CS 4/591: Introduction to Computer Security

Lecture 9: Assurance & Evaluation

James Hook

Objectives

- Introduce Assurance as a concept/goal
- Introduce methods to increase assurance
- Introduce frameworks for Certification

Why do you trust an Airplane?

- Which of these do you trust more? Why?

NASA images from web site: http://www.dfrc.nasa.gov/Gallery/Photo/

Boeing images from web site: http://www.boeing.com/companyoffices/gallery/flash.html

Discussion points:

- Who's flying?
  - How many hours have they been awake?
- How long have the airframes been in service?
- Risk/benefit: If you want to go into space you don't have a lot of choices
  - Best to limit to "apples to apples"

Trusting Commercial Aircraft

- Specification integrity
  - Clear scope of project; goal of aircraft
- Design integrity
  - State of the art engineering analysis of design
  - Extensive modeling (physical and simulation) based on established best-practices of a mature engineering discipline
  - FAA review
- Manufacturing integrity
  - Extensive process controls and tests for all components
  - Rigor appropriate to risk (entertainment system vs. autopilot)

Trusting Commercial Aircraft

- Operational integrity
  - Maintenance is performed by certified mechanics
  - Maintenance performed on schedule
  - Maintenance includes diagnostic measurements confirming conformance to design specifications
  - Pilot is licensed to fly
  - Pilot inspects aircraft prior to flight (and she's on the plane!)
  - Pilot does not perform maintenance (Separation of duty)
- Feedback
  - Independent investigation of failures
  - If design defects or manufacturing defects are identified the entire fleet can be grounded or repaired

Are all Aircraft Trustworthy?

- Federal regulations reflect risk
- Crudely: Level of assurance increases as potential cost of failure increases
- Commercial aviation is “high assurance”

Can you trust systems that include software?

- Some modern aircraft are "fly by wire"
- How do we trust them?
- FAA
  - Lots of testing
  - Lots of review
  - Lots of process-based controls of both
- Techniques that work for high assurance embedded systems are hard to scale

Trusting Information Systems

- How can we trust an information system?
- What can we trust it to do?
- Can we trust a mechanism to implement a policy?
- How well does the analogy to aviation apply?

The Analogy

- Key factor of trust of commercial airplanes is that we trust the engineering processes used to design, build, maintain, and improve them
- Assurance techniques for information systems are predicated on software engineering practices
  - Is our discipline a sufficiently mature engineering discipline to earn the trust that the public has placed in us?
- Sullivan and Bishop's presentation builds on what are accepted as best practices in Software Engineering
- Anderson's presentation is a little more skeptical

Assurance & Trust

- Sullivan builds on three related ideas:
  - *Trustworthy*: sufficient credible evidence that the system will meet … requirements
  - *Trust*: a measure of trustworthiness
  - *Security Assurance*: confidence that an entity meets its security requirements, based on evidence provided by the application of assurance techniques
    - E.g.: development methodology; formal methods; testing; …
- So what's the difference between "trustworthy" and "security assurance"?
  - Does a system have to be correct to be secure?

Ross Anderson on Assurance

- "Fundamentally, assurance comes down to the question of whether capable, motivated people have beat up on the system enough. But how do you define enough? And how do you define the system? How do you deal with people who protect the wrong thing, … out of date or plain wrong? … allow for human failures?"

Software Engineering

- Taxonomy of failures and design methods presupposes Software Engineering Principles
  - Classic lifecycle view of SE posits:
    - Requirements
    - Design
    - Implementation
    - Integration and Test
    - Operation and Maintenance

Design Assurance (broad)

- Requirements: statements of goals that must be satisfied
- For Security assurance, requirements should determine the security policy, or the space of possible security policies (*security model*), for the system
  - E.g. What is the access control mechanism? What are the subjects? What are the objects? What are the rights?
  - Is the access control policy mandatory? Discretionary? Originator controlled?
- The tools introduced in class to date provide a vocabulary for expressing security models, policies, and mechanisms

