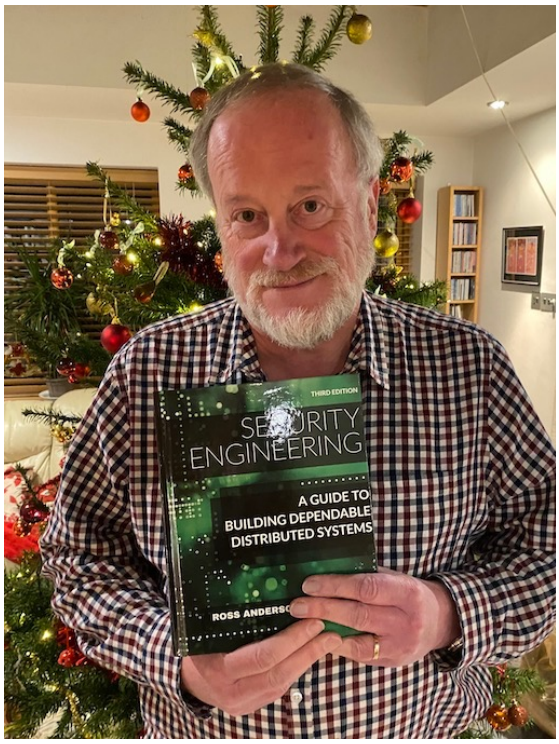
With many thanks to Ross Anderson

1

This course is usually lectured by Prof Ross Anderson and therefore much of the material is derived from an earlier version of the lecture course prepared by him.

Ross Anderson

1956 - 2024



2

This course is usually lectured by Prof Ross Anderson and therefore much of the material is derived from an earlier version of the lecture course prepared by him.

Course outline and lecturers

2 May	1	What is a security policy or a safety case?	Prof Beresford
5 May	2	Examples of safety and security policies	Prof Mortier
7 May	3	Attitudes to risk	Prof Mortier
9 May	4	The software crisis	Prof Madhavapeddy
12 May	5	Software engineering as the management of complexity	Prof Madhavapeddy
14 May	6	Software testing	Prof Madhavapeddy
16 May	7	Security protocols	Dr Kleppmann
19 May	8	Software as a Service (SaaS)	Dr Sharp
21 May	9	Attacks on TLS	Dr Kleppmann
23 May	10	Decentralised social protocols	Dr Kleppmann
26 May	11	Critical systems	Prof Harle

Aims

- Introduce software engineering with focus on:
  - Large systems
  - Safety-critical systems
  - Systems to withstand attack by capable opponents
- Illustrate what goes wrong
- Best practice to avoid failure

4

So far in Part 1A you have written small sample programs by yourself. Most software development in industry is at a significantly larger scale, involving teams of people and often involves safety or security requirements.

All significant pieces of software contain latent defects – bugs yet to be discovered. This affects both safety and security.

In this course we look at what has gone wrong in the past through case histories, and look at the development and management practices which have arisen in order to avoid failures in the future.

Objectives

- By the end of the course, you should be able to:
  - Write programs with tough assurance targets
  - Work effectively as part of a team
- Understand
  - Software development models
  - Development lifecycle
  - Understand bugs, vulnerabilities and hazards

5

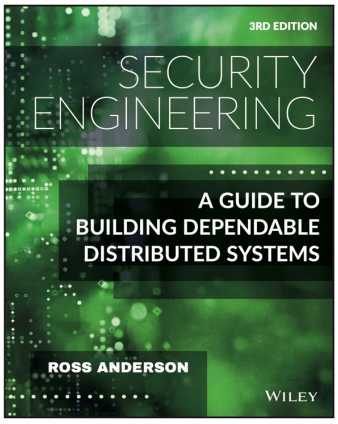
In order to write programs which meet tough assurance targets and work effectively as a team member, you need to be able to apply appropriate programming best practice as described in this course. For example, how to use version control for source code, automated build systems, and suitable testing strategies.

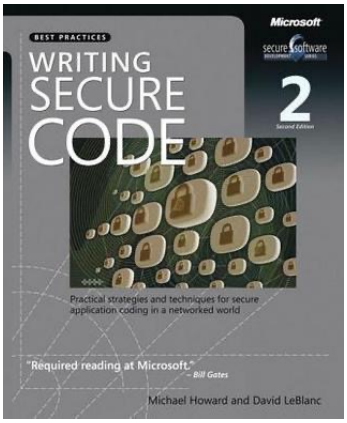
An understanding of historical and current software development models such as waterfall, spiral and agile methods will allow you to select the right approach for a given project.

