UNIVERSITY OF CAMBRIDGE
Computer Laboratory

Security II

2011-2012

Frank Stajano and Steven Murdoch

+ 5 guest lecturers

Computer Science Tripos Part II

University of Cambridge

Based on slides by Ross Anderson, with changes

Aims

- Give you a thorough understanding of information security technology
  - Policy (what should be protected)
  - Mechanisms (cryptography, hardware security…)
  - Attacks (malicious code, protocol failure…)
  - Assurance (assessing how secure it is)
- How do we make this into a proper engineering discipline?

Objectives

By the end of the course, you should be able to tackle an information protection problem by

- drawing up a threat model
- formulating a security policy and
- designing specific protection mechanisms to implement the policy.

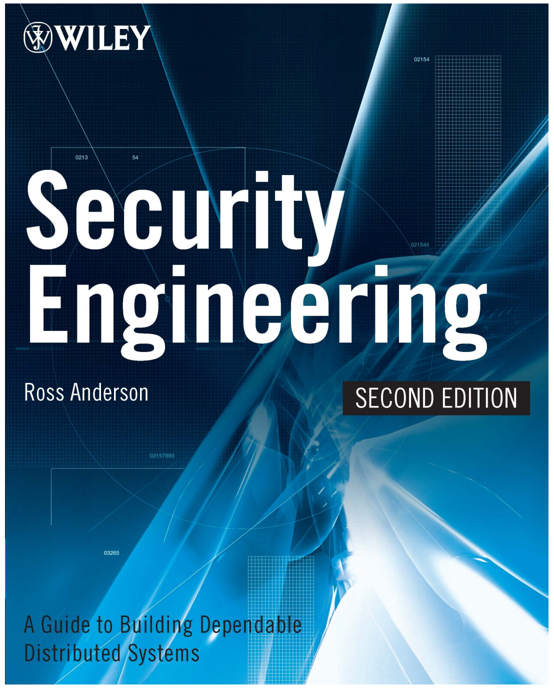
Broad range of topics

Not an "axiom, theorem, exercise" subject

You must learn to think outside the box

- Human factors, Security Policy, Crypto, Protocols, Incentives etc
- Many guest lectures, to broaden horizons further:
  - Richard Clayton, security economics
  - George Danezis, privacy technology
  - Joe Bonneau, web authentication
  - Robert Watson, concurrency vulnerabilities
  - Sergei Skorobogatov, hardware security

Resources



- Anderson: *Security Engineering*
  - developed from lecture notes
- Web page for course
- Menezes, van Oorschot and Vanstone: *Handbook of Applied Cryptography* (reference)
- Stinson: *Cryptography: theory and practice*
- Schneier: *Applied cryptography*
- Gollmann: *Computer Security*

Resources

- History:
  - David Kahn: *The Codebreakers*
  - Gordon Welchmann: *The Hut Six Story*
- Specialist:
  - Biham and Shamir: *Differential Cryptanalysis*
  - Koblitz: *Course in number theory and cryptography*
  - Stajano: *Security for Ubiquitous Computing*
- **Use the Source, Luke:**
  - Read the original papers (books come after papers)
- Lab:
  - Security seminars, usually Tuesdays, 1615-1715
  - Security group meetings, Fridays, 1600-1700

www.cl.cam.ac.uk/teaching/current/SecurityII/materials.html

What is Security Engineering?

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

As a discipline, it focuses on the tools, processes and methods needed to design, implement and test complete **systems**, and to adapt existing **systems** as their environment evolves.

Systems!

- A *system* can be:
  - a product or component (PC, smartcard,…)
  - some products plus O/S, comms and infrastructure
  - the above plus applications
  - the above plus internal staff
  - the above plus customers / external users
- Common failing: policy drawn too narrowly
  - Want a secure system? You need to consider *users*

Human factors

Whitten, Tygar: ‘Why Johnny Can’t Encrypt’, 1999.

- Study of encryption program PGP – showed that 90% of users couldn’t get it right give 90 minutes
- Private / public, encryption / signing keys, plus trust labels was too much – people would delete private keys, or publish them, or whatever
- Security is hard – unmotivated users, abstract security policies, lack of feedback…
- Geeky “security experts” would rather deal with machines than with unpredictable people. They miss the point.

Users are not the enemy

Adams, Sasse: “Users are not the enemy”, 1999.

- Insufficient communication with users produces unusable systems
- Users forced to comply with password mechanisms incompatible with work practices will look for workarounds
- Vicious circle:
  - Security departments think users are inherently insecure
  - Users think security departments get in the way of real work
- But “users never motivated to behave securely” is wrong!
- Treat users as stakeholders and they'll cooperate
- Provide feedback, guidance, awareness; and usable security

Think like an attacker

- Mitnick: *The art of deception*, 2003.
  (Also *The art of intrusion*, 2005.)
  - You don't have to pick the lock or break into the server: get someone on the inside to open the door for you
  - Pretext calls surprisingly effective
- Traditional responses:
  - mandatory access control
  - operational security

  But why do the attacks work?

Think like a victim

Stajano, Wilson: “Understanding scam victims”, 2009-11.

- Learn from fraudsters: they know how to push the victims' buttons
- The Real Hustle (BBC3): hundreds of scams recreated for hidden cameras

What makes these scams work?

Understanding scam victims

- **Distraction –** When focused, you forget the rest
- **Herd** - If others do it, it's probably ok
- **Need and greed** - There are things you just can't say no to
- **Dishonesty** - It's illegal, that's why it's such a good deal
- **Kindness** - People are actually nice
- **Authority** - If the man in uniform says so, you do it
- **Time and scarcity –** If there isn't enough of it, it must be good

Exact principles not actually that important...

But their existence is!

Social Psychology

- Solomon Asch, 1951: two-thirds of subjects would deny obvious facts to conform to group
- Stanley Milgram, 1964: a similar number will administer torture if instructed by an authority figure
- Philip Zimbardo, 1971: you don't need authority: the subjects' situation / context is enough

  Cfr Herd principle and Authority principle

- The Officer Scott case

• What about users you can't train (your customers)?

Weapons of influence

Cialdini: *Influence: science and practice*, 5th ed 2009.

Based on undercover study of salesmen "and other compliance professionals"

Salesmen are like scam artists, except legal (perhaps)

_ Reciprocation: they'll feel compelled to respond
_ Commitment and consistency: "but you previously said X"
_ Social proof: like to do what others do
_ Liking: want to deal with people they can relate to
_ Authority: will defer to authority figure
_ Scarcity: less is best and loss is worst

Framing effects include 'Was £8.99 now £6.99' and the estate agent who shows you a crummy house first. Take along an ugly friend on a double date.

Phishing

- Started in 2003 with six reported (there had been isolated earlier attacks on AOL passwords)
- By 2006, UK banks lost £35m (£33m by one bank) and US banks maybe $200m
- Early phish crude and greedy; but phishermen learned fast
- E.g. ‘Thank you for adding a new email address to your PayPal account’
- The banks make it easy for them

Types of phishing website

- Misleading domain name

  `http://www.banckname.com/`

  `http://www.bankname.xtrasecuresite.com/`

- Insecure end user

  `http://www.example.com/~user/www.bankname.com/`

- Insecure machine

  `http://www.example.com/bankname/login/`

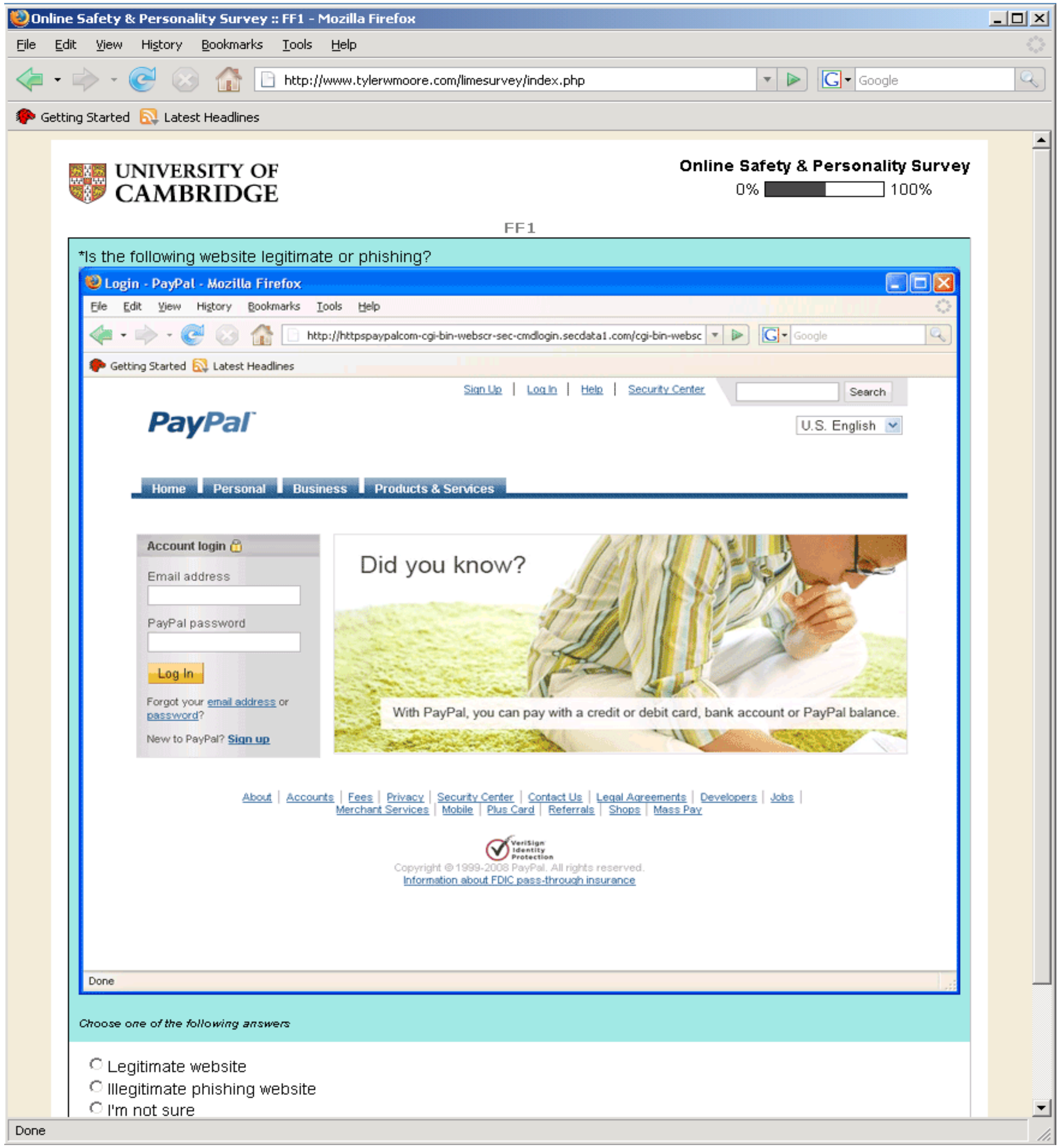
  `http://49320.0401/bankname/login/`

- Free web hosting

  `http://www.bank.com.freespacesitename.com/`

Fraud and Phishing Patterns

- Fraudsters do pretty well everything that normal marketers do
- The IT industry has abandoned manuals – people learn by doing, and marketers train them in unsafe behaviour (click on links…)
- Banks' approach is 'blame and train' – long known to not work in safety critical systems
- Their instructions 'look for the lock', 'click on images not URLs', 'parse the URL' are easily turned round, and discriminate against nongeeks


Online Safety & Personality Survey :: FF1 - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://www.tylerwmoore.com/limesurvey/index.php

Getting Started Latest Headlines

UNIVERSITY OF CAMBRIDGE

Online Safety & Personality Survey

0% 100%

FF1

*Is the following website legitimate or phishing?

