Principles of Computer Security

Dr George Danezis
(g.danezis@ucl.ac.uk)

Why SecAppDev?

- Learning security on the job is necessary. However,
- Foundations:
  - Principles. (Today)
  - Access control.
- Advances:
  - Privacy-friendly computations.
  - Selective disclosure credentials.
  - On-line currencies.

What makes computer security a distinct subject?

“Properties of a computer system must be maintained despite a resourced strategic adversary”

Compare with:

- Safety
- Robustness
- Program correctness

What properties?

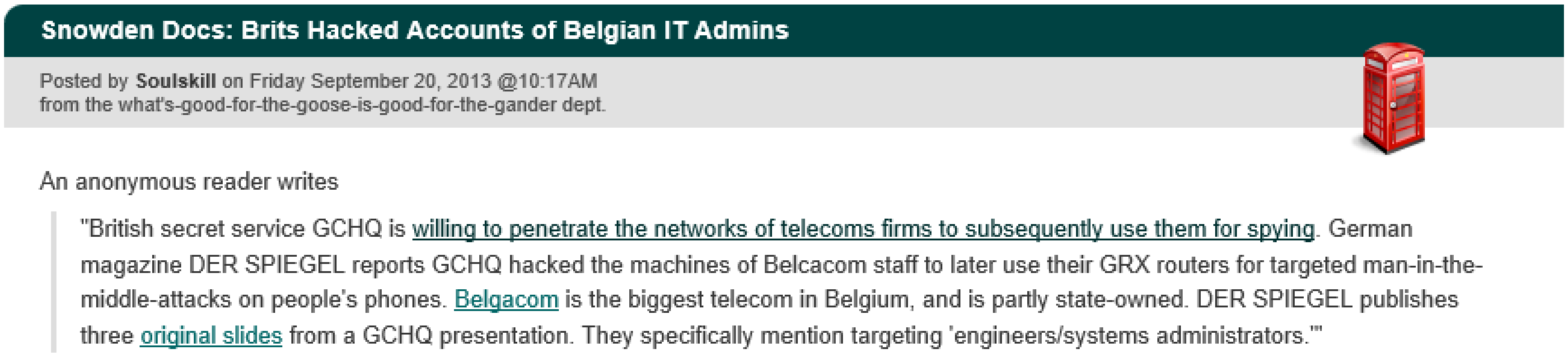
"Is this systems secure?" – meaningless question outside the right context

- Key concept: **"Security Policy"**
  A high level description of the principals,
  assets and properties that must hold in the system.
- Traditional: (CIA)
  - Confidentiality – e.g. authorized principals may read.
  - Integrity – e.g. authorized principals may write.
  - Availability – e.g. authorized principals can access the system.
- Others:
  - Authenticity, anonymity, non-repudiation, forward secrecy
- Some properties have no name:
  - Define security properties in terms of programs or games.
  - Is the system doing what it was meant to do: nothing more, nothing less.

Well resourced strategic adversary

- Key concept: **“Threat Model”**
- Adversary resources and capabilities:
  - Every power that the adversary has.
  - E.g. parts of the system observed, parts of the system that can be influenced, parties they can corrupt.
- Strategic:
  - The adversary will chose to commit resources optimally to violate the security properties.
  - “Ask what not how!”

Example I: The State level adversary



Snowden Docs: Brits Hacked Accounts of Belgian IT Admins

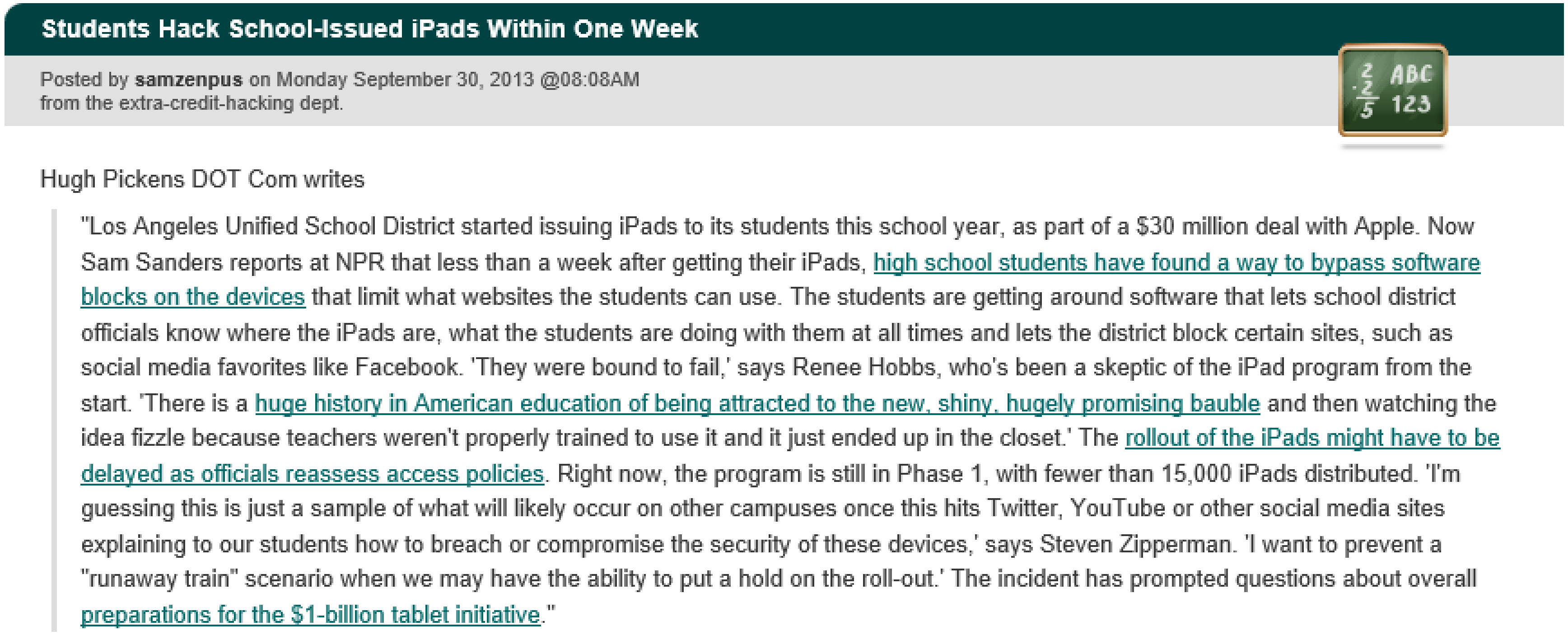
Posted by **Soulskill** on Friday September 20, 2013 @10:17AM
from the what's-good-for-the-goose-is-good-for-the-gander dept.