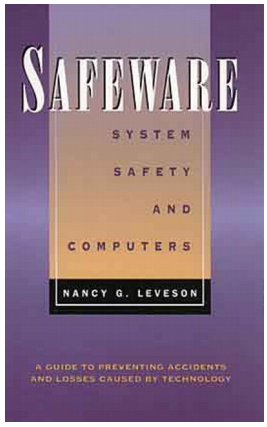
It is important to understand the terminology used (e.g. what is a *bug*, a *hazard*, or a *vulnerability*?) so you can correctly understand case histories and can communicate effectively with others.

An important aim is to prepare you for the Part IB group project. The techniques described here will also help with your Part II and Part III projects as well as other later courses such as Security and Concurrent and Distributed Systems.

Books







6

This is a course which requires you to read around the subject. Textbooks are very helpful for this course, and here are three which offer useful, complementary perspectives.

R. Anderson, *Security Engineering* (3rd Edition, 2020). You may like to start by reading Part I and also Chapters 25 and Chapter 26. The 2nd Edition is also very relevant and available online: https://www.cl.cam.ac.uk/~rja14/book.html

M. Howard and D. LeBlanc, *Writing Secure Code* (2nd Edition, 2003).

N. Leveson, *Safeware: System Safety and Computers* (1994). See also her more recent book, *System Safety Engineering: Back To The Future* (2002), which is also available online: http://sunnyday.mit.edu/book2.pdf

Make use of additional reading

F.P. Brooks, *The Mythical Man Month*

J. Reason, *The Human Contribution*

S.W. Thames, *Report of the Inquiry into the London Ambulance Service*

S. Maguire, *Writing Solid Code*

H. Thimbleby, *Improving safety in medical devices and systems*

O. Campion-Awwad et al, *The National Programme for IT in the NHS – A Case History*

https://www.cl.cam.ac.uk/teaching/current/SWSecEng/materials.html

7

In addition to the core textbooks, these are some of the additional textbooks, academic articles and white papers which you may find interesting.

The course materials page contains further links to related reading and an annotated slide deck:
https://www.cl.cam.ac.uk/teaching/current/SWSecEng/materials.html

You should not feel constrained to these materials. Read about any application areas which are interesting to you. This will help both with your general understanding and also provide you with useful perspectives and examples which you can use to inform your future career as well as supporting material to refer to in the forthcoming Tripos examinations.

Remember: wide-reading driven by curiosity will help a lot with this course.

Use the ICAP framework to guide your learning

- **I**nteractive
- **C**onstructive
- **A**ctive
- **P**assive

"Teachers open the door,
But you must enter by yourself.
Tell me and I forget.
Teach me and I remember.
Involve me and I learn."
– Benjamin Franklin

Or: reading is essential but insufficient

8

"interactive activities are most likely to be better than *constructive* activities, which in turn might be better than *active* activities, which are better than being *passive."*