Login - PayPal - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://httpspaypalcom-cgi-bin-webscr-sec-cmdlogin.secdata1.com/cgi-bin-websc

Getting Started Latest Headlines

Sign Up | Log In | Help | Security Center Search

PayPal

U.S. English

Home Personal Business Products & Services

Account login

Email address

PayPal password

Log In

Forgot your email address or password?

New to PayPal? Sign up

Did you know?

With PayPal, you can pay with a credit or debit card, bank account or PayPal balance.

About | Accounts | Fees | Privacy | Security Center | Contact Us | Legal Agreements | Developers | Jobs | Merchant Services | Mobile | Plus Card | Referrals | Shops | Mass Pay

VeriSign Identity Protection

Copyright © 1999-2008 PayPal. All rights reserved.
Information about FDIC pass-through insurance

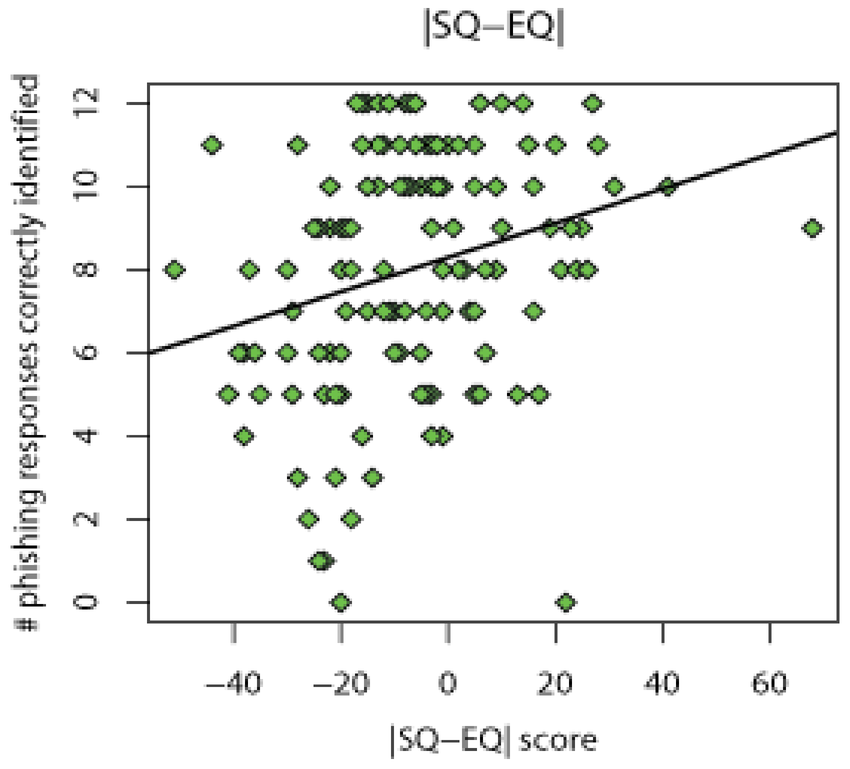
Done

Choose one of the following answers

- ◯ Legitimate website
- ◯ Illegitimate phishing website
- ◯ I'm not sure

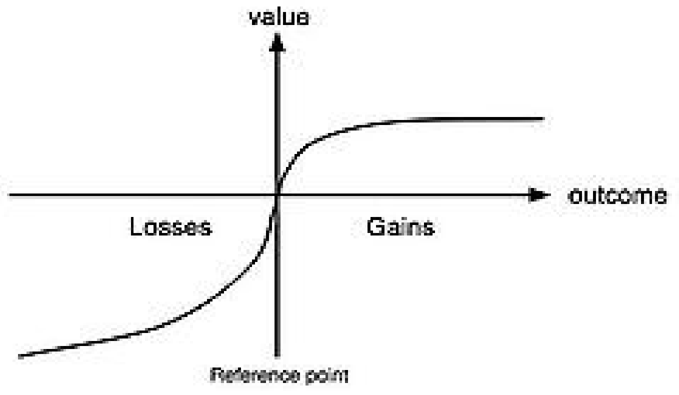
Done

Results



- Ability to detect phishing is correlated with SQ-EQ (S=systemizer, E=empathizer)
- It is (independently) correlated with gender
- So the gender HCI issue applies to security too

Prospect theory



- Kahneman & Tversky, 1970s: people dislike a loss more than they like a win (of same amount)
- Evolutionary logic of risk aversion, status quo bias
- Can drive fear marketing, ‘savings’, and (some of the) irrational behaviour of financial markets
- We’re oversensitive to low-probability large losses

Risk Misperception

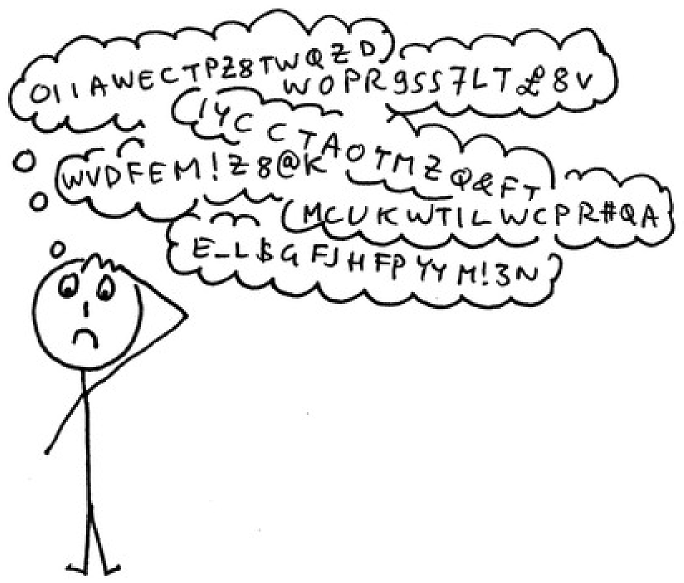
- Why do we overreact to terrorism?
  - Risk aversion / status quo bias
  - ‘Availability heuristic’ – easily-recalled data used to frame assessments
  - Our behaviour evolved in small social groups, and we react against the out-group
  - Mortality salience greatly amplifies this
  - We are also sensitive to agency, hostile intentions
- Anderson chapters 2, 24

Passwords

- Have many usability shortcomings
- Also have many security shortcomings
- But continue to be dominant

"There is no doubt that over time, people are going to rely less and less on passwords. People use the same password on different systems, they write them down and they just don't meet the challenge for anything you really want to secure.
--Bill Gates, 2004.

What's wrong with passwords



- Must resist guessing
- Must resist brute-force
- Must be all different
- Must be regularly changed
- Must be memorable (cannot be written down)
- Must be typed correctly, despite ***************

...but the intersection of these requirements is the empty set!

And the more accounts we have, the worse it gets

What else is wrong with passwords?

That was “just” usability. Then there's security.

- Thanks to the previous usability flaws...
  - weak passwords are guessed or brute-forced
  - (reused) passwords leaked at A cause a break-in at B
  - attacks on the recovery mechanism for forgotten passwords
  - passwords on post-its enable (casual or targeted) attacks
- Phishing
  - automated industrial-scale social engineering
- Keylogging
  - automated industrial-scale shoulder-surfing
  - or local, with hardware that can't be detected by security software
- Incompetent implementation at verifier end

  Bonneau, Preibusch: “The password thicket”, 2010.

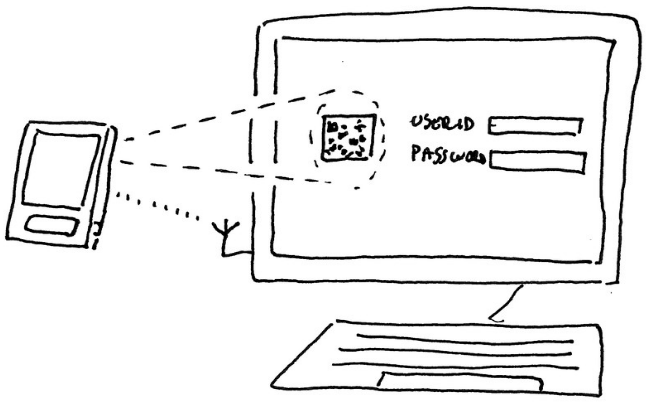
Will we ever get rid of passwords?

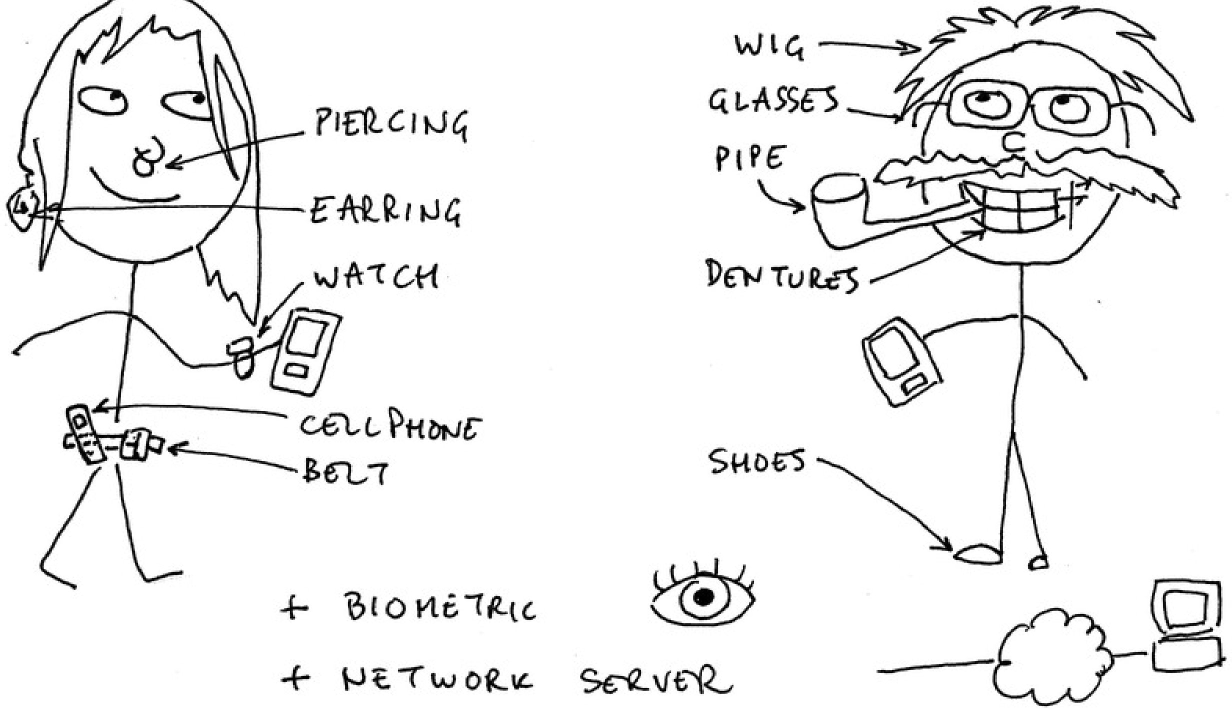
- Cheapest for implementers
  - No need to explain them to users
  - Cost to support one more user is negligible (Facebook reached 1 M users before external funding)
  - Can't be that bad: everyone else uses them too ("tragedy of the commons")
- Very many alternatives have been proposed
- "easier to remember" schemes: can they scale?
- Nice properties: Single Sign-On; password managers.

Unsustainable in the long run, but still...
expect password to be around for quite a bit more

Pico: no more passwords!

Stajano, 2011.





Two of you are building prototypes of this...

The compliance budget

Beautement, Sasse, Wonham, 2008.