An anonymous reader writes

> "British secret service GCHQ is willing to penetrate the networks of telecoms firms to subsequently use them for spying. German magazine DER SPIEGEL reports GCHQ hacked the machines of Belcacom staff to later use their GRX routers for targeted man-in-the-middle-attacks on people's phones. Belgacom is the biggest telecom in Belgium, and is partly state-owned. DER SPIEGEL publishes three original slides from a GCHQ presentation. They specifically mention targeting 'engineers/systems administrators.'"

- What is the security policy?
  - Who are the principals?
  - What are their assets?
  - What are the security properties they try to maintain?
- What is the threat model?

Example II: The Teenage Adversary



Students Hack School-Issued iPads Within One Week

Posted by **samzenpus** on Monday September 30, 2013 @08:08AM
from the extra-credit-hacking dept.

Hugh Pickens DOT Com writes

> "Los Angeles Unified School District started issuing iPads to its students this school year, as part of a $30 million deal with Apple. Now Sam Sanders reports at NPR that less than a week after getting their iPads, high school students have found a way to bypass software blocks on the devices that limit what websites the students can use. The students are getting around software that lets school district officials know where the iPads are, what the students are doing with them at all times and lets the district block certain sites, such as social media favorites like Facebook. 'They were bound to fail,' says Renee Hobbs, who's been a skeptic of the iPad program from the start. 'There is a huge history in American education of being attracted to the new, shiny, hugely promising bauble and then watching the idea fizzle because teachers weren't properly trained to use it and it just ended up in the closet.' The rollout of the iPads might have to be delayed as officials reassess access policies. Right now, the program is still in Phase 1, with fewer than 15,000 iPads distributed. 'I'm guessing this is just a sample of what will likely occur on other campuses once this hits Twitter, YouTube or other social media sites explaining to our students how to breach or compromise the security of these devices,' says Steven Zipperman. 'I want to prevent a "runaway train" scenario when we may have the ability to put a hold on the roll-out.' The incident has prompted questions about overall preparations for the $1-billion tablet initiative."

- What is the security policy?
  - Principals, Assets, Security Properties
- What is the threat model?

Discussion

- Consider the security policies and threat models of the two previous examples.
  - “The State Level Adversary”, where a national telecommunication carrier tries to prevent a national security agency from eavesdropping on customer calls.
  - “The Teenage Adversary”, where the education authorities are trying to prevent teenagers accessing Facebook from a device given to them.
- Which of the two security systems is most likely to preserve its security policy? (And why?)

Note I: Asymmetry between attack and defence

- Attacker: needs to find one way to violate a security property.
  - Given the resources in the threat model.
  - Any one: lowest hanging fruit.
- Defender: needs to ensure that no adversary strategy can violate the security policy.
  - Much harder job!

Note II:
Are Adversary motives important?

- Possibly many motives: fame, money, commercial advantage, military advantage, political …

  (+ Other unknown motives)
- Verus: One bottom line motive:

  Violate the security policy.
- Security mechanisms: Motives above distilled into adversary resources in the threat model.

Note III:
Where do security policies come from?

- Factors in formulating security policy:
  - Security Engineering,
  - Business,
  - Risk Management,
  - Legal and Compliance.

- Must be revised as the above change.

So, when is a system “secure”?

- A system is “secure” if an adversary within a specific threat model cannot violate the security policy.

- Observe security systems around you!
- Key skill: Infer policies and threat models
  - Learn to discern the “Security policy”
  - Learn to discern the “Threat model”