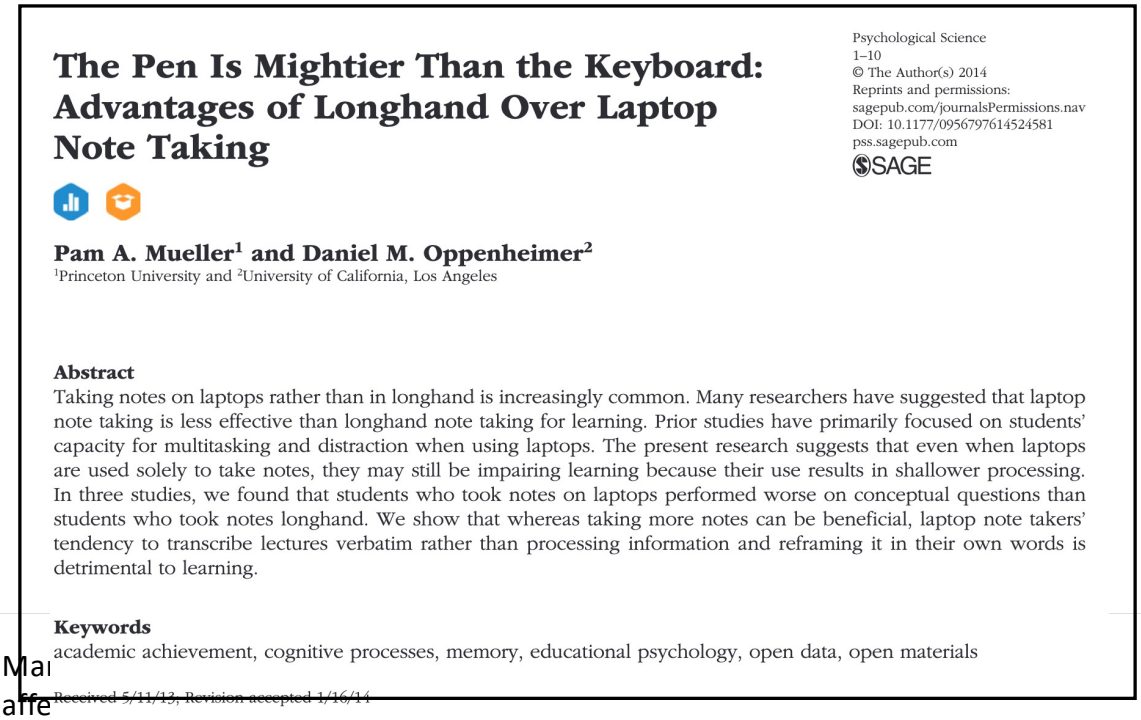
Michelene Chi, Active-Constructive-Interactive: A Conceptual Framework for Differentiating Learning Activities, Topics in Cognitive Science, 1(1):73-105 2009. https://onlinelibrary.wiley.com/doi/full/10.1111/j.1756-8765.2008.01005.x

Examples of different types of learning:

- Interaction: discussing with peers, a supervision
- Construction: completing an example sheet, writing a summary in your own words
- Active: Taking notes of what the lecturer says, highlighting a passage
- Passive: Reading a book, listening to a lecture or video

Takeaway message: You need to read books and papers, but to really understand the material, you need to build artefacts, talk to others and critique the ideas.

Using laptops in lectures can harm everyone's learning outcomes



The Pen Is Mightier Than the Keyboard: Advantages of Longhand Over Laptop Note Taking

Psychological Science
1–10
© The Author(s) 2014
Reprints and permissions:
sagepub.com/journalsPermissions.nav
DOI: 10.1177/0956797614524581
pss.sagepub.com
SAGE

Pam A. Mueller[1] and Daniel M. Oppenheimer[2]
[1]Princeton University and [2]University of California, Los Angeles

Abstract
Taking notes on laptops rather than in longhand is increasingly common. Many researchers have suggested that laptop note taking is less effective than longhand note taking for learning. Prior studies have primarily focused on students' capacity for multitasking and distraction when using laptops. The present research suggests that even when laptops are used solely to take notes, they may still be impairing learning because their use results in shallower processing. In three studies, we found that students who took notes on laptops performed worse on conceptual questions than students who took notes longhand. We show that whereas taking more notes can be beneficial, laptop note takers' tendency to transcribe lectures verbatim rather than processing information and reframing it in their own words is detrimental to learning.

Keywords
academic achievement, cognitive processes, memory, educational psychology, open data, open materials
Received 5/11/13; Revision accepted 1/16/14

The use of laptops in classrooms is controversial. Many professors believe that computers (and the Internet) serve as distractions, detracting from class discussion and important but relatively unsurprising, given the literature on decrements in performance when multitasking or task switching (e.g., Iqbal & Horvitz, 2007; Rubinstein, Meyer,

Mar
affe
Stu as
see n)
for this course.

If you really want to use a laptop, then try and summarise what I say (a constructive activity), don't transcribe the lecturers presentation verbatim (merely an active learning activity). If you want to browse the Internet, then please do that elsewhere, not in the lecture. (Lecture attendance in Cambridge is not compulsory.)