Policy Assurance

- Evidence that the set of security requirements is complete, consistent and technically sound
  - Complete:
    - Logic: complete means every sentence is either true or false
    - Security: every system state can be classified as "safe" or "unsafe"
  - Consistent:
    - Logic: there is no sentence that is both true and false, or, equivalently that the sentence "false" is not a theorem
    - Security: no system state is both "safe" and "unsafe".
  - Technically sound:
    - Logic: a rule is sound if it does not introduce inconsistencies
    - ? I think the author intends a necessarily informal notion that the model is appropriate to the situation

Policy Assurance Examples

- The original BLP papers show that the model is complete and consistent
- The Volpano, Irvine and Smith paper shows that the Denning and Denning Information Flow Security concepts can be made sound
  - That analysis is necessarily incomplete (halting problem)
- Many Policy Assurance arguments are carried out using
  - "rigorous mathematics" (I.e. pencil and paper proofs)
  - some use theorem provers (machine checked proofs)

Design Assurance (strict)

- Design is sufficient to meet the requirements of the policy
  - What is a design?
    - Architecture
    - Hardware software components
    - Communication mechanisms
    - Use-cases?
    - Threat profile?

Implementation Assurance

- Evidence establishing the implementation is consistent with the requirements and policy
  - Generally this is done by showing the implementation is consistent with the design, which is consistent with requirements and policy…
- Considerations
  - Design implemented correctly
  - Evidence that appropriate tools and practices used to avoid introducing vulnerabilities (e.g. code insertion/buffer overflow)
  - Testing
  - Proof of correctness
  - Documentation