Security Policy and Security Mechanisms

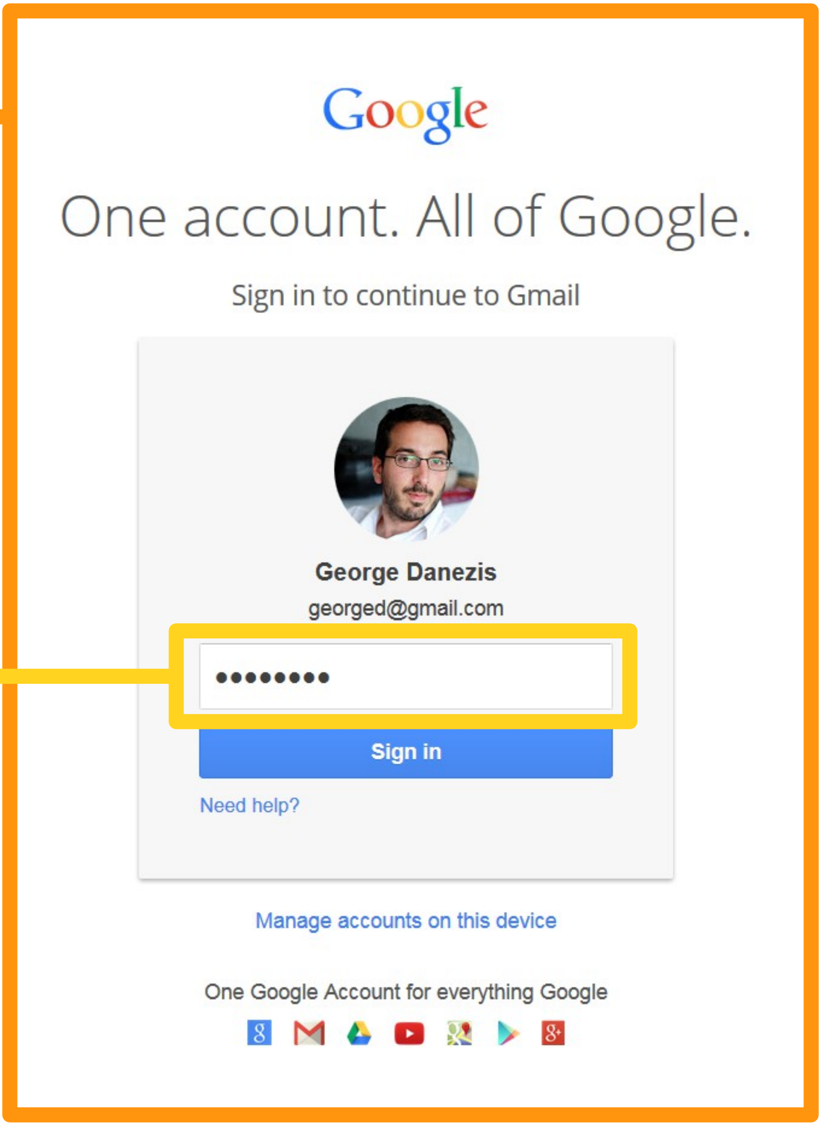
- Technical security mechanisms are used to ensure that the security policy is not violated by an adversary within the threat model.
- A rigorous argument is needed to show the security mechanisms are indeed effective in maintaining the security policy. (Key concept: the **“security argument”**)
- These mechanisms are the essence of the technical side of computer security.

Policy or mechanism? (I)

Authenticity: only the authorized user / owner of the account is allowed to see or modify data.

Confidentiality: The password needs to remain secret for the security policy to hold.

What is the security policy: **Confidentiality** or **Authenticity**?



Google

One account. All of Google.

Sign in to continue to Gmail

George Danezis

georged@gmail.com

••••••••

Sign in
Need help?
Manage accounts on this device
One Google Account for everything Google

Policy or mechanism? (II)

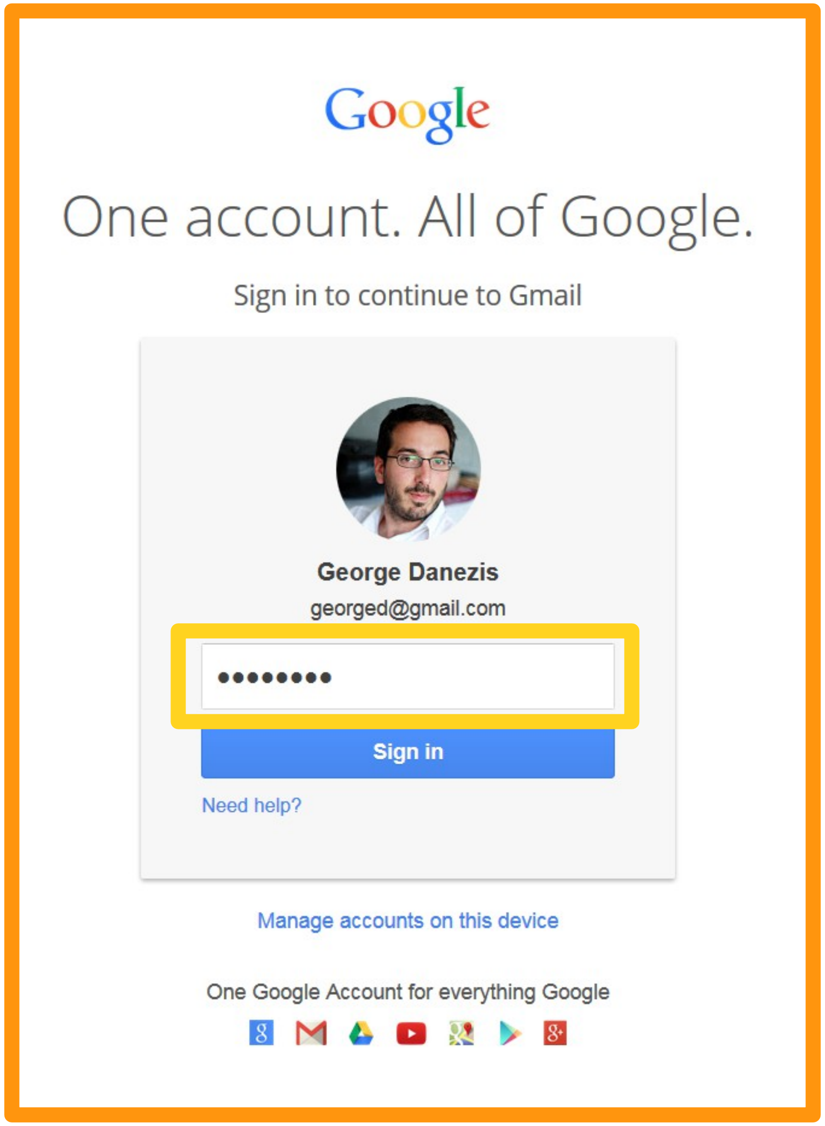
High level property: authenticity.