- Users assess cost/benefits of security measures and put up with some inconvenience for the good of the company
  - With virus scanner on, program takes longer to build
  - Encrypting USB stick might prevent me from giving talk
- But only up to a point!
  - User patience is finite: “the compliance budget”
  - Once exhausted, user quickly stops cooperating

Must be managed like any other budget

Seven (?) principles for systems security

- Principles for sales, scams and other “persuasion” contexts
- Those principles predate computers! Rooted in human nature
- Thus, each also applies to computer *systems* security
  - Phishing: user is fixated on task completion
    (e.g. finding why new payee on PayPal account)
  - Advance fee frauds (419) take this to extreme lengths!
- You have to accept them, almost like the rules of physics
  - That's the way people work, no matter what you tell them to do
  - “If only it weren't for those gullible users” is arrogant and idiotic

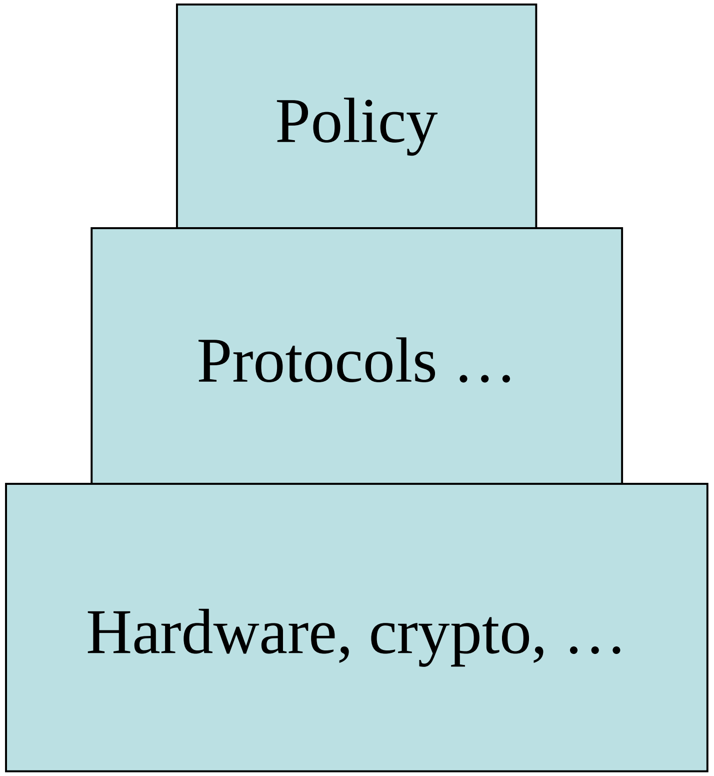
Wise system engineer understands and acknowledges the principles and makes the system robust by preventing their exploitation

Managing security

- Security awareness: measures must have, and be seen to have, full support of management
- Measuring security is hard
  - Measure security bugs, attack surface, attack cost...
- Risk analysis
  - Assets, vulnerabilities, threats, probabilities
  - That's quantitative, but inputs are usually guesswork
- Security policy: an instrument of communication

Design Hierarchy

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



Security vs Dependability

- Dependability = reliability + security
- Reliability and security are often strongly correlated in practice
- But malice is different from error!
  - Reliability: “Bob will be able to read this file”
  - Security: “The Chinese Government won’t be able to read this file”
- Proving a negative can be much harder …

Terminology

- A *subject* is a physical person
- A *person* can also be a legal person (firm)
- A *principal* can be
  - a person
  - equipment (PC, smartcard)
  - a role (the officer of the watch)
  - a complex role (Alice or Bob, Bob deputising for Alice)
- The level of precision is variable – sometimes you need to distinguish 'Bob's smartcard representing Bob who's standing in for Alice' from 'Bob using Alice's card in her absence'. Sometimes you don't.

Terminology

- *Secrecy*: mechanisms limiting the number of principals who can access information
- *Privacy*: control of your own secrets
- *Confidentiality*: an obligation to protect someone else's secrets

Thus your medical privacy is protected by your doctors' obligation of confidentiality

Terminology

- *Anonymity* is about restricting access to metadata. It has various flavours, from not being able to identify subjects to not being able to link their actions
- An object's *integrity* lies in its not having been altered since the last authorized modification
- *Authenticity* has two common meanings –
  - an object has integrity plus freshness
  - you're speaking to the right principal

Terminology

Trust is the hard one! It has 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

We'll use the NSA definition – number 2 above – by default.

E.g. an NSA man selling key material to the Chinese is trusted but not trustworthy (assuming his action unauthorised)

Terminology

- A *security policy* is a succinct statement of protection goals – typically less than a page of normal language
- A *protection profile* is a detailed statement of protection goals – typically dozens of pages of semi-formal language
- A *security target* is a detailed statement of protection goals applied to a particular system – and may be hundreds of pages of specification 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?

Policy: Multi Level Security

- Multilevel Secure (MLS) systems are widely used in government / intelligence / military contexts
- Basic idea: a clerk with ‘Secret’ clearance can read documents at ‘Confidential’ and ‘Secret’ but not at ‘Top Secret’
- 1960s/70s: problems with early mainframes
- First security policy to be worked out in detail following Anderson report (1973) for USAF which recommended keeping security policy and enforcement simple

Levels of Information

- Levels include:
  - **Top Secret:** compromise could cost many lives or do exceptionally grave damage to operations. E.g. intelligence sources and methods, battle plans
  - **Secret:** compromise could threaten life directly. E.g. weapon system performance, combat reports
  - **Confidential:** compromise could damage operations
  - **Restricted:** compromise might embarrass
- Resources have classifications
- Principals have clearances
- Information flows upwards only

Context of Multilevel Security

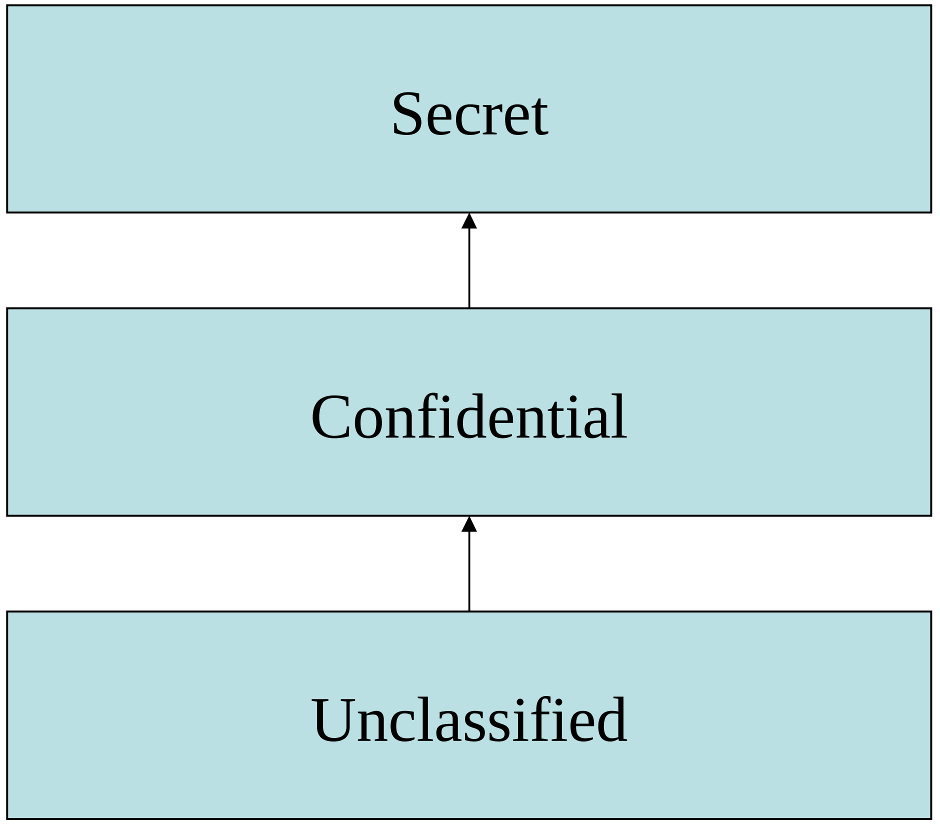
- Information mustn’t leak from High to Low
- Enforcement must be independent of user actions
- Perpetual problem: careless staff
- 1970s worry: operating system insecurity
- 1990s worry: virus at Low copies itself to High and starts signalling down (e.g. covert channel)

Context of Multilevel Security

Nagaraja, Anderson ‘The Snooping Dragon’, 2009.

- September 2008: Dalai Lama’s office realised there had been a security failure
- Initial break: targeted email with bad pdf
- Then: took over the mail server and spread it
- About 35 or their 50 PCs were infected
- Fix (Dharamsala): take ‘Secret’ stuff offline
- Fix (UKUSA agencies): use MLS mail guards and firewalls to prevent ‘Secret’ stuff getting out

Authorized Information Flow



Formalising the Policy

- Bell-LaPadula (1973):
  - *simple security policy*: no read up
  - **-policy*: no write down
- With these, one can prove that a system which starts in a secure state will remain in one
- Ideal: minimise the Trusted Computing Base (set of hardware, software and procedures that can break the security policy) so it's verifiable
- 1970s idea: use a reference monitor

Objections to BLP

- Some processes, such as memory management, need to read and write at all levels
- Fix: put them in the trusted computing base
- Consequence: once you put in all the stuff a real system needs (backup, recovery, comms…) the TCB is no longer small enough to be easily verifiable

Objections to BLP

- John MacLean's "System Z": as BLP but lets users request temporary declassification of any file
- Fix: add tranquility principles
  - Strong tranquility: labels never change
  - Weak tranquility: they don't change in such a way as to break the security policy
- Usual choice: weak tranquility using the "high watermark principle" – a process acquires the highest label of any resource it's touched
- Problem: have to rewrite apps (e.g. license server)

Covert Channels

- In 1973 Butler Lampson warned BLP might be impractical because of covert channels: “neither designed not intended to carry information at all”
- A Trojan at High signals to a buddy at Low by modulating a shared system resource
  - Fills the disk (storage channel)
  - Loads the CPU (timing channel)
- Capacity depends on bandwidth and S/N. So: cut the bandwidth or increase the noise
- But it’s really hard to get below 1 bit/s or so…

Objections to BLP

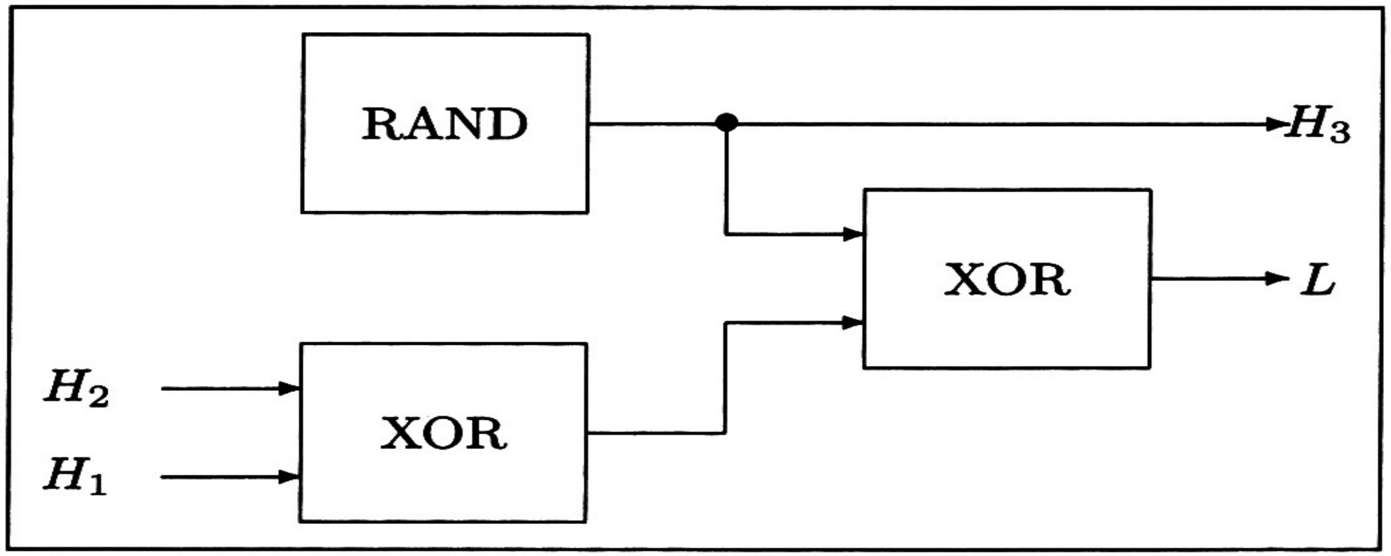
- High can’t acknowledge receipt from Low
- This blind write-up is often inconvenient: information vanishes into a black hole
- Option 1: accept this and engineer for it (Morris theory) – CIA usenet feed
- Option 2: allow acks, but be aware that they might be used by High to signal to Low
- Use some combination of software trust and covert channel elimination (more later …)