Paper reference: Pam Mueller and Daniel Oppenheimer. The Pen Is Mightier Than the Keyboard: Advantages of Longhand Over Laptop Note Taking, Psychological Science, 1(10), 2014. https://cpb-us-w2.wpmucdn.com/sites.udel.edu/dist/6/132/files/2010/11/Psychological-Science-2014-Mueller-0956797614524581-1u0h0yu.pdf

Further reading: https://cs.brown.edu/courses/cs019/2018/laptop-policy.html

What is Security Engineering?

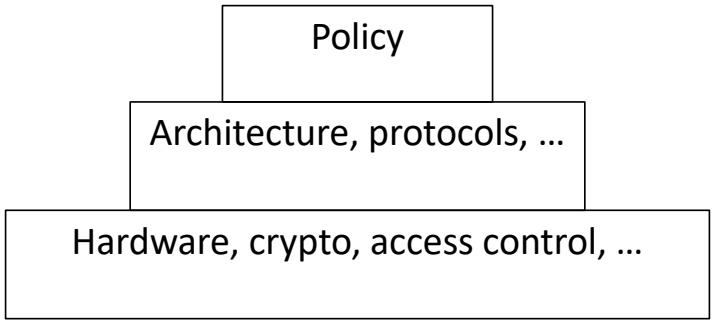
Security engineering is about building systems to remain dependable in the face of malice, error and mischance.

10

Security Engineering as a discipline focuses on the tools, processes and methods needed to design, implement and test complete systems, and to adapt existing systems as their environment evolves. Note that safety engineering has a similar high-level goals.

When considering the security or safety of the system, it is not sufficient to look at each component in isolation (although that is important). It is essential to look at how the whole system fits together. Security and safety is not composable.

The Design Hierarchy



What are we trying to do? How? With what?

11

The traditional design hierarchy requires us to start by defining, at a high level, what we are trying to achieve. Then we explore the question of how to do so in terms of overall strategy and architecture. Finally we need to explore the detail: which hardware platform should we use, what cryptographic primitives are the right ones, and so on.

Waymo self-driving car

Picture taken by Alastair Beresford, May 2024.

A system can be…

- equipment or a component (laptop, smartcard, …)
- a collection of products, their operating systems, and some networking equipment
- The above plus applications
- The above plus internal staff
- The above plus external users

Common failure: policy drawn too narrowly

13

The definition of the system is often too narrowly defined.

For example, if a company produces a mobile app for a smartphone, it might assume that the app is the system. This is too narrow a definition. For example, what about the operating system running on the phone, or its hardware? Does the operating system get regular updates? What other apps are installed on the phone, and are they malicious? What about all the servers that the app communicates with?

We study cybercrime in the department. Most of the failures are not hi-tech, but rather use a system in bad and unintended ways. Cyberbullying via a messaging platform is one example: the platform is delivering the messages as requested, but overall the system is failing to protect users from harm.

Electric bike should not propel bicycle when speed exceeds 15.5 mph

14

Here is an example where you need to think broadly about the definition of the "system".

In the UK, in common with many countries, electric bikes can only be ridden without a license, tax or registration provided it has pedals and electric assistance does not occur when travelling over 15.5 mph (25 km/h). So how do cyclists get around this restriction?

The bicycle in this picture has electric assist. The electronics in the bike estimates the speed of the bike by counting revolutions of the rear wheel passing in front of a sensor. The "badassbox" works by supressing the sensor reading in every other revolution, allowing you to travel twice as fast with electric assist. A by-product of this is that the speedometer on the bike no longer provides an accurate reading.

Further reading:

- Details on the badass box, https://www.ebiketuning.com/badass-box-4-for-shimano.html
- Rules around electric bikes in the UK, https://www.gov.uk/electric-bike-rules

Definitions

Security vs Dependability

Dependability = Reliability + Security

- Malice is different from error
- Reliability and security are often strongly correlated

16

Reliability and dependability sounds like they might mean the same thing. However, we demand greater precision in our use of terms.

Since malice is different from error, we wish to capture this. For example, a system might state a reliability guarantee, such as “Bob will be able to read this file”, while a security guarantee might state that ”Foreign governments won’t be able to read this file”. Typically we want a dependable system with both reliability and security.

This example motivates the need to define terms carefully. Note that while we will be consistent in this course, terms may have different meanings (or different terms have the same meaning) in different communities. For example, the safety and security communities currently use different language.