Mechanisms rely on “things that the adversary should not be able to do”:

- These are like “security properties”, but are also internal to the security mechanism.
- How to distinguish: “Is there another way of achieving authenticity?” Yes! (Token, Biometric, Social)

Problems with internal security properties:

- Repercussions not well understood by system users.
- User security decisions at the level of “internal security properties” will be misunderstood by users. Not within their realm. Violate the interface a user expects.


Google

One account. All of Google.

Sign in to continue to Gmail

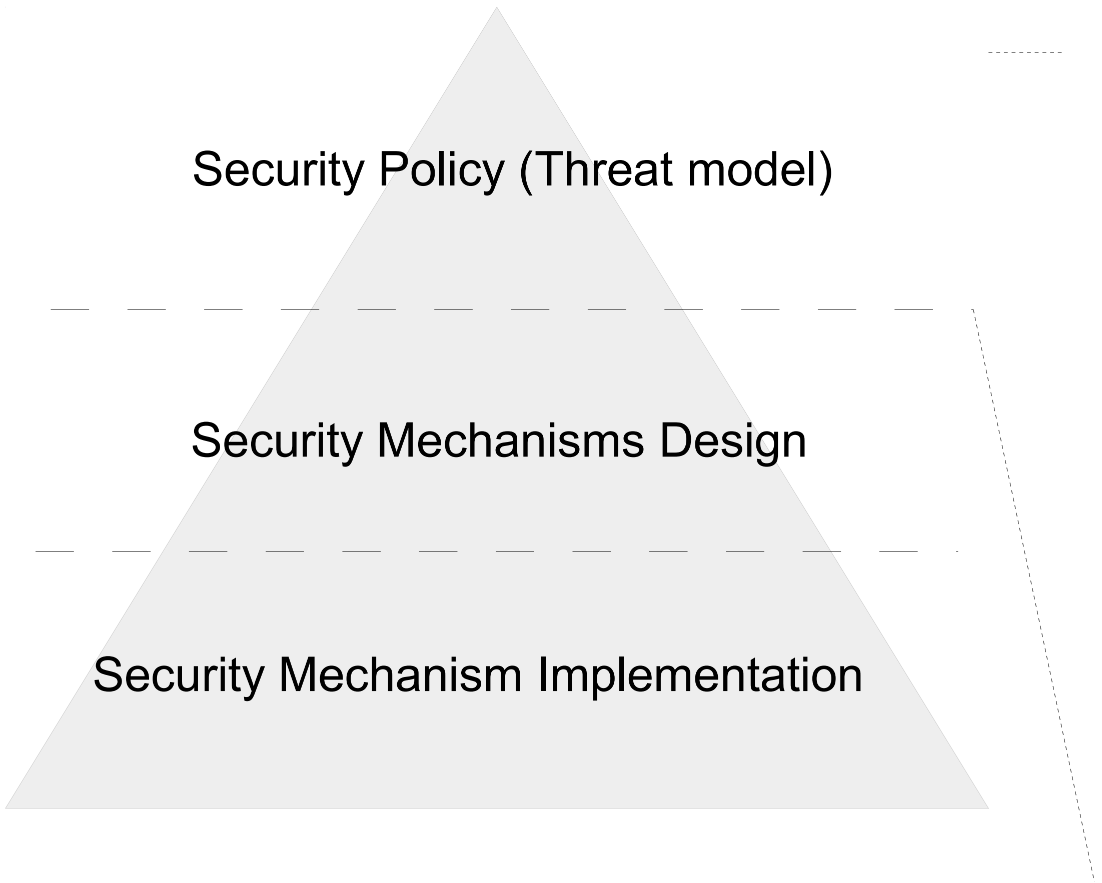
George Danezis

georged@gmail.com

••••••••

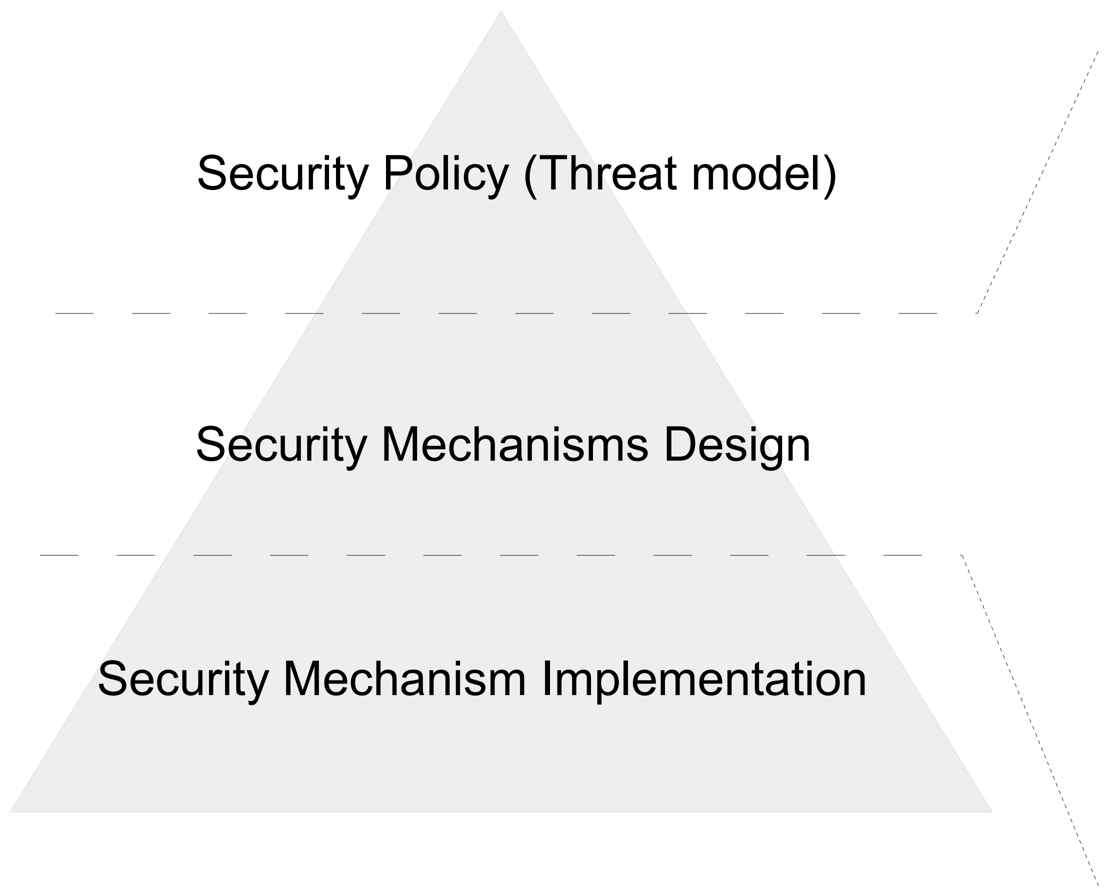
Sign in
Need help?
Manage accounts on this device
One Google Account for everything Google

The Security Engineering Pyramid (I)


Security Policy (Threat model)
Security Mechanisms Design
Security Mechanism Implementation

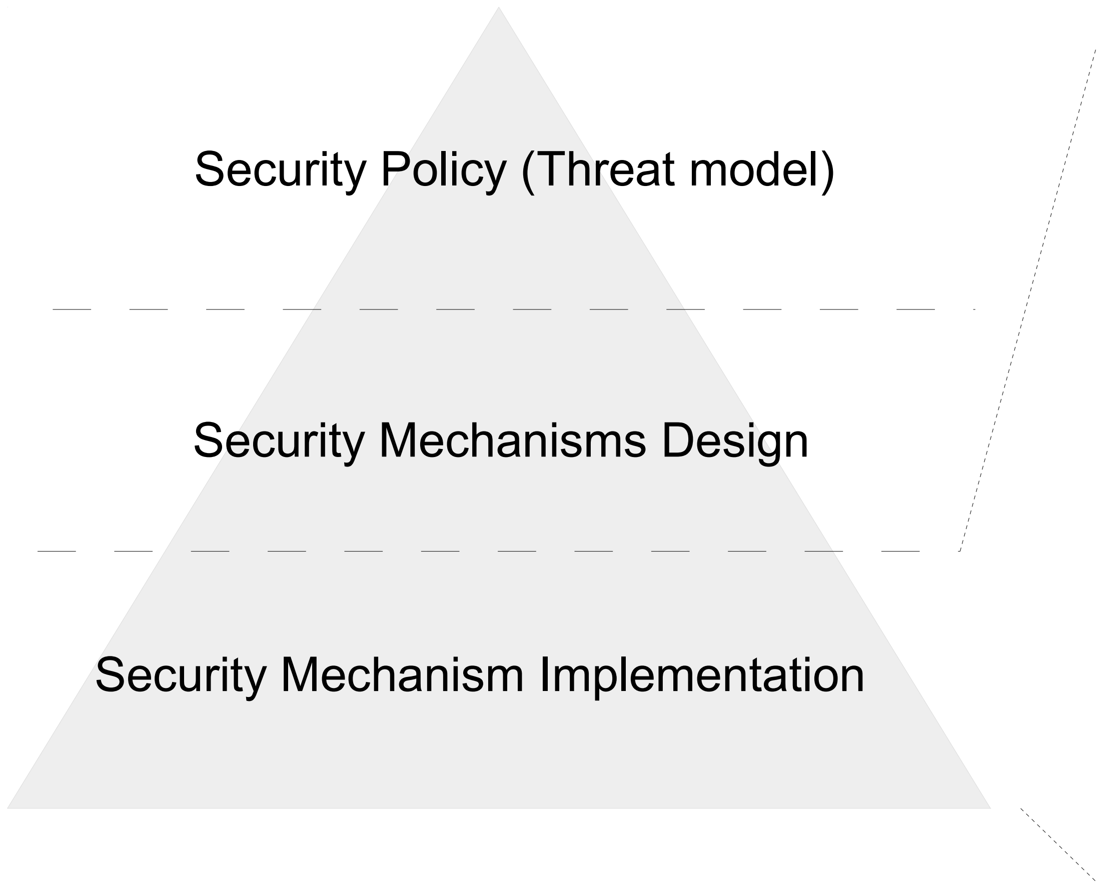
- Security Policy:
  - Principals
  - Assets
  - Security Properties
- Threat Model
  - Capabilities of adversary