Variants on Bell-LaPadula

- Noninterference: no input by High can affect what Low can see. So whatever trace there is for High input X, there's a trace with High input Ø that looks the same to Low (Goguen and Messeguer 1982)
- Nondeducibility: weakens this so that Low is allowed to see High data, just not to understand it – e.g. a LAN where Low can see encrypted High packets going past (Sutherland 1986)

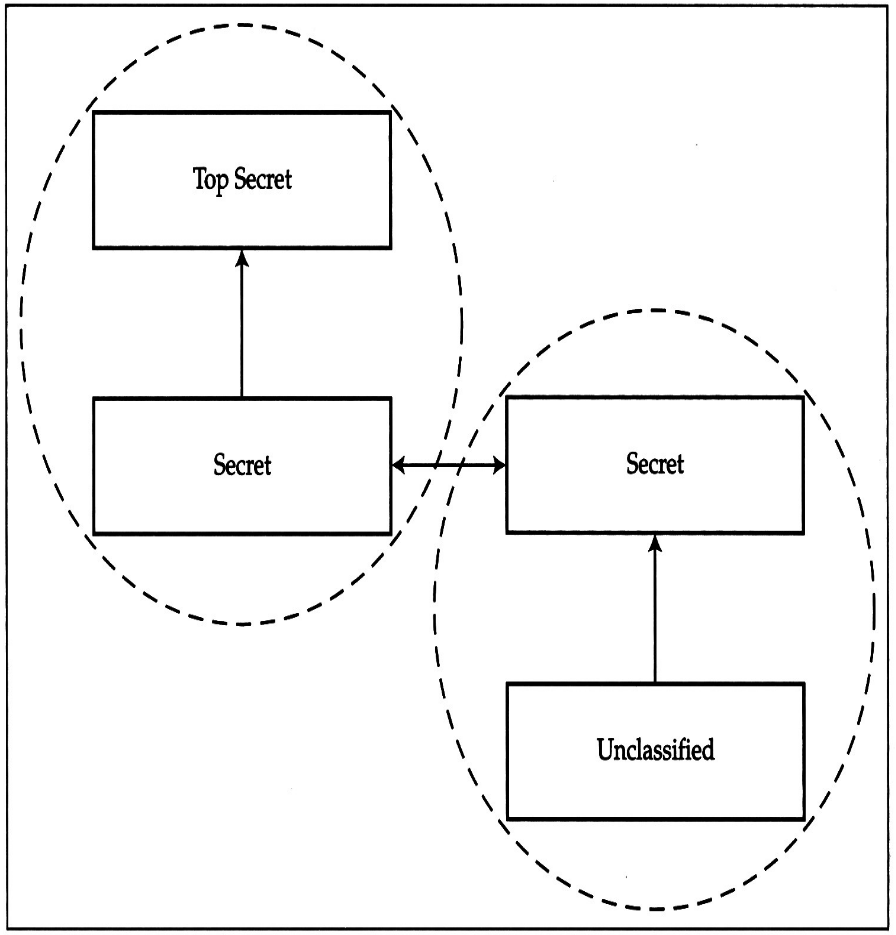
Composability

- Systems can become insecure when interconnected, or when feedback is added
- Nondeducibility and noninterference also fail to compose



Covert channels and Cascade Problem

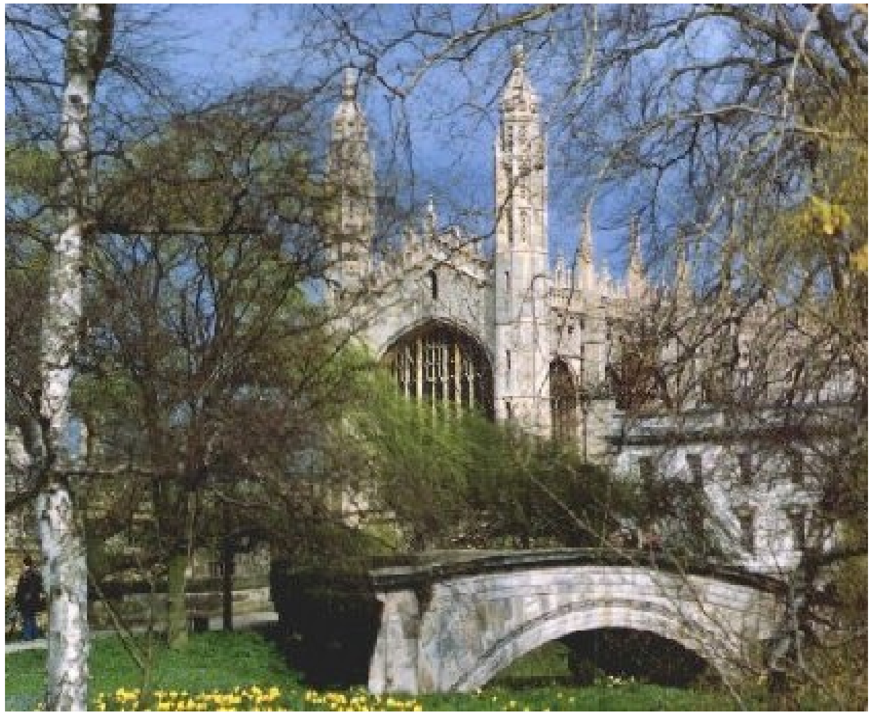
- For fear of *covert channels*: a single MLS system is not allowed to span all levels
- Still, two systems could be connected in a way that will break this policy
- Covert channel: not designed for comms, but allows leakage from High to Low (e.g. filling the disk or affecting some other shared resource)

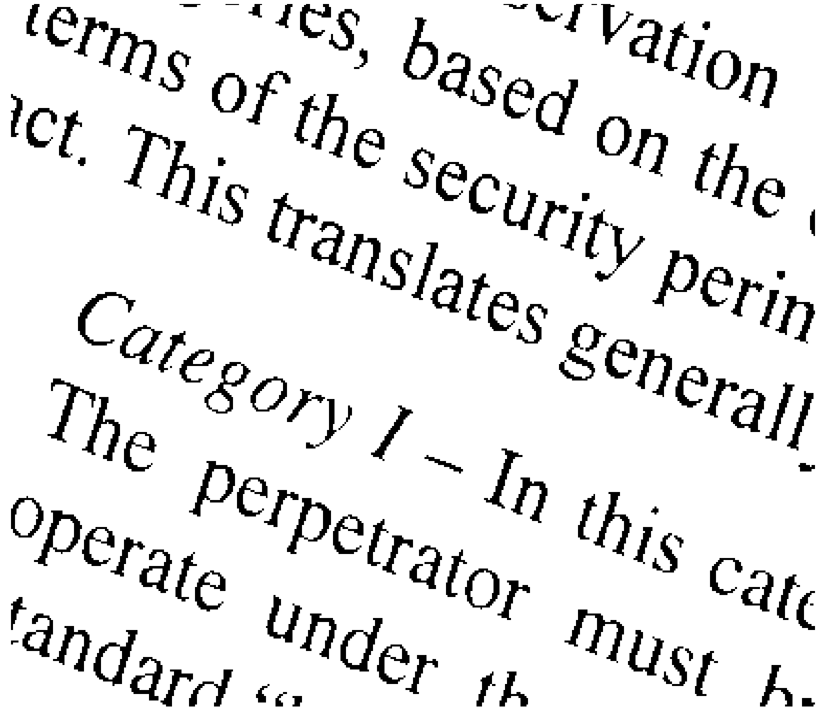


Downgrading

- A related problem to the covert channel is how to downgrade information
- Analysts routinely produce Secret briefings based on Top Secret intelligence, by manual paraphrasis
- Also, some objects are downgraded as a matter of deliberate policy – an act by a trusted subject
- For example, a Top Secret satellite image is to be declassified and released to the press

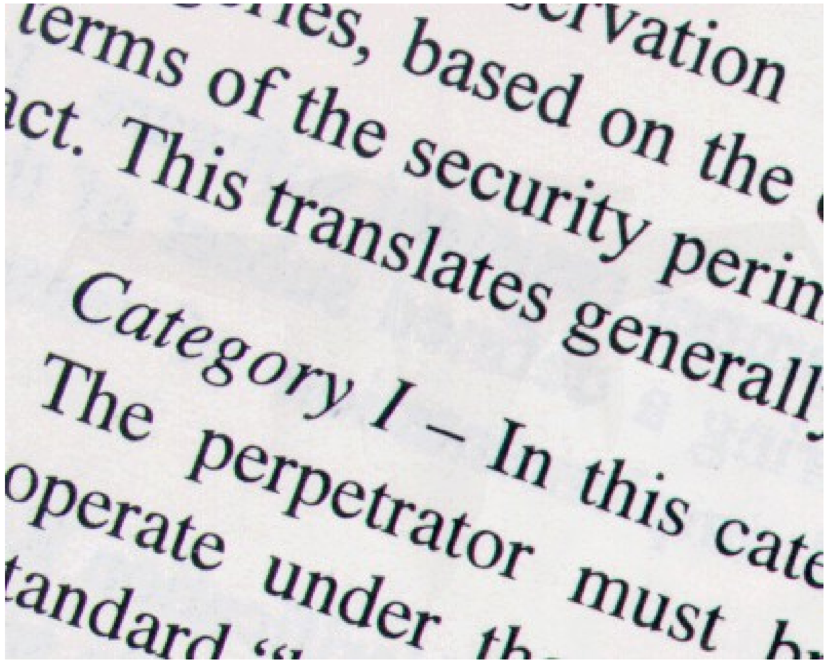
Downgrading



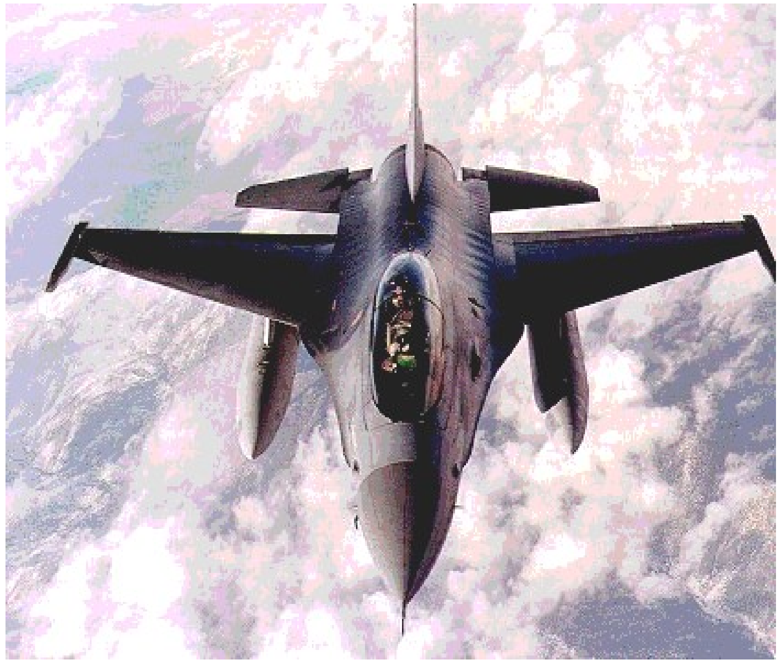

ervation
ies, based on the
terms of the security perin
ct. This translates generall
Category I – In this cat
The perpetrator must b
operate under th
tandard “

Text hidden in least significant bits of image

Downgrading



ervation
es, based on the
terms of the security perin
ct. This translates generall
Category I – In this cate
The perpetrator must b
operate under th
tandard “