Subjects and principals

Subject: a physical person

Person: a subject or a legal person (firm)

Principal:

- A person
- Equipment
- A role, including complex roles

17

We adopt the same language as used by the legal profession and define a person as either a *subject* (physical person) or a legal person which can also include a limited company (e.g. Google) or a charity (University of Cambridge).

We use the term principal as a more general term to cover people, equipment and more general labels. The term role is often used to as a means of indirection between a principal and a person. For example, “the officer on watch”, or “Alice and Bob” or “Alice or any of her current direct reports”.

The definition of a principal can get quite complex. Sometimes we need to distinguish between “Bob’s smartcard representing Bob who’s standing in for Alice” from “Bob using Alice’s card in her absence”. For example, consider the case of a bank, whose policy states that all withdrawals over 10,000 GBP must be approved by any two bank managers out of the set of Alice, Bob or Charlie.

Secrecy and privacy

Secrecy: mechanism to control which principals can access information

Privacy: control of your own secrets

Confidentiality: an obligation to protect someone else's secrets.

18

Secrecy often, but not always, implies a technical mechanism. This does not necessarily involve cryptography.

Privacy has many definitions which are wider than the one used in this course. For those who are interested in such things, you might wish to look up the right to be forgotten.

These three concepts are interrelated. For example, your medical privacy is protected by your doctors' obligation of confidentiality.

Anonymity, integrity, authenticity

- *Anonymity*: restrict access to metadata
- *Integrity*: an object has not been altered since the last authorised modification
- *Authenticity* has two common meanings:
  - an object has integrity plus freshness
  - You are speaking to the right principal

19

Anonymity has various flavours, from not being able to identify subjects to not being able to link their actions. A simple example is k-anonymity where subjects are indistinguishable from k-1 others (subjects are said to be “k-anonymous”).

A cheque has integrity (a signature) and freshness (a recent date) together giving it authenticity.

Trust is hard; several meanings…

1. A warm fuzzy feeling
2. **A trusted system or component is one that can break my security policy**
3. A trusted system is one I can insure
4. A trusted system won't get me fired when it breaks
5. …

Trust is really hard. It can exist at different levels: human norms (you trust your doctor, and he has a warm manner, a nice office, etc). Trust can also be a trusted system. Are these the same thing? Yes and no. Yes, because the doctor can break a trusted system by malice or by accident (writes down his password which is visible to others); no in other cases.

We are going to use the second definition (the NSA definition) for this course. Under this definition, an employee of GCHQ selling cryptographic key material to a foreign power is *trusted* but not *trustworthy* (assuming of course that such a sale has not been authorized).

Errors, failures, reliability, accidents

- *Error*: a design flaw or deviation from intended state
- *Failure*: nonperformance of the system when inside specified environmental conditions
- *Reliability*: probability of failure within a specified period of time
- *Accident*: an undesired, unplanned event resulting in a specified kind or level of loss

21

These are terminology from the safety community.

Failure is often expressed as Mean-Time-Before Failure (MTBF), or Mean-Time-To-Failure (MTTF). For example, a single-engine plane might have an MTBF of 240,000 hours overall. This isn't necessarily a meaningful summary on its own – other steps might be required to ensure this is the case. For example, it might have an MTBF of 5,000 hours if its not serviced correctly (e.g. oil change is forgotten).

Hazards and risks

- *Hazard*: a set of conditions in a system or its environment where failure can lead to an accident
- A *critical* system, process or component is one whose failure will lead to an accident
- *Risk* is the probability of an accident
  - Often combined with *unit of exposure*; e.g. a *micromort*
- Uncertainty is where the risk is not quantifiable
- Safety is simple: freedom from accidents

22

In a single-engined aircraft, a hazard might be the mountain you fly over at night since you will crash if the engine fails. There is less hazard due to engine failure when flying over the East Anglia during the day since its flat and you can probably land safely in a field somewhere.

David Speigehalter uses the "micromort" as a unit of risk, defined as a one-in-a-million chance of death. For example, taking the data from the Office of National Statistics for 2012, 499,331 people died in England and Wales out of a population of 56,567,000. Therefore, the chance of death overall for each citizen is, on average, 24 micromorts per day. We can use this concept of micromorts per unit of exposure to assess the comparative risk of activities. For example, data can be used to estimate that scuba diving is 5 micromorts per dive; skydiving, 8 micromorts per jump; and skiing, 0.7 micromorts per day.