The Security Engineering Pyramid (II)


Security Policy (Threat model)
Security Mechanisms Design
Security Mechanism Implementation

- Protection mechanisms:
  - Architecture
  - Special hardware
  - Software design
  - Cryptography design

The Security Engineering Pyramid (III)


Security Policy (Threat model)
Security Mechanisms Design
Security Mechanism Implementation

- Mechanism Implementation
  - Hardware Side channels
  - Coding bugs
- Operations
  - Corrupt insiders

Eight Principles for security mechanisms (and Two more)

- Key reading:
  J. Saltzer and M. Schroeder. The Protection of Information in Computer Systems. Fourth ACM Symposium on Operating Systems Principles (October 1973)

- A tutorial that draws together existing know-how ('75)
- Note the "Glossary"
- Principles are not hard rules, but deviations should be justified.

Economy of mechanism

- "*Keep the* [security mechanism / implementation] *design as simple and small as possible*" [SS75]
  - Why? Easier to audit and verify.
  - Operational testing is not appropriate for security. (But penetration testing may be valuable)

- Related key concept: "Trusted Computing Base" (TCB)
  - Every component in the system on which the security policy replies upon.
  - Needs to be kept small.

A bit more on the “Trusted Computing Base” (TCB)

- The TCB contains all aspects of the system on which the security policy relies.
  - Hardware, firmware, software
- If any of those go wrong in the TCB, the security policy can be violated.
  - If something goes wrong outside the TCB, the security policy still holds.

- Counter intuitive connotations of “Trusted”
  - Relying on many trusted parts is bad – Key concept: “The attack surface”
  - Want to minimize the attack surface.
  - Proper use of the verb “to trust” in Security Engineering:
    “X trusts Y will do Z.”

Fail-safe defaults

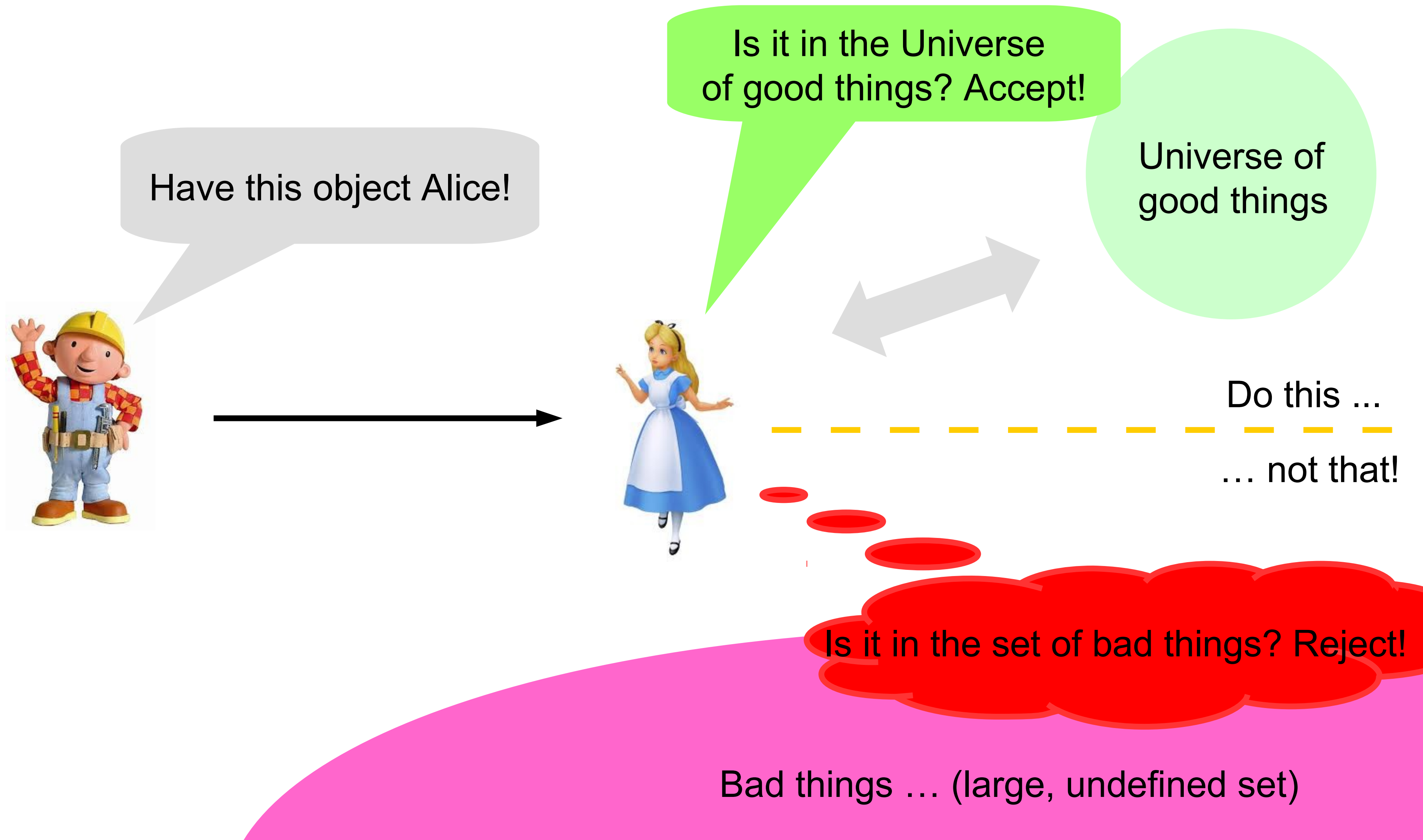
- "*Base access decisions on permission rather than exclusion*" [SS75]
  - Whitelist, do not blacklist.
  - Errors / uncertainty should err on the side of the security policy.
- Examples:
  - (Integrity) User input in forms.
  - (Confidentiality) Cipher suit negotiation in SSL