Picture hidden in three least significant bits of text

Many examples of MLS Systems

- MULTICS (1965). Became template for Orange Book “trusted systems”.
- SCOMP. Evolved from Multics. Had formally verified hw & sw.
- Blacker. Prevent leakage from “red” (cleartext) to “black” (ciphertext).
- Compartmented Mode Workstations (CMWs) – for analysts who downgrade Top Secret material. Allow cut-and-paste from L $\rightarrow$ H, L $\rightarrow$ L and H $\rightarrow$ H but not H $\rightarrow$ L
- The NRL Pump: copy from Low to High with minimal covert channel leakage
- RAF Logistics IT System
- DERA’s ‘Purple Penelope’ (later ‘Sybard Suite’)

Many more details in Anderson ch 8

Multilevel Integrity

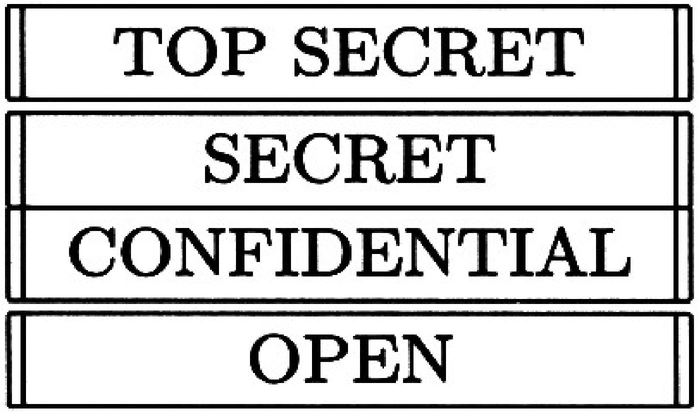
- The Biba model – data may flow only down from high-integrity to low-integrity
- Dual of BLP:
  - **Simple integrity property:** subject may alter object only at same or lower level
  - ***-integrity property:** subject that can observe X is allowed to alter objects dominated by X
- So you have low watermark properties, etc
- Example: medical equipment with two levels, "calibrate" and "operate"

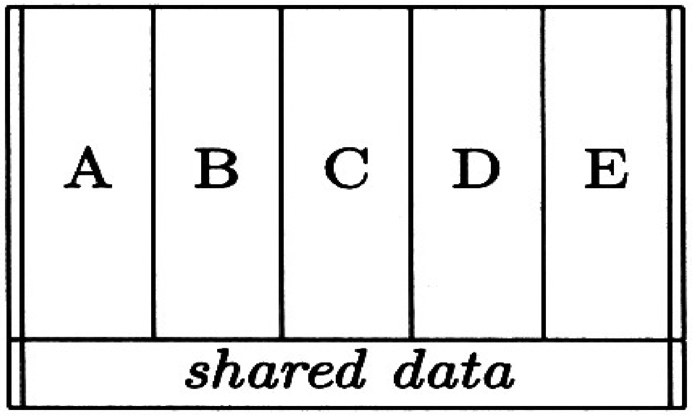
Multilevel Integrity

- LOMAC was an experimental Linux system with system files at High, network at Low
- A program that read traffic was downgraded
- Vista adopted this – marks objects Low, Medium, High or System, and has default policy of NoWriteUp
- Critical stuff is System, most other stuff is Medium, IE is Low
- Could in theory provide good protection – in practice, User Account Control trains people to override it!
  Too many false alarms make any safeguard useless

Multilateral Security

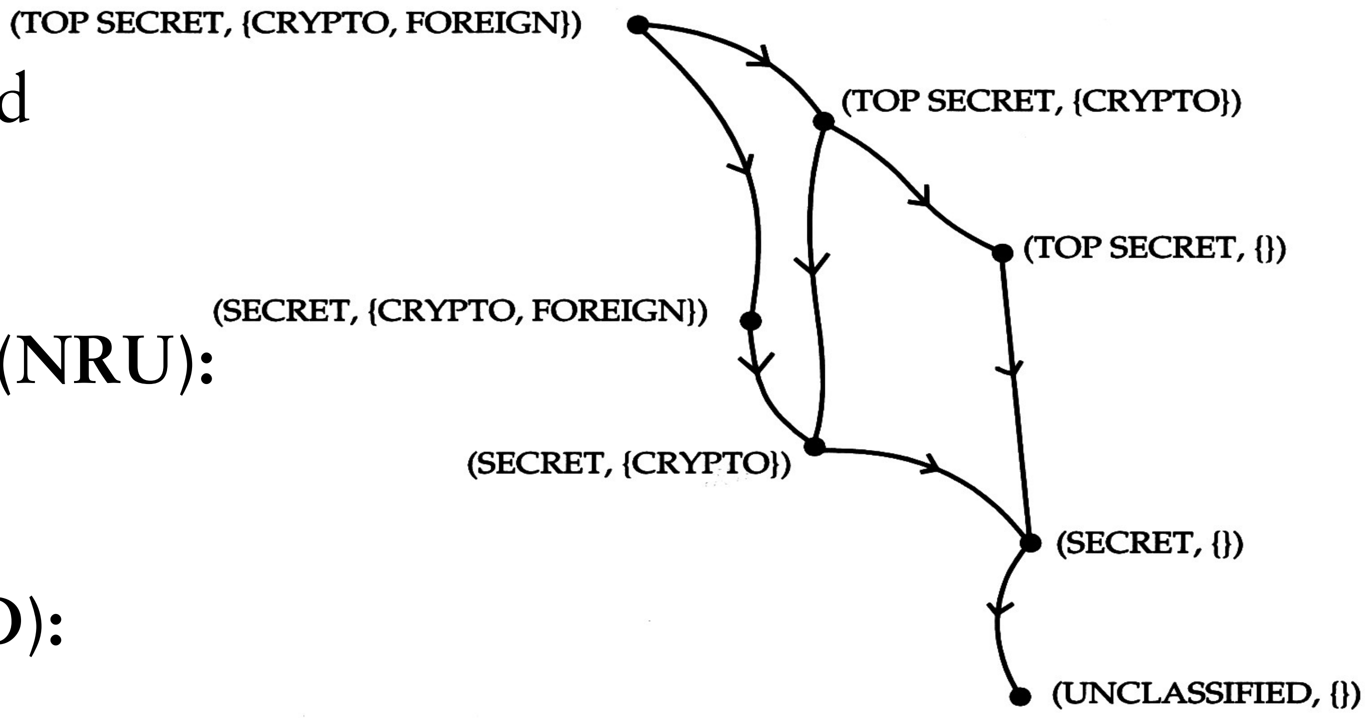
- Sometimes the aim is to stop data flowing down
- But other times, you want to stop lateral flows
- Examples:
  - Intelligence
  - Competing clients of an accounting firm
  - Medical records by practice or hospital





The Lattice Model

- Intelligence agencies manage ‘compartmented’ data by adding categories. Label = (level, {set of categories})
- Basic idea: BLP requires only a partial order (*dominates*).
- X dominates Y iff level(X) $\geq$ level(Y) and cat(X) $\supseteq$ cat(Y)



- **BLP simple property (NRU):** X can read Y iff X dominates Y
- **BLP *property (NWD):** X can write Y iff X is dominated by Y

The Lattice Model

- The labels "Secret {Crypto}" and "Top Secret {}" are incomparable – no data flow either way
- Categories can have special handling rules – e.g. wartime "Ultra" (now "Umbra") meant you could not risk capture
- Problem: either you end up with millions of compartments, or you end up with all the good stuff in one pot (the CIA's problem with Ames)
- Post-9/11, the big-pot model is gaining ground …

The Chinese Wall Model

- Industries such as investment banking, advertising and accounting have too few top firms for each big client to have its own
- So maybe you're auditing BP, and Shell too!
- Add a "Chinese Wall" that stops the two teams communicating
- *A subject can only access information that does not conflict with information they have already accessed*
- Idea: use a refinement of Bell-LaPadula

The Chinese Wall Model

- Subtlety: it's not enough to stop a Shell analyst reading BP data; must also stop a BP analyst writing data to a Barclays file that the Shell analyst can also read.
- For each object O, let y(O) be the firm it relates to.
  O = bp-payroll.xls
  y(O) = BP
- Let x(O) be that firm's conflict-of-interest class.
  x(O) = {set of all oil companies}
- Let x(O) = x_0 (a special COI class) if the information has been sanitized, so anyone is allowed to see it

The Chinese Wall Model

(Brewer and Nash, 1989)

Reminder: x(bp-file) = all oil companies; y(bp-file) = BP.

- **Simple property:** access (for reading or writing) is granted only if the object belongs to a firm the subject has already accessed, or to a firm in a different conflict-of-interest class

  S can access O only if, for all O' that S has accessed, either $y(O)=y(O')$ *or* $y(O) \notin x(O')$

  Boolean matrix holds state: $N_{S,O}$ = "did S ever access O?"

- ***-property:** Writing is only allowed if simple property satisfied and the subject cannot read any unsanitized object from other firms

  S can write O only if (simple property grants access) and (S is not able to read any O' with $y(O)\neq y(O')$ *and* $x(O')\neq x_0$*)*

  Flow of unsanitized info is kept to within the firm's dataset.

Bookkeeping, c. 3300 BC

Bookkeeping c. 1100 AD

- How do you manage a business that's become too large to staff with your own family members?
- Double-entry bookkeeping – each entry in one ledger is matched by opposite entry in another
  - E.g. firm sells £100 of goods on credit – credit the sales account, debit the receivables account
  - Customer pays – credit the receivables account, debit the cash account

  (Some of these may sound backwards but make sense to accountants)
- So bookkeepers have to collude to commit fraud

Banking Security Policy

- Threat model:
  - 1% of staff go bad each year
  - Mistakes happen – 1 in 500 paper transactions
  - There are clever fraudsters too
  - Loss of confidence means ruin
- Protection goals:
  - Deter/prevent the obvious frauds
  - Detect the rest as soon as possible
  - Be able to defend the bank's actions in court

The Clark-Wilson Policy Model

Work by David Clark (MIT) and David Wilson (Ernst & Whinney) in 1986 to model double-entry bookkeeping

- In addition to the normal objects in your system, which we call unconstrained data items (UDIs), you add constrained data items (CDIs)
- CDIs are acted on by special programs called transformation procedures (TPs) that preserve the invariants
- IVPs (integrity verification procedures) verify the validity of CDIs (eg that the books balance)
- Mental model: a TP in a bank must increase the balance in one CDI (account) by the same amount that it decrements another

Clark-Wilson rules

1. There's an IVP to validate integrity of each CDI
2. Applying a TP to a CDI maintains integrity
3. A CDI can only be changed by a TP
4. Subjects can use only certain TPs on certain CDIs
5. Triples (subject, TP, CDI) enforce separation of duty
6. Special TPs on UDIs can produce CDIs
7. Each TP application must be logged to special append-only CDI
8. System must authenticate subjects that attempt to launch a TP
9. Only special subjects (admins) can change auth lists

Clark-Wilson importance

- First influential security policy model not based on BLP
- Application-level security state
  - The audit log (with enough info to reconstruct each TP)
  - The triples
- Separation of duties
  - In parallel (require 2 signatures, e.g. for large and irreversible transactions)
  - In series (different people for raising an order, accepting delivery, paying invoice, balancing budget)

Internal Control: Theory

- Employees optimise their own utility, not their employers' (the 'agency problem')
- Internal controls should mitigate not just fraud but nepotism, empire-building, …
- Corporate governance rules like Sarbanes-Oxley (USA), Cadbury (UK) set the tone
- The big accountants drive 'good practice'
- People talk of 'risk management' but the process is basically evolutionary

Internal Control: Practice

- Mustn’t just audit the finances – McKesson and Robbins collapse, 1938, had fictitious trading partners and a bogus Montreal bank
- Enforcement often cyclical; firms centralise then decentralise, ease up then crack down
- Systematic analysis: as in software engineering 1b, can trace worst outcomes back along workflow, or look for greatest opportunities for individual staff (ask them!)
- Strategy: deter – detect – alarm – delay – response
- And with social malware now appearing – how do you cope with a rootkit on 35 or the 50 machines in finance?