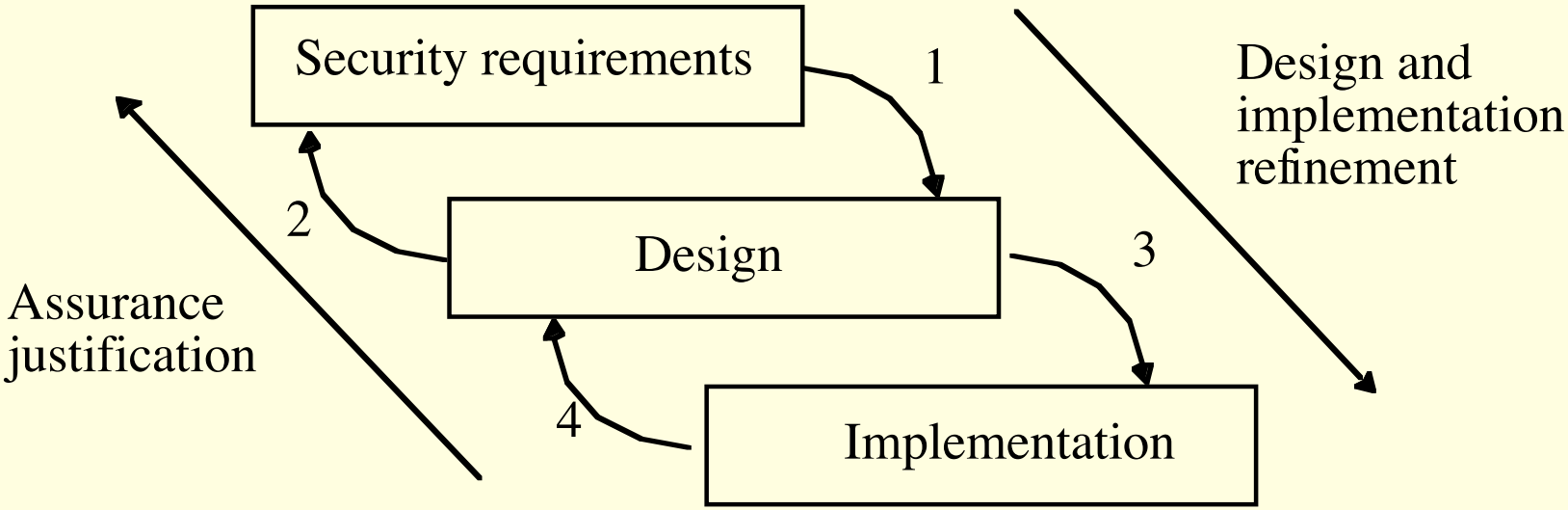
Operational Assurance

- Evidence the system sustains the security policy requirements during installation, configuration, and day-to-day operation
- Text mentions documentation
- Usability testing is also key
  - Human-Computer Interaction studies are underutilized in mainstream assurance practices!
- Ross Anderson: usability is "the spectre at the feast"

Structure of An Assurance Argument

- Software Engineering Process View is typically used to organize assurance argument
- Software is viewed to have a "life cycle"
  - Inspired by biology

Life Cycle


Security requirements
1
Design and
implementation
refinement
2
Design
3
Assurance
justification
4
Implementation

Life Cycle Assurance

- Conception
  - Initial focus is on policy and requirements
- Manufacture
  - Select mechanisms to enforce policy
  - Give evidence that mechanisms are appropriate
- Deployment
  - Prepare operational plans that realize policy goals
  - Provide mechanism for distribution and delivery that assures product integrity
  - Support appropriate configuration
- Fielded Product Life
  - Update and patch mechanism
  - Customer support
  - Product decommissioning and end of life

Assurance

- Myth or Reality?
- Are we behaving like good engineers and avoiding the Failures of Past?
  - Or are we alchemists promising to make gold out of manure?
- If we really cared about code insertion attacks would we use C for routine programming 18 years after the Morris worm?

Confounding Issue

- In Software Engineering which matters more:
  - People
  - Tools
  - Process
- All evidence of which I am aware says people matter more than tools or process
- Given this, can we achieve assurance by mandating tools and process?

Anderson: Incentives

- Security Engineering
  - Incentives: If people don't want to protect a system it's hard to make them
  - Policy: People often end up protecting the wrong things, or protecting the right things in the wrong way
  - Mechanisms: US export controls led to …DVDs being shipped …that were intrinsically vulnerable.
  - Assurance of architecture/implementation: Does this address the exploitable bugs, such as stack overflows, race conditions, etc.

Incentives

- Who is at risk if it fails?
  - The developers?
  - The certifying agency?
  - The operating agency?

Quis Custodiet?

- Quis custodiet ipsos custodes?
  - Who shall watch the watchmen?