So what about terrorism? It has a tiny micromort! Yet we still care – because humans are not always rational.

It is worth highlighting that risk, or the probability of an accident, is different from the probability of failure or MTBF. A component can fail without it causing an event resulting in loss. In a twin-engine plane, one engine can fail and yet there is no accident.

Policies, multilevel and multilateral security

Security policy, profile, and target

- A *security policy* is a succinct statement of protection goals
- A *protection profile* is a detailed statement of protection goals
- A *security target* is a detailed statement of protection goals applied to a particular system

A security policy is typically less than a page of text written in plain language.

A protection profile is typically dozens of pages written in a semi-formal language.

A security target may run to hundreds of pages for both functionality and testing.

What often passes as ‘policy’

1. This policy is approved by Management.
2. All staff shall obey this security policy.
3. Data shall be available only to those with a need-to-know.
4. All breaches of this policy shall be reported at once to Security.

What’s wrong with this?

25

[Ask the audience to talk to a neighbour and producing a list of problems with this policy]

Many things wrong with this policy. Examples include:

* The policy reduces to "need-to-know", but what does this mean?
* What's a "breach"?
* Reporting a breach is passive voice: who does the reporting of a breach? Over what time period?
* You need to trust the employees to adhere to policy. Do they feel trusted and empowered?
* There's nothing in this policy which you can implement (e.g. support employees, or turn into software): this policy is *security theatre*.

In the UK there's no general requirement to report a crime (with the exception of terrorism).

Edward Snowden is an example worth considering: he released lots of data because he felt that his duty as a solider was to leak data since, in his opinion, what the NSA was doing was contrary to the constitution, and he had signed up to protect the constitution.

Traditional government approach

- Start from the threat model: an insider who is disloyal or careless.
- Solution: limit the number of people you trust, and make it harder for them to be untrustworthy

Basic idea since 1940: a clerk with 'Secret' clearance can read documents at 'Confidential' and 'Secret' but not at 'Top Secret'

26

Examples of disloyal insiders include Burgess/MacLean, Aldrich Ames, Edward Snowden. Carelessness can include "loose talk", reading papers on train, being photographed outside Number 10 with papers in hand, malware on PC, and so on.

Another important concept is *vetting,* in which the background of employees working with sensitive information is investigated. Does the employee have any weaknesses which might be exploited by a third-party? For example, does the employee have unsustainable debts, a drinking or drug problem, can they be "bought off", etc.

Multilevel Secure Systems (MLS)

- Classify all documents and data with a level, such as *official*, *secret*, *top secret*; or *high* and *low*.
- Principals have clearances; clearance must equal or exceed classification of any documents viewed.
- Enforce handling rules for material at each level.
- Information flows upwards only:
  - No read up
  - No write down

27

MLS are widely used by governments.

There are a variety of classification levels in use around the world. The UK Government uses Official, Secret, Top Secret (https://en.wikipedia.org/wiki/Government_Security_Classifications_Policy). In this course, in common with much of the computer science literature, we will often use High and Low to describe a simple abstract representation of an MLS system.

Recall 'mandatory access control' from OS course.

Information flows are integrated with the operating systems as used by government employees. Clearly the OS needs to prevent employees with clearance to work with material up to secret level from accessing any files classified at top secret. No write down is also important to stop information leakage. For example, the OS will allow the user to cut-n-paste data from a confidential file to a secret one, but not vice-versa (or if you can, then the confidential file then becomes classified at secret).

Bell-LaPadula formal model

- Bell-LaPadula (1973):
  - *simple security policy* (no read up)
  - **-policy* (no write down)
- With these two rules, one can prove that a system that starts in a secure state will remain in one
- Aim is to minimise the Trusted Computing Base

28

At first people thought that you only needed no read up, but then if you get malware running at high, it can leak data to low; so we also need the *-property.

The Trusted Computing Base includes all the hardware and software required to enforce security policy.

Covert channels cause havoc

- BLP lets malware move from Low to High, just not to signal down again.
- What if malware at High modulates shared resource (e.g. CPU usage) to signal to Low?
- How can you let message traffic pass from Low to High, if any acknowledgement of receipt could be delayed and used to signal?

Moral: covert channel bandwidth is a complex.
It's an emergent property of whole systems!