Ubiquitous computing

- Authentication and device pairing without infrastructure
  - Two devices meet for the first time
  - No online servers available
    - Can't do the key distribution protocols studied later in this course
    - Can't use PKI, because you can't check for revoked keys
  - One device wants the other to “do something”
  - Authentication as temporary master-slave pairing:
    Secure Transient Association
    e.g. Smart Home devices + Universal Controller (phone)

Big Stick policy

(Stajano, 2000)

Whoever has physical control of the device
is allowed to take it over

- E.g. you can press the hard reset button on your router and the admin password is restored to factory default
- A trivial policy; but effective because cynically realistic
- Works fine for your lawnmower, pocket calculator (stateless) or for your fridge or router (inside the home)
- But not good for devices with valuable state that may be left unattended (e.g. a vending machine, a wireless sensor)

Imprinting

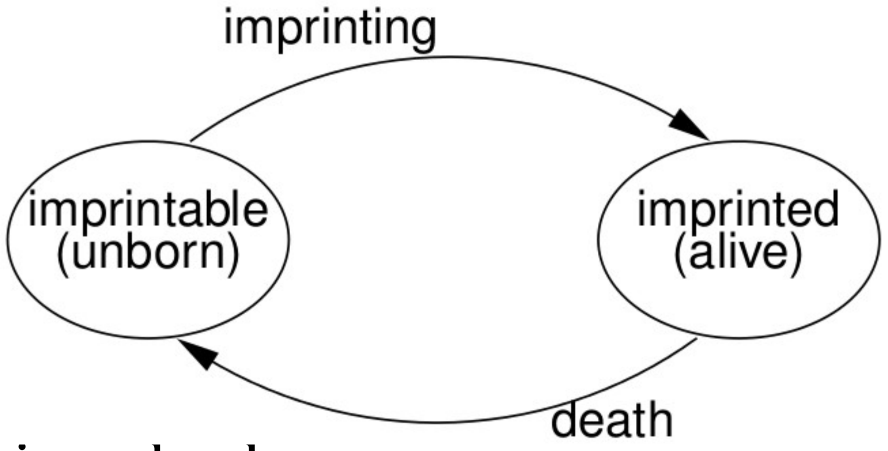
- Inspired by Konrad Lorenz (1973 Nobel prize)
- First moving subject seen by duckling becomes its mother
- Duckling stays faithful to mother until death

Out of metaphor:

- Slave device starts as imprintable
- First device that gives it a key becomes its master
- Bootstrap with unmediated physical channel
  ...but what if you then want to sell your Blu-Ray player?

Resurrecting Duckling policy

Stajano, Anderson 1999



- **Two-state:** *duckling* can be imprintable or imprinted
- **Imprinting:** transition to imprinted when someone (henceforth *mother duck*) sends imprinting key over secure channel
- **Death:** transition to imprintable when mother duck orders it (like seppuku of the samurai)
- **Assassination:** uneconomical for attacker artificially to cause transition to imprintable (implies tamper resistance)

Most work on bootstrapping security associations refers to this

Medical Record Systems

- Perceived problem: many incompatible systems in hospital depts / GP surgeries, leading to higher costs, clunky procedures (e.g. pediatrics, geriatrics) and difficulty of collecting data for management
- Proposed solution (1992–95): integrated hospital system where everyone could see everything
- 1995 roll-out in health minister’s constituency
- Nurse in Basingstoke found that her ward system let her (and colleagues) see results of tests she’d had at her GP!

Medical Record Systems

- NHS ‘Information Management and Technology Strategy’ adopted an MLS policy

AIDS	Highly sensitive	No network
Medical records	Sensitive	Password generator
Admin (inc. drugs)	Non-sensitive	Dialback
Libraries	Open	Password

- Problem 1: where is a prescription for ARV (HIV treatment)?
- Problem 2: all GP receptionists see all UK records!

Medical Record Systems

- What should the security policy be?
- Early attempt at policy for a ‘lifetime electronic medical record’ gave up after 60+ rules
- BMA project 1995–6 (Anderson)
- Key simplifying assumption: define the ‘record’ as the maximum set of health information with a single access control list
- Reflects actual practice! Your lifetime GP record has pointers to further stuff in hospitals etc via referral letters and discharge summaries

Methodology

- First, get the threat model right. Will the attackers be insiders or outsiders? Capable or opportunistic? Institutions or individuals?
- Next, set protection priorities and get agreement from application experts
- Then build a policy model
- Then validate it by peer review, public consultation, pilot projects etc

Threat Model

- ‘Pretexting’ is the standard way for private eyes to get data
- 1996 pilot – staff at N Yorks Health Authority trained to log information requests, get them signed off, and call back to a number you can check independently
- Anderson's team detected 30 false-pretext calls per week!

The BMA Policy

Anderson, 1996

1. Each record has a single access control list
2. Record opening: ACL = clinician + patient + referrer (if any)
3. Ownership: one clinician must control the ACL
4. Notification: controller must notify patient of changes to the ACL
5. Persistence: no-one may delete data until statutory period has expired
6. Attribution: all accesses (even reads) logged with name, date and time
7. Information flow: Information derived from record A may be appended to record B iff $ACL(B) \subseteq ACL(A)$
8. Aggregation: controls must exist, and patients must be notified if anyone on the ACL has access to many other patients' data
9. TCB: the mechanisms that enforce this policy must be evaluated by independent experts

The policy was put to consultation, and tested in general practice
Everything worked except auditing read access

Inference Control

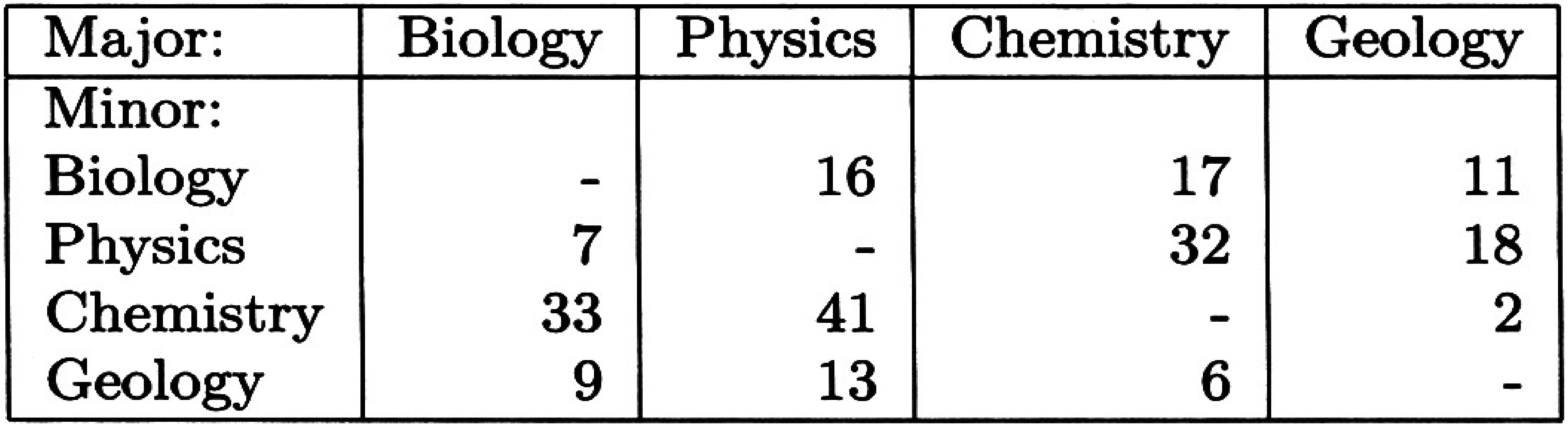
- Secondary uses (eg using medical data for research) were the hard problem. “Anonymized” data usually isn't.
- Inference control is also known as “statistical security” or “statistical disclosure control”
- Previously, only totals and samples were published, e.g. population and income per electoral ward, plus one record out of 1000 with identifiers removed manually
- Move to online database system changed the game
- Dorothy Denning bet her boss at the US census that she could work out his salary – and won!

Inference Control

- Query set size controls are very common. E.g. New Zealand medical records query must be answered from at least six records
- Problem: tracker attacks. Find a set of queries that reveal the target. E.g if there is only one female full professor:
  - “Average salary professors”
  - “Average salary male professors”
- Or even these figures for all “non-professors”!
- On reasonable assumptions, trackers exist for almost all sensitive statistics

Inference Control

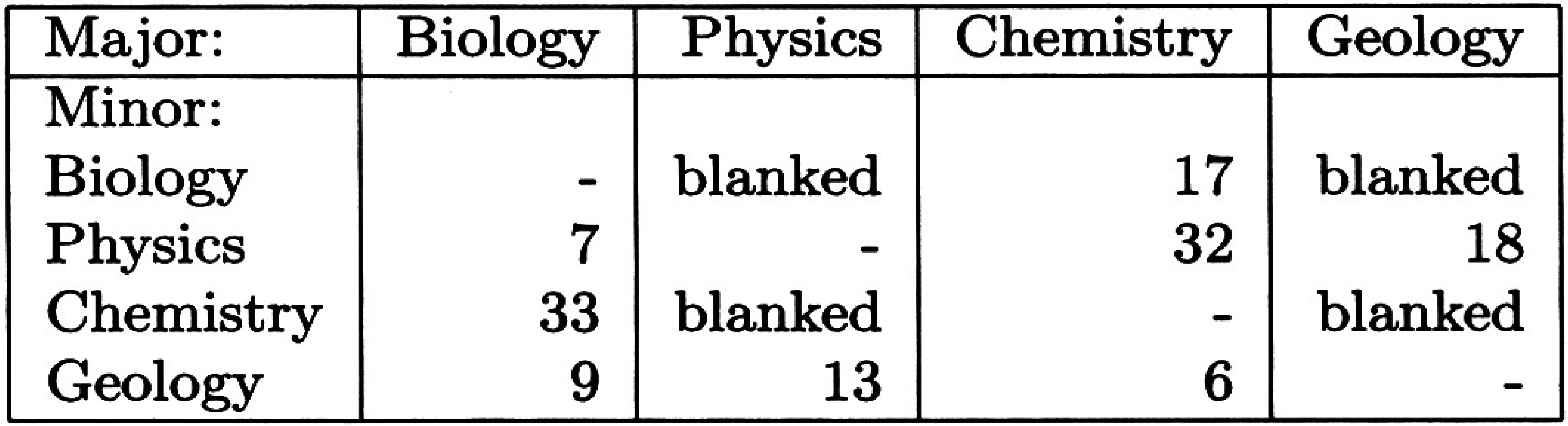
- Cell suppression: e.g. suppose we can’t reveal exam results for two or fewer students (else anyone in a 2-set can work out the other's mark).
- Then we must hide Geology-with-chemistry.
- But could be worked out from the Chemistry-minor average, so we must also blank one of {Biology-with-chem, Physics-with-chem}
- Same when considering the Geology-major average



Major:	Biology	Physics	Chemistry	Geology
Minor:				
Biology	-	16	17	11
Physics	7	-	32	18
Chemistry	33	41	-	2
Geology	9	13	6	-

Inference Control

- The more attributes in the scheme, the more data is lost
- With n-dimensional data, complementary cell suppression costs 2^n cells for each primary suppression
- Database can quickly become unusable



Major:	Biology	Physics	Chemistry	Geology
Minor:				
Biology	-	blanked	17	blanked
Physics	7	-	32	18
Chemistry	33	blanked	-	blanked
Geology	9	13	6	-

Inference Control

More evidence that anonymizing is hard: trends in drug prescribing

- How many prescriptions from each doctor in each week?
  - Remove patient names
  - Remove doctor names too: no pestering from drug salesmen please