Fundamental principle of sanitization

- Positively verify that "low" integrity objects are within a valid set before elevating their integrity to "high" integrity.
  - White list: check that all properties of good objects hold.
  - Do not blacklist: just checking for bad objects or properties.

- Examples:
  - Expect a telephone number? Check that it is of the form "\+[0-9]{1,20}", not simply that it does not contain letters.
  - Expect text to display as a caption of a photo in a web album? Ensure that it is from a restricted set of Unicode, or apply to it a transform to "escape" / "encode" any characters not from that safe set into it. Do not simply check it does not contain "<script>". (XSS Attack)

Principle illustrated


Have this object Alice!
Is it in the Universe of good things? Accept!
Universe of good things
Do this ...
… not that!
Is it in the set of bad things? Reject!
Bad things … (large, undefined set)

Complete mediation

- "*Every access to every object must be checked for authority*" [SS75]
  - All actions are subject to the security policy
- Key concept: "The reference monitor"
  - The component (both design and implementation) that mediates all actions and ensures they are according to the policy.
- Problems with complete mediation:
  - Bottleneck?
  - Distributed systems?
  - Whose reference monitor?

Open Design

“No Security by Obscurity”

- “*The design should not be secret*” [SS75]
  - Only very specific passwords / keys should be assumed are secret. (Kerckhoffs’ principle from 1883)
  - Ease of auditing: To many eyes all bugs are shallow.
  - Difficult in practice to keep design secret (unrealistic threat models assume designs stay secret).

- This is the key principle that allows, and requires, an open academic discipline devoted to understanding computer security.

Open Design and Open Systems

- Is “Open Design” just a fall back?
  - Design may leak or be reverse-engineered.
  - No problem if that happens – but still some benefit from obscurity?
  - Maybe: custom system with one stakeholder.

- Open Systems:
  - Cars, Power Stations, Operating Systems, Network Protocols, DVD players.
  - Many stakeholders = multilateral properties.
  - Who gets to evaluate the design?
  - All security properties reduce to the interests of that entity.

Case Study:
GSM Security.
Great for billing security.
Not so great for privacy.

Separation of Privilege

- “*Where feasible, a protection mechanism that requires two keys to unlock it is more robust and flexible than one that allows access to the presenter of only a single key*.” [SS75]

- Down side?
  - Availability
  - Complexity of orchestration.
  - Dilution of responsibility.

Least Privilege

- "*Every program and every user of the system should operate using the least set of privileges necessary to complete the job*" [SS75]
- Examples:
  - (Integrity) DB program, can only write the DB.
  - (Privacy) Data minimization principle.

Least Common Mechanism

- "*Minimize the amount of mechanism common to more than one user and depended on by all users*" [75]
- What does that mean? What about simplicity?
  - "Every shared mechanism represents a potential information path between users and must be designed with great care to be sure it does not unintentionally compromise security"
  - Interactions in design make validation of security design hard.
  - Interactions in the implementation may e.g. lead to unintentional leaks of information.
  - Robustness of single points of failure.

Psychological Acceptability

- "*It is essential that the human interface be designed for ease of use, so that users routinely and automatically apply the protection mechanisms correctly*" [SS75]

- Mental model of the (honest) users must match security policy and security mechanisms.
- Cultural acceptability:
  - (Authentication) Photographs that must uncover faces.
  - (Safety) Register of everyone who sleeps in a dorm.

Problematic: Work Factor

- "*Compare the cost of circumventing the mechanism with the resources of a potential attacker*" [SS75]

- Upside: Refine threat model
- Downside: difficult to quantify.
  - How do you know the cost of seducing an insider?
  - What is the cost of finding a bug?
  - Dangerous extension: model "monetary gain" of adversary

Problematic: Compromise Recording

- "*Reliably record that a compromise of information has occurred [...] in place of more elaborate mechanisms that completely prevent loss*." [SS75]

- Upsides:
  - Instantiation: Keep logs
  - Instantiation: Designs based on tamper evidence (seals).
  - Can sometimes recover after compromise.
- Downsides:
  - How do you know a compromise has occurred?
  - What if you cannot recover?
  - Integrity of logs? Logs may add to vulnerability (privacy)?
  - Who analyses the logs? (Expensive)

Exercise: Compare two security mechanisms on the basis of established computer security principles.

- Back to our on-line banking example: Mechanism to achieve “two-factor authentication”
  - Mechanism one: web password & mobile phone app.
  - Mechanism two: web password & special card reader device.

- Discuss...