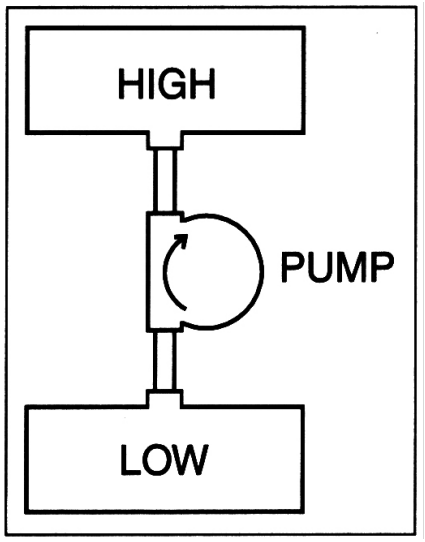
29

A covert channel occurs when the performance of a resource shared between Low and High allows information to flow which contravenes policy. An example shared resource might be a CPU shared between processes, some of which are running at High and some at Low. Then a High process can transfer data to a Low process by either using lots of CPU (to send a "one") or not using the CPU (signaling a "zero").

More information: https://en.wikipedia.org/wiki/Covert_channel

High assurance MLS system

- The pump simplifies the problem: replace the complex emergent property of the whole system with a simple property of a testable component
- Nevertheless, often harder than it looks!

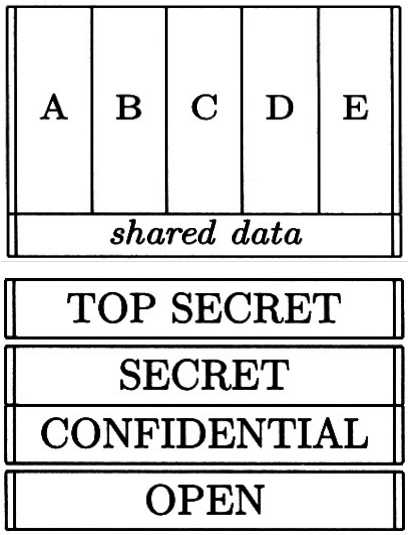


30

Multilateral Security

Stop lateral flow, examples:

- Intelligence, typically with compartments
- Medical records
- Competing clients of an accounting firm



31

MLS is good at stopping data from flowing from High to Low. In other settings, you want to stop lateral flows of information. Accounting firms use this to allow them to work for two or more firms who compete in the same sector. Accountants at one of the Big Four working on accounts for BP need to make sure that they don't talk to colleagues who are working on Shell's accounts.

Biba formal model for integrity

- Biba (1975)
  - Simple integrity policy (no read down)
  - *-integrity policy (no write up)
- Dual of the Bell-LaPadula model
- Examples:
  - Medical devices with *calibrate* and *operate* modes
  - Electricity grid controls with safety at the highest level, operational control as the next, and so on.

32

The Biba model is the inverse of the Bell-LaPadula model: instead of protecting confidentiality, it protects integrity. In this model, we do not want a process at High from being influenced by data from Low.

For example, a nuclear power station will have safety as at the top level: and any control of the power station for the safety of the power grid itself will be ignored (a blackout is better than a nuclear meltdown).

In practice, safety systems need more than simply High and Low. Compartmentalization is often a good way forward, and to do this we can apply the multi-lateral security model.

Recap: ICAP, systems and policy

- Read widely (necessary, but not sufficient)
- ICAP
- Consider a system in broad terms
- Make sure you use terms precisely
- Define policy carefully
- Multilevel Secure Systems (MLS); Bell LaPadula
- Multilateral Security; Biba

33

Subsequent lectures

2 May	1	What is a security policy or a safety case?	Prof Beresford
5 May	2	Examples of safety and security policies	Prof Mortier
7 May	3	Attitudes to risk	Prof Mortier
9 May	4	The software crisis	Prof Madhavapeddy
12 May	5	Software engineering as the management of complexity	Prof Madhavapeddy
14 May	6	Software testing	Prof Madhavapeddy
16 May	7	Security protocols	Dr Kleppmann
19 May	8	Software as a Service (SaaS)	Dr Sharp
21 May	9	Attacks on TLS	Dr Kleppmann
23 May	10	Decentralised social protocols	Dr Kleppmann
26 May	11	Critical systems	Prof Harle