	Week 1	Week 2	Week 3	Week 4
Doctor 1	17	21	15	19
Doctor 2	20	14	3	25
Doctor 3	18	17	26	17

But Dr Jones went skiing in 3rd week…

And her partner Dr Smith probably covered for her…

Inference Control

- Perturbation – add random noise (e.g. to mask small values)
- Trimming – to remove outliers (else ‘average earnings Swaffham Bulbeck’ might leak Michael Marshall’s income)
- Random sampling – answer each query with respect to a subset of records, maybe chosen by hashing the query with a secret key

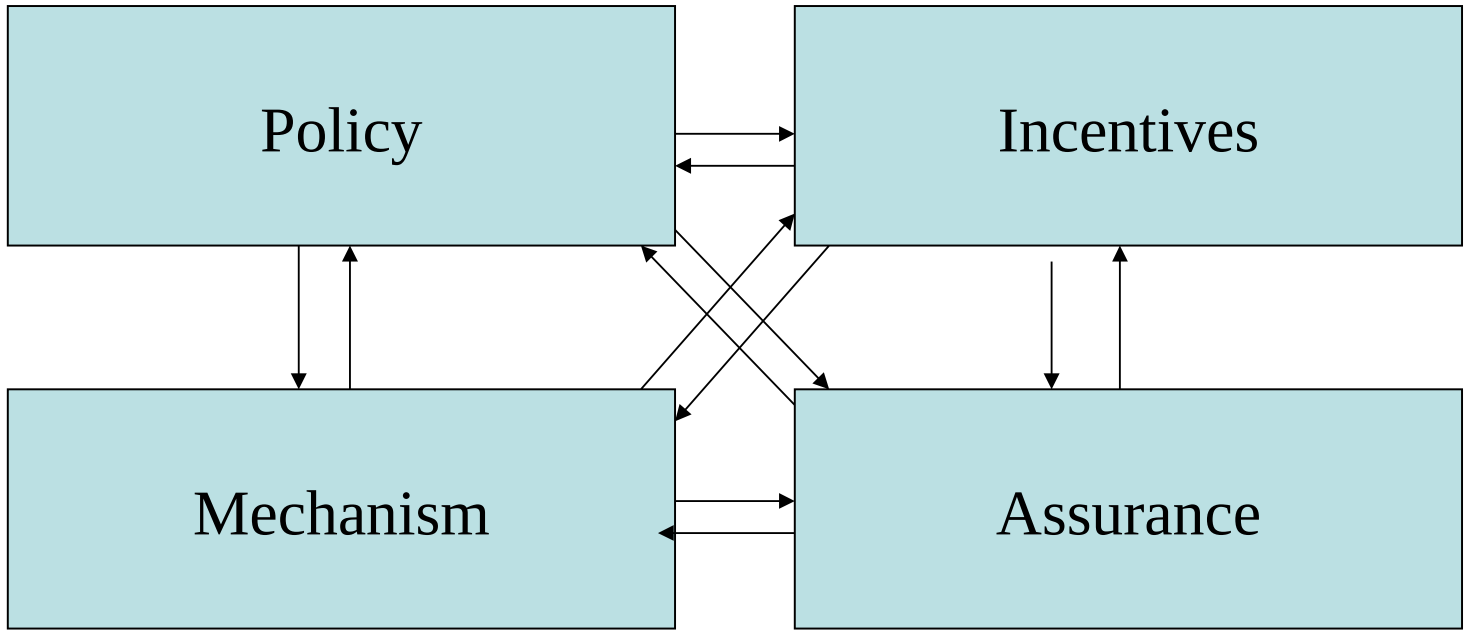
Inference Control

- Big problem in medical databases: context
- ‘Show me all 34-yo women with 9-yo daughters where both have psoriasis’
- If you link episodes into longitudinal records, most patients can be re-identified
- Add demographic, family data: worse still
- Active attacks: worse still
- Social-network stuff such as friends, or disease contacts: worse still
- Only way to stay legal: consent (offer an opt-out)

Policy

- Bell-LaPadula, Biba and Clark-Wilson are only three early examples of policy
- Many industries develop their own, and may get Protection Profiles evaluated
- Many things go wrong – people protect the things they can, not the things they should
- We often see deception at the policy level!
- For now, here’s a useful framework

A Framework



Note how tinkering with mechanisms often feels easier than fixing incentives …

Availability Policies

- Until recently, researchers ignored availability, but it's where the money goes

	research	industry
confidentiality	90%	1%
integrity/authenticity	9%	9%
availability	1%	90%

- Good example: alarms!

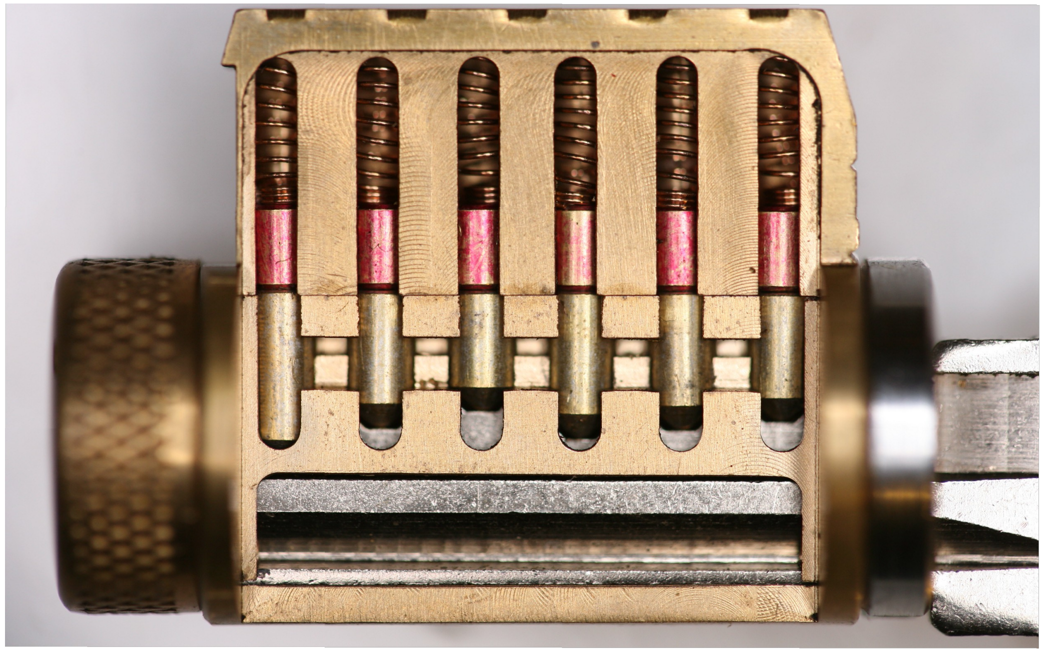
The Physical Security Revolution

- IT security is rooted in physical security (for server rooms, crypto boxes etc)
- Old model: firms like Chubb with proprietary fire / burglar alarm systems; locks with master-keying systems
- New model: sensors, striker plates, sensors run off ethernet like everything else
- We should be able to do better than metal systems
- Should be much easier to manage too – but many tensions (manageability, dependability)

Basic lockpicking

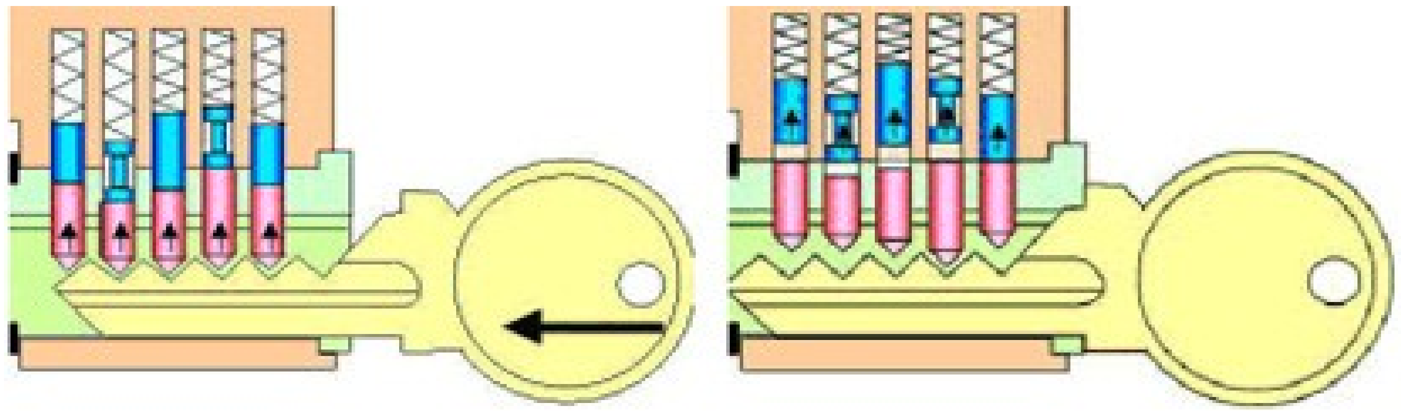
- When all pins align at the shear line, cylinder can rotate and lock opens.

- While applying torque, pick one pin at a time, in order of stickiness
- Cost is linear in number of pins (not exponential)
- Requires a modest but non-zero amount of skill



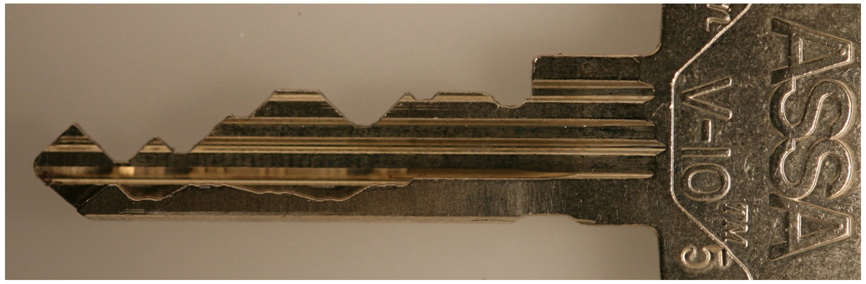
Lock Bumping

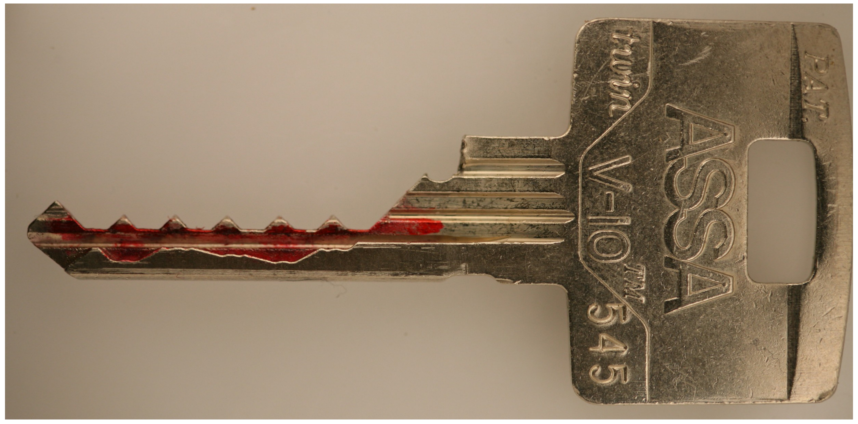
With lock bumping, that's not necessary



- Cut a key down to the (0,0,0,0,0) bitting
- Put it in the keyway, apply torque, and tap
- Pins bounce up to shear line and cylinder turns

Lock Bumping





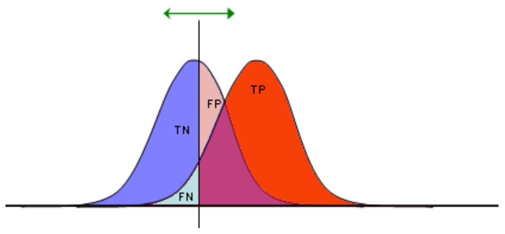
- With fancy locks, like sidebar here: break that separately first
- pick it, or steal or photograph a key
- Enthusiasts have now defeated most mechanical locks