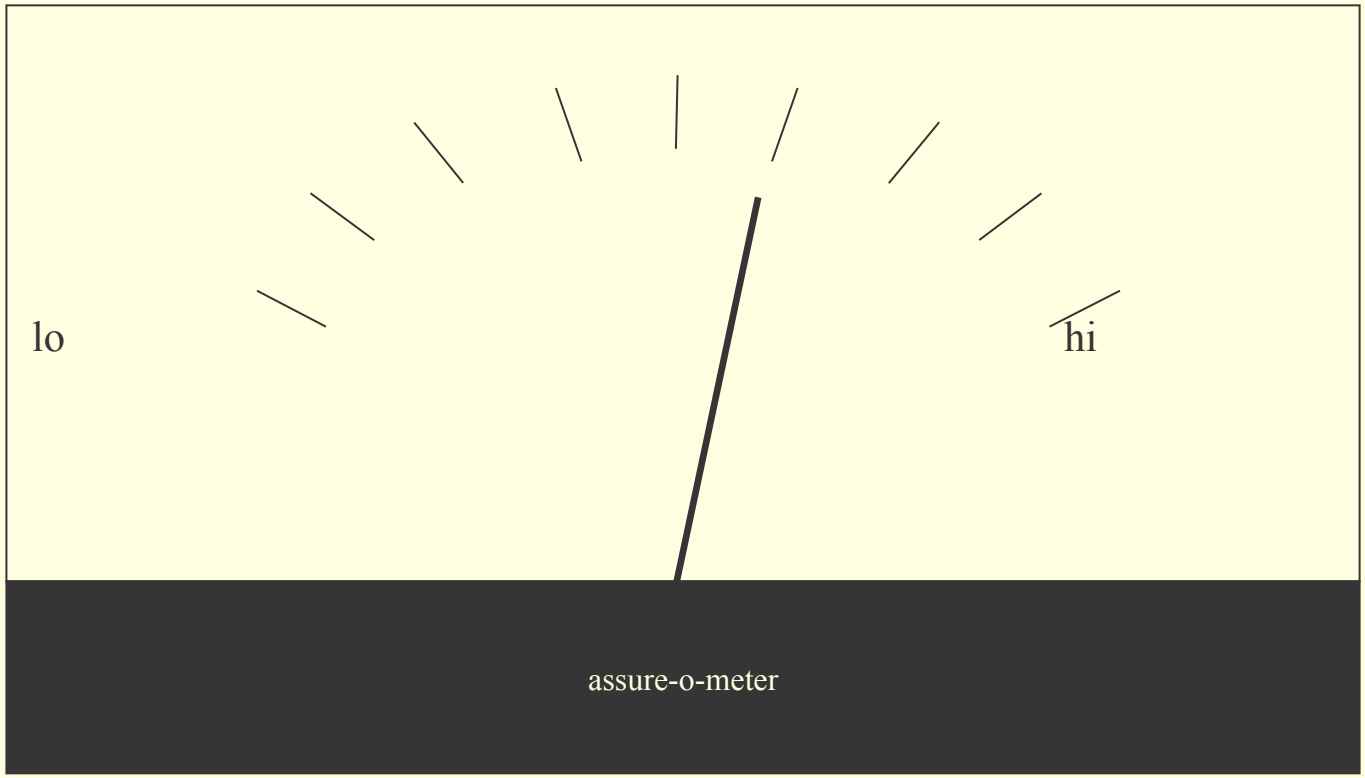
Anderson (cont)

- "Government agencies' … dream is to be able to buy commercial off-the-shelf (COTS) products, replace a small number of components …, and end up with something they can use with existing defense networks. … There is little concern with usability … This wish list is unrealistic given not just the cost of high assurance, but also the primacy of time-to-market, …, and the need for frequent product versioning to prevent the commoditization of markets."

Anderson scenario

- Paddy, IRA terrorist: 1,000 hours per year
  - Finds 1 exploitable bug
- Brian, GCHQ + NSA: 10,000,000 hours per year
  - Finds 10,000 bugs
- Probability Brian found Paddy's bug?
  - Less than 1%

Evaluation



Evaluation

- Context:
  - DoD identifies computer security as important in ‘70s (Anderson 1972)
  - Recognizes trend toward networking: computing is communication
  - Economic forces dictate they purchase products built outside of the DoD
  - Need: Procurement guidelines for DoD to purchase security critical software

First Step

- James Anderson's "Computer Security Planning Study" provides a blueprint
- Needs analysis:
  - Multi-level operation
  - Systems connected to the world
  - On-line operation
  - Networks
- Vision
  - Security engineering
  - Secure components (hardware & software)
  - Handbook of Computer Security Techniques

Issues

- How to accelerate maturation of a discipline?
- Desire: codify best practices
- What if current practice is insufficient?
  - Legislate what we think best practices should be!

First Attempt

- Trusted Computer Systems Evaluation Criteria (aka "Orange Book")
  - 1985 -- 2000
- Classify systems in a scale:
  - C1, C2, B1, B2, B3, A1

Orange Book

- C1: Discretionary access control by groups of users
- C2: Discretionary access control by single users; object reuse; audit
  - "Carefully configured commercial systems"

Orange Book (cont)

- B1: Mandatory access control.
  - MAC labels; BLP-like policy enforced
- B2: Structured protection
  - B1 +
  - formal model of policy,
  - proof of consistency,
  - tools for administration and configuration management
  - TCB structured and interface clearly defined
  - Cover channel analysis
  - Trusted path from User to TCB
  - Severe testing (penetration testing)

Orange Book (cont)

- B3: Security domains
  - As B2 +
  - TCB
    - minimal
    - Mediates all requests
    - Tamper resistant
    - Able to withstand formal analysis and testing
  - Real-time monitoring and alerting
  - Structured techniques used in implementation
- A1: Verification design
  - As B3, but formal techniques are used to prove equivalence between TCB spec and security policy

Orange Book evaluations

- Orange book evaluators worked for the government
  - Government is an interested party here (purchaser)
- Evaluations took a lot of time
  - Products, even if successfully certified were generations behind current technology
  - Both production and certification was very expensive
- Orange book evaluation led to paralysis
  - Producers and consumers were both frustrated

Orange Book issues

- Applied in broad domains
  - Eventually expanded to "rainbow series"
- Each level increased
  - Sophistication of threat model
  - Sophistication of required mechanisms
  - Sophistication of analysis
- Increasing any one dimension is hard, doing 3 simultaneously is nearly impossible
- Incentives
  - Vendor paid for evaluation
  - Motivated vendor to shop around

Crypto Standards

- Bishop/Sullivan outline a success in certification standards for crypto [FIPS 140-1]
- Domain was narrow
- Evaluation was informative to developers (evaluators found real bugs)
  - Adding value is key!
- Perceived as a success

Son of Orange Book

- Common Criteria attempts to "fix" Orange book issues
- Separates conflated dimensions
  - Identify a Target of Evaluation (ToE)
  - Identify a Security Target (ST)
  - Identify a Protection Profile (PP) reflecting threat context and domain-specific requirements
  - Classify development by "Evaluation Assurance Level"

Evaluation Assurance Level

- EAL 1: functionally tested
- EAL 2: structurally tested
- EAL 3: methodically tested and checked
- EAL 4: methodically designed, tested and reviewed
- EAL 5: semiformally designed and tested
- EAL 6: semiformally verified design and tested
- EAL 7: formally verified design and tested

Common Criteria

- International standard
- EAL 1 -- 5 transferred across borders
- EAL 6 and 7 are not

Follow up

- NIST: National Institute of Standards
  - Founded to make fire fighting equipment interoperable across municipal boundaries
  - Now tasked with standards that support commerce
- NSA: National Security Agency
  - Signals Intelligence
  - Protect all sensitive information for DoD
  - Make the Internet safe for commerce (expanded interpretation of mission in last decade)

NIST and NSA

- Both agencies are involved in CC and Crypto certification
- NIST is the agency designated with to evaluated Engineering Assurance Levels 1 - 5 and FIPS crypto
- NSA is the agency designated to evaluate EAL 6 and 7 and DoD crypto

NSA's Crypto levels

- Type 1: Used for classified information. Tamper resistant. No tempest radiation. Uses NSA certified algorithms.
- Type 2: NSA endorsed for telecommunications. Not for classified data. Government proprietary algorithms.
- Type 3: NIST certified FIPS crypto
- Type 4: Registered with NIST but not certified

Issues

- Sullivan and R Anderson present two perspectives on the result
  - "Orange Book" over promised for formal methods
  - Organizations failed to deliver most trusted products
    - Good engineers thought they weren't solving the **real** problems
  - Common Criteria attempt to avoid some Orange Book faults
    - Still: some science, some science fiction (EAL 6 and 7)
  - Can post-hoc analysis ever work?

DoD practice

- Practice is less strict than the dogma
- New "COTS strategy" appears to bypass CC and Orange Book
- Evaluation has become a barrier to procurement
  - If I ask for too much assurance and my procurement is delayed I fail at my mission

Looking Forward

- Good luck on the exam!
- Remember to hand in your term paper proposal at exam
- Have fun with Professor Binkley!

Thank you!