Cheat Sheet of principles [SS75]

Economy of mechanism

Fail-safe defaults

Complete mediation

Open Design

Separation of Privilege

Least Privilege

Psychological Acceptability

Work Factor

Compromise Recording

Other key concepts:

- Trusted Computing Base (TCB)
- Reference Monitor

Subsequent principles: Weakest link, versus defence in depth

- Big security systems are build from smaller ones:
  - "Composition" of secure systems
  - It is not always secure to compose two secure systems.
  - Two models for composition of secure systems:

Weakest link: if any sub-system is broken the security policy is violated.

Defence in depth: if any sub-system remains secure, the security policy is enforced.

Examples of composition

- Back to the example of authentication:
  - 2-factor authentication with web password and device (like phone): layered, defence in depth.
  - Password recovery mechanisms, where a password reset is sent to an email, or personal knowledge questions are asked: weakest link.
- Secure composition of security systems is one of the hardest aspects of security engineering.

Subsequent Principles: Worse case versus Average case

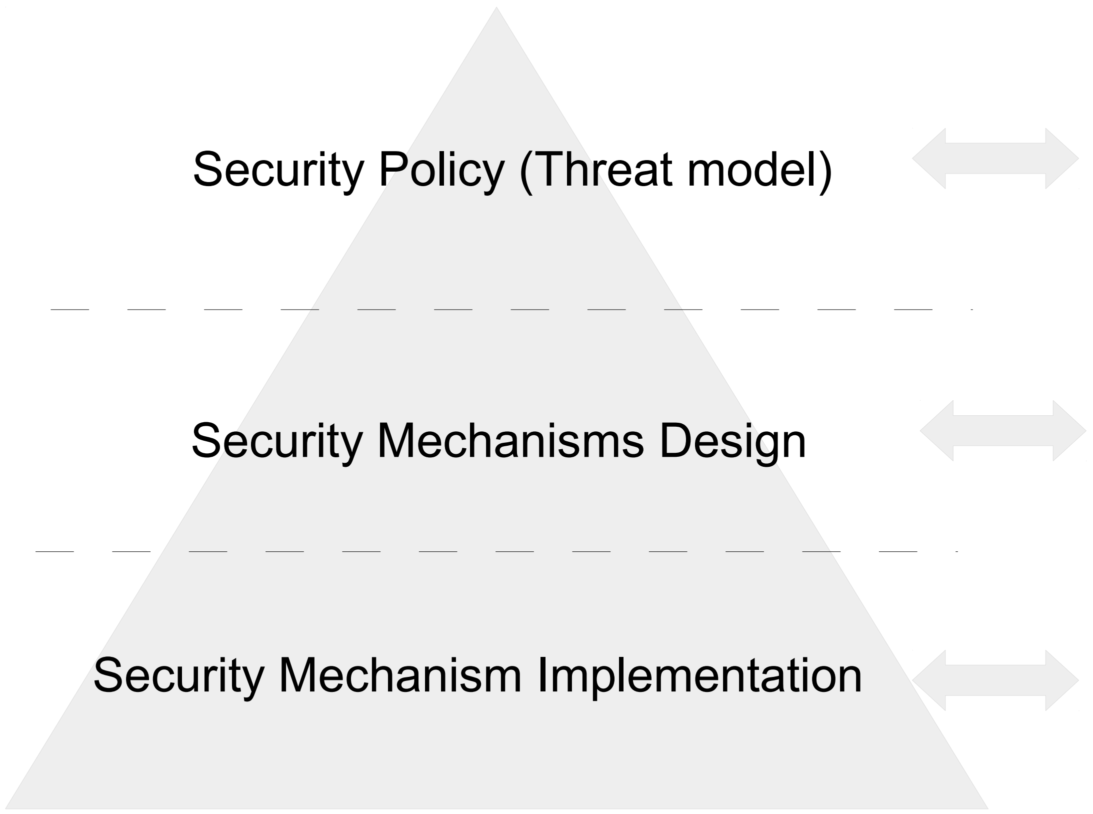
- How to measure the degree of protection afforded by a security system:
  - In general: Important open question!
- On the basis of the worse case: Take the inputs from both the honest users, and the adversary that produces the worse outcomes (in terms of violating the security policy).
- On the basis of the average case: Given the actions of a “typical” / “average” user, and the worse actions of an adversary measure the outcome.

Pros and cons

- “Worse case” security measure:
  - Makes no assumptions on the user behaviour within the security policy.
  - Strong guarantee
  - Pessimistic – low performance.
  - Examples: Cryptographic primitives
- “Average case” security measures:
  - What is a typical user?
  - Difficult to second guess which actions are more important to protect within the security policy.
  - More fragile.
  - Examples: data anonymization, network anonymization.

Engineering Secure Systems

Security engineering may have profound implications on the rest of the system design.



Security Policy (Threat model)
Security Mechanisms Design
Security Mechanism Implementation

Business Model, Policies, Values, Assets, Legal context, Human context, attitude to risk.

Technical Architecture, choice of hardware, network protocols, platforms, engineering methodology, evaluation and testing

Supply chain, programming methods, operations, HR, vetting, backups, ...

A Final Note on Principles

“Ultimately, you must forget about technique. The further you progress, the fewer teachings there are. The Great Path is really No Path.”
– Morihei Ueshiba

- Principles allow us to identify safe and unsafe patterns in the security engineering process.
  - Security Policies and Threat Models
  - The Security Pyramid
  - The 10 principles [SS75]
  - And a few more.

- Do not use principles as a blind checklist.