Burglar Alarms

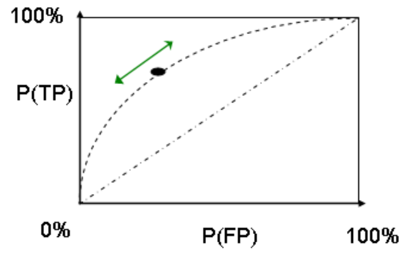
- Good example of a service where availability matters!
  - If there's a burglar in my vault I want the alarm company to be told!
  - Not bothered about confidentiality: you can tell other people too
  - Nor about authenticity: I don't care who tells them
- Wide range of systems: homes – supermarkets – jewellery stores – banks – nuclear facilities
- Wide range of standards, from Underwriters' Labs to the IAEA

How to Steal a Painting (1)

- Hollywood idea of art theft: cut through roof, climb down rope, grab painting without stepping on pressure mat (i.e. sensor defeat), get girl…
  - Response to this perceived threat is more, fancier sensors
  - There are limits: set by false alarm rates and environmental conditions
  - Critical science: the Receiver Operating Characteristic (ROC) curve
  - Multisensor data fusion is really hard!
- But most high-grade attacks don't defeat sensors



TP	FP
FN	TN
1	1



How to Steal a Painting (2)

- More common type of art theft: hide in broom cupboard, come out at midnight, grab the Rembrandt and head for the fire exit
  - Understand the service you're supplying: deter – detect – alarm – delay – response
  - Don't rely on tech too much: 'Titanic effect'
- Or just toss in a smoke grenade. The fire alarm turns off the burglar alarm. Dash in and grab the Rembrandt
  - If caught, claim you were passing by and dashed in to save the national heritage

How to Steal a Painting (3)

- Wait for a dark and stormy night, when false alarms will be common. Create several (fence rattling). Wait till guards stop responding
  - Typical police force blacklists a property after 3–4 false alarms
  - Fix: multiple sensors, e.g. CCTV inside
  - Problem: we want best sensors on the outside for delay, but on the inside for low false alarm rate
- This is the standard way for professionals to do a bank vault!

How to Steal a Painting (4)

- Cut the wire from the sensor to the controller
- Connect a bogus controller to the phone line
- Cut the communications to the controller
- Cut the communications to many controllers
  - E.g. Holborn explosion
  - LPC: 2 independent channels for risks over £20m
  - Armed response force on premises for plutonium
- Insurance companies would like resilient anonymous communications to make service denial hard

Alarms – Lessons Learned

- Dealing with service denial is becoming more important, and harder
- Trade–off between false alarm rate and missed alarm rate is central
- You need to be clear what the service you're supplying (or buying) is – is it about sounding the alarm, or more?
- Critically, we need to design the system around the limitations of the human response. E.g. in airport screening, you insert deliberate false alarms. But what more can be said?

Bonus slides
(not examinable)

Broadcast stream authentication

- **Problem.** One sender broadcasts packets to many unknown receivers. Receivers want source authentication before they believe the packets.
- **Solution.** Sender signs each packet. Each receiver verifies each packet. Problem solved. Go home.
  - Except you can't do that in low-cost ubiquitous computing contexts. Public key crypto costs too much time and energy. Battery won't last.
- **Solution:** Amortize one signature over thousands of packets.
  - But: high latency; high vulnerability to packet loss and to DoS (verify bogus sigs).
- **Solution.** Sender attaches a MAC to each packet. Each receiver verifies it.
  - Even worse because too many people need the MAC key (to verify) and anyone with the MAC key could send forged packets.

How can we get asymmetric properties with symmetric crypto?

(failed) **attempts at hash chains**

Series of messages M_0, M_1, M_2,… inside packets P_0, P_1, P_2…

$P_i = M_i, h(M_i)$ *validated by self (no chaining)*

Hopeless: forger trivially substitutes $P^*_i = M^*_i, h(M^*_i)$

$P_i = M_i, h(P_{i+1})$ *validated by previous*

Nice if you could, but needs infinite knowledge of future

$P_i = M_i, h(M_{i+1})$ *validated by previous*

Needs knowledge of next message

Forger intercepts P_{i+1} and substitutes $P^*_{i+1} = M_{i+1}, h(M^*_{i+2})$

$P_i = M_i, h(P_{i-1})$ *validated by next*

Hopeless: forger trivially substitutes $P^*_i = M^*_i, h(P_{i-1})$

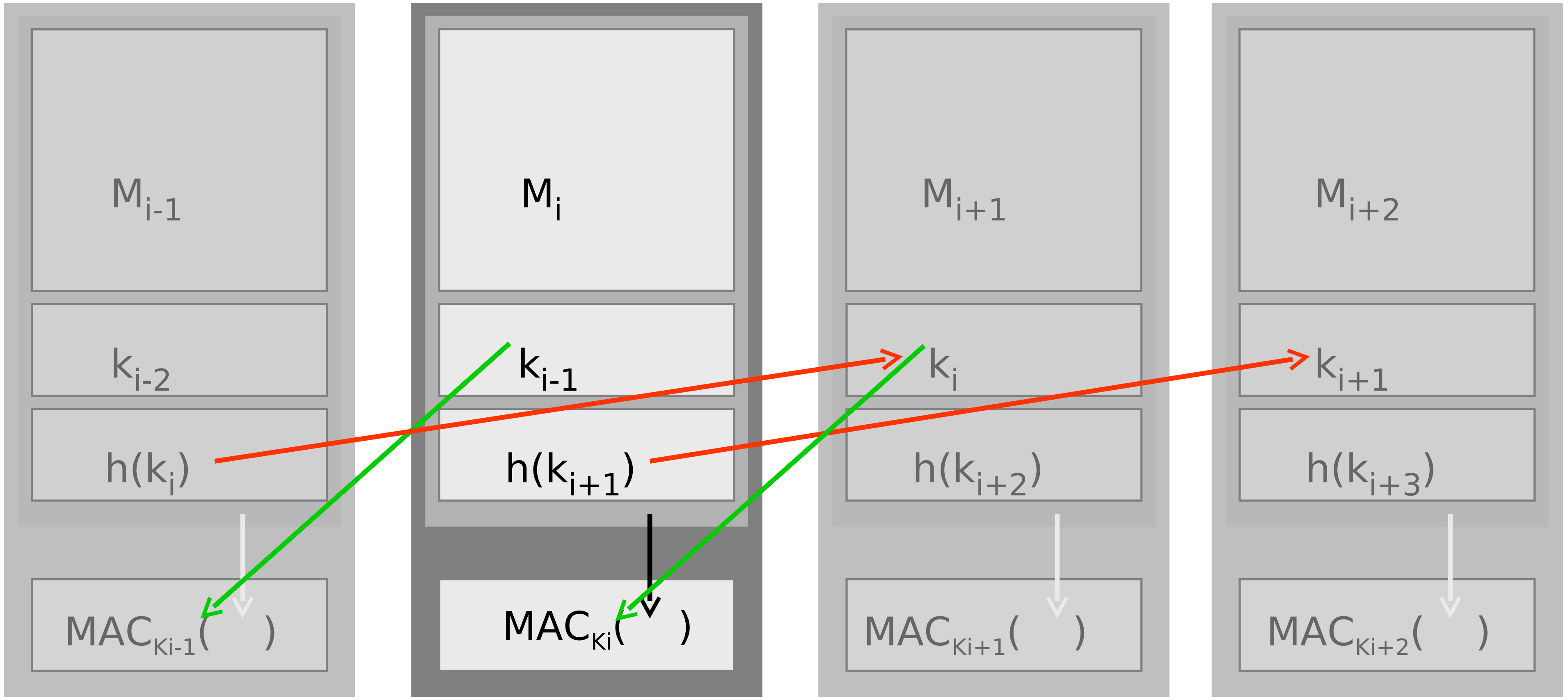
Guy Fawkes protocol (Anderson et al. 1998)

- Add a level of indirection (standard solution to most computer science problems): packet key k_i
- Not hash, but MAC using packet key
- P_i contains its own MAC value
- But k_i revealed later
  - So far this is like "validated by next"
- But (clever part) k_i validated by previous!
- $P_i = M_i, k_{i-1}, h(k_{i+1}), MAC_{ki}(M_i, k_{i-1}, h(k_{i+1}))$

You can do this without having to know the next message

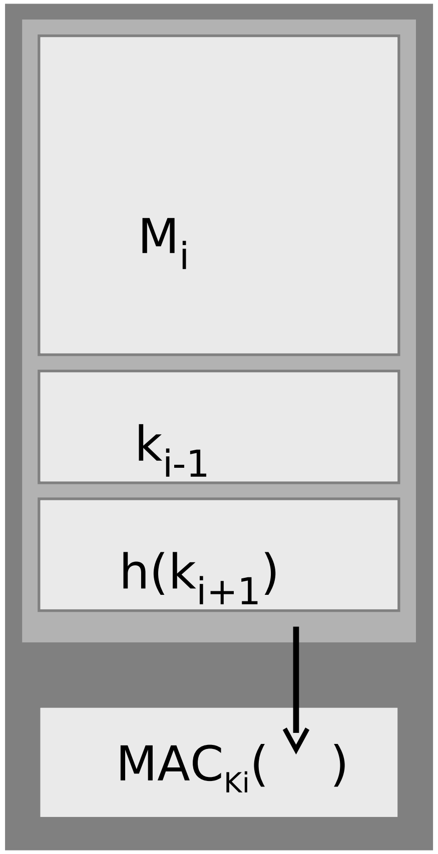
Guy Fawkes protocol (Anderson et al. 1998)

$P_i = M_i, k_{i-1}, h(k_{i+1}), MAC_{ki}(M_i, k_{i-1}, h(k_{i+1}))$



Why you can no longer forge Pi

$P_i = M_i, k_{i-1}, h(k_{i+1}), MAC_{ki}(M_i, k_{i-1}, h(k_{i+1}))$



Changes to M_i later revealed by MAC

Recomputing fake MAC requires forging k_i (and continuing chain)

Forged k_i must then be revealed (in P_{i+1}), and it won't match with the commitment $h(k_i)$ contained in P_{i-1}

Guy Fawkes limitations

- Once k_i revealed in P_{i+1}, anyone can forge P_i
  - So, attack: delay packets, get both P_i and P_{i+1}, then start forging and pass along P^*_i
  - Before sending P_{i+1}, sender must be convinced that recipient got P_i
  - Can't guarantee that in broadcast scenario
- If recipient loses a packet, the chain is broken
  - Missing the $h(k_{i+1})$ in P_i means she can't validate P_{i+1} or any packet after that

TESLA protocol (Perrig et al. 2000)

Timed Efficient Stream Loss-tolerant Authentication

- Can be seen as evolution of Guy Fawkes, with extra tricks

- k_i revealed in P_{i+d}, not necessarily P_{i+1} ($d \in \{1,2,3\ldots\}$)
  - Allows several packets in transit, so improves throughput
  - More resistant to network delays
  - But trade-off with freshness
  - At least here you can choose

TESLA: Efficient

- Each packet Pi has multiple MACs!
  - Computed with different keys, say k_{i1}, k_{i2}, k_{i3}
  - The different keys are released at different delays: k_{i1} released in P_{i+d1}, k_{i2} in P_{i+d2}, k_{i3} in P_{i+d3}
  - Even with network delays, you can still validate P_i if at least one key has not been sent yet
  - But as soon as you get a key, you can validate it
  - Robustness vs freshness: no longer a trade-off!

TESLA: Loss-tolerant

Tolerance to packet loss:
keys are generated with a Lamport hash chain

$k_{i-1} = h(k_i)$

Sender starts with $k_{1000000}$, chosen at random, and generates a million k_i in reverse; then uses them one by one, starting with k_1.

Even if you lose packets, any key k_j can be validated by any previous known-good key k_i in that chain. Just hash k_j for j-i times and see if you get k_i.