Goals of a security mechanism

- Given a policy that specifies what is “secure” and what is “non-secure” goal of security is to put in place mechanisms that provide:
  - Prevention
  - Detection
  - Recovery

Types of Security Mechanisms/controls

- Cryptography and cryptographic protocols.
- Software controls.
- Hardware controls.
- Physical controls.

Operational Issues in Security

- Risk Analysis or Assessment
- Cost-Benefit Analysis
- Laws and Regulations
- Human Issues: usability

Design Principles for Secure Systems

- Two basic themes:
  - Simplicity – KISS[1]
    Makes design and interactions easy
    Easy to prove its safety
  - Restriction
    Minimize the power of entities
- There are no “laws” of security
- Know the basic ideas
- Use these to help you reason about security

[1]KISS is an acronym for “Keep it simple, stupid” as a design principle noted by the U.S. Navy in 1960.

Principles of design

1. Principle of least privilege
2. Principle of fail-safe defaults
3. Principle of economy of mechanism
4. Principle of complete mediation
5. Principle of open design
6. Principle of separation of privilege
7. Principle of least common mechanism
8. Principle of psychological acceptability

Principle of least privilege

- Entity should be given only the information / privileges needed to finish a task
  - Temporary elevation of privilege should be relinquished immediately
  - Granularity of privileges
    Append permission only for logging process.
  - Strong privacy implications.

Principle of fail-safe defaults

- Use sane defaults. The default should be secure.
  - Default access to an object is none
  - Access Control Lists (ACLs), firewall examples.
  - Restricting privileges at the time of creation
  - What if the attacker's goal is to cause denial- of-service?
- “Fail-closed” (as opposed to "fail-open")

Principle of economy of mechanism

- Security mechanisms should be as simple as possible.
  - Fewer errors
  - Testing and verification is easy
  - Assumptions are less
- “Minimizing the Trusted-Computing Base”

Principle of complete mediation

- All accesses to objects should be checked to ensure they are allowed.
  - UNIX file descriptor
  - DNS cache poisoning.
  - Restrict caching policies
  - Security vs. performance issues

Principle of open design

- Security of a mechanism should not depend upon secrecy of its design or implementation (why not?)
  - Secrecy != security
  - Complexity != security
  - “Security through obscurity”
  - Cryptography and openness

Principle of separation of privilege

- System should not grant permission based on single condition
  - Company checks over $75,000 to be signed by two officers.
  - Example: “su” on BSD requires
    - User be in group “wheel”
    - User knows root password
    - Restrictive because it limits access
- “Don't put all of your eggs in one basket”

Principle of least common mechanism

- Mechanisms used to access resources should not be shared
  - Shared resources need resource isolation to prevent becoming a denial-of-service target
  - Restrictive because it limits sharing

Principle of psychological acceptability

- Security mechanism should not make the resource difficult to access
- Recognizes the most important element in security: HUMAN
- “Usability vs security”

Example

- The Stork package manager shares immutable copies of installed packages across OS VMs. It reduces duplicate package downloads between VMs and saves disk space, network bandwidth, and even memory.
- Which of the above principles does Stork follow or violate?

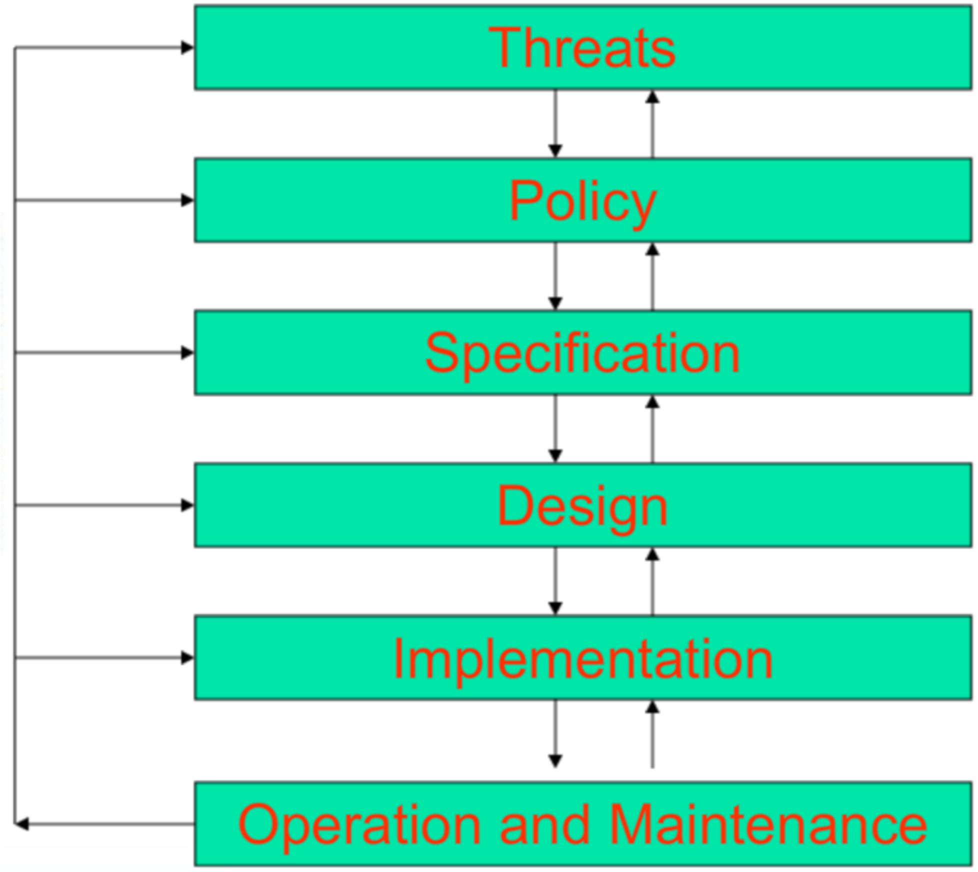
- Principle of least privilege
- Principle of fail-safe defaults
- Principle of economy of mechanism
- Principle of complete mediation
- Principle of open design
- Principle of separation of privilege
- Principle of least common mechanism
- Principle of psychological acceptability

- Stork violates the principles of least common mechanism and least privilege to prevent duplicate downloads. How would this impact the threat from a man-in-the-middle attacker?

3. Threat modeling

Security Life Cycle

- So far what we have learnt helps us in design, specification and implementation mainly.
  - What about others?
  - We start with threat analysis/modeling.



Why Threat Modeling

- Helps you understand your application better.
- Discover potential design flaws and vulnerabilities
- Prioritize security analysis
- Understand overall security risk
- Develop mitigating strategies
- Provide more complete analysis

Why Threat Modeling

- “My house is secure” is almost meaningless
  - Against a burglar? Against a meteor strike? A thermonuclear device?
- “My system is secure” is almost meaningless
  - Against what? To what extent?
- Threat modeling is a process to define the goals and constraints of a security solution
  - Translate user requirements to security requirements

Threat Modeling

- Threats and assets are key – vulnerabilities and attacks are only concerns if there is a threat to an asset to be concerned about.
- How do we identify and evaluate threats?
  - Arbitrary Threat or Attack Lists
    - Random and unstructured
    - Dubious completeness
  - Threat Trees or Attack Trees
    - More structured
    - Modular and Re-usable
    - Currently favored approach

Threat Modeling

- Start with questions like the following:
  - Who are my potential adversaries?
  - What is their motivation, and what are their goals?
  - How much inside information do they have?
  - How much funding do they have?
  - How averse are they to risk?
  - [Be paranoid: do not underestimate the attacker’s capability; do not also ignore easy/dumb attacks]
- Then enumerate threats by stepping through each of the system’s assets, reviewing a list of attack goals for each asset. Assets and threats are closely correlated.

Threat Modeling – main steps

1. Understand your system
2. Understand what assets/resources need to be protected
3. Predict who the potential attackers are against a particular asset and what are the possible (known) attacks
4. Perform risk assessment
    1. Determine what is the expected risk (quantitative or qualitative) because of an attack
5. Perform risk management: Employ security mechanisms (mitigation), if needed
    1. Determine if they are cost effective

Defining, using a threat model

- A Threat Model (TM) defines the security assertions and constraints for a product
  - Assets: What we’re protecting
  - Threats: What we’re protecting it against
  - Mitigations: How we’re protecting our Assets
- Use TM to narrow subsequent mitigation efforts
  - Don’t secure review, fuzz test all interfaces
  - Select the ones that are critical
- TM is part science, part art, part experience, part nuance, part preference
  - Few big assets vs lots of focused assets

Types of threats – Remember?

- Can be classified into four broad categories
  - Disclosure - unauthorized access to information
  - Deception - acceptance of false data
  - Disruption - interruption or prevention of correct operation
  - Usurpation - unauthorized control of some part of a system
- Examples include – snooping, sniffing, spoofing, delay, denial of service, theft of computational resources...

STRIDE Model

- In general, threats can be classified into six classes based on their effect :
  - Spoofing - Using someone else’s credentials to gain access to otherwise inaccessible assets.
  - Tampering - Changing data to mount an attack.
  - Repudiation - Occurs when a user denies performing an action, but the target of the action has no way to prove otherwise.
  - Information disclosure - The disclosure of information to a user who does not have permission to see it.
  - Denial of service - Reducing the ability of valid users to access resources.
  - Elevation of privilege - Occurs when an unprivileged user gains privileged status.

Ranking Threats

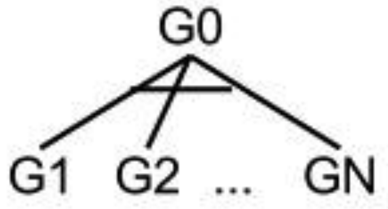
- Used for prioritizing work
- One methodology for ranking threats is the use of DREAD (used by Microsoft!)
  - Damage Potential
  - Reproducibility
  - Exploitability Cost (or cost and ease of performing attack)
  - Affected Users
  - Discoverability
- DREAD rating is calculated by adding the rating for each component
  - For example, 3: High, 2: Medium, 1: Low
  - For a particular threat, we might have
    - Damage Potential = 3
    - Reproducibility = 3
    - Exploitability Cost (or cost and ease of performing attack) =2
    - Affected Users = 2
    - Discoverability = 2
    - Total Rating: 12, which might be regarded as High, since one can set 12–15 as High, 8–11 as Medium, and 5–7 as Low risk.

Attack Trees

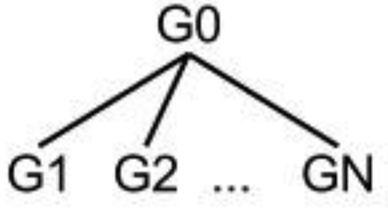
- Data structure to represent an attack
- Look at system from attackers point of
- view.
- The root node of the tree is the global goal of the attacker
- Children are refinements of this goal
- Nodes can be conjunctive (AND) or disjunctive (OR)

Notations for nodes

- Can be represented graphically or textually
- Conjunctive (AND) node
  - To achieve G0, you must achieve G1 AND G2 ... AND GN

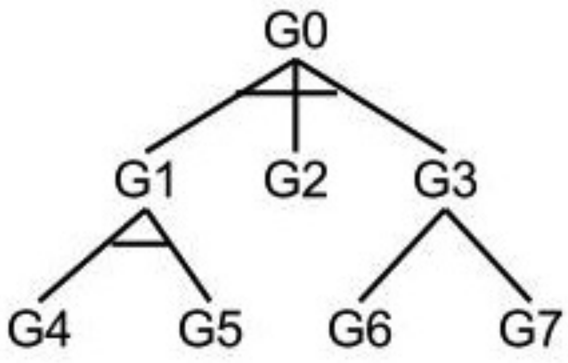


- Disjunctive (OR) node
  - To achieve G0, you must achieve G1 OR G2 ... OR GN



Attack Trees

- Attack trees consist of any combination of conjunctive and disjunctive nodes.
- Individual intrusion scenarios are created by depth first traversal.

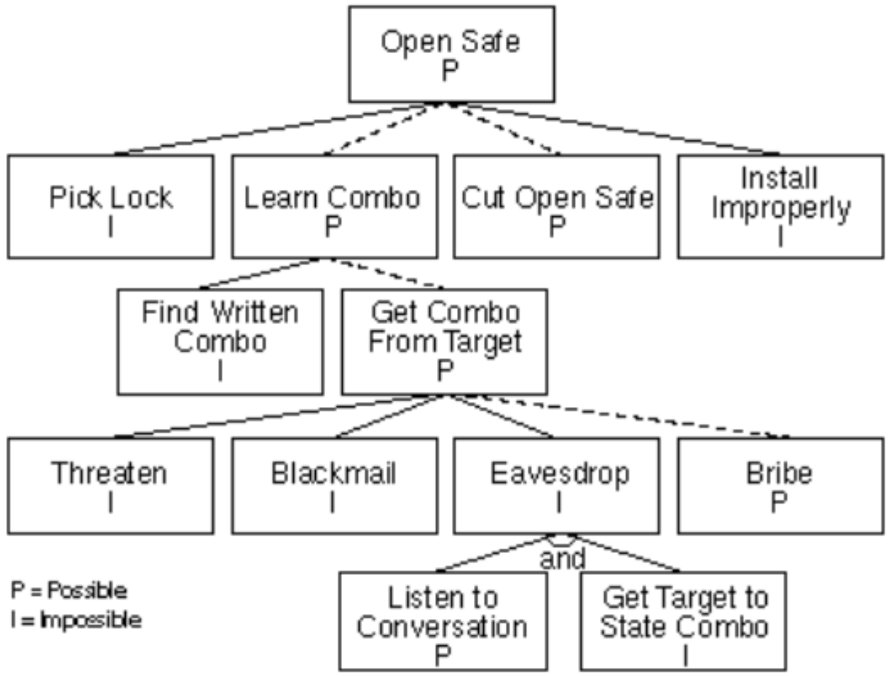


So the tree to the left leads to the attack scenarios:

<G4, G5, G2, G6>
<G4, G5, G2, G7>

Attributes: Boolean

- You can assign attributes to nodes in the tree to help you reason about them
  - Can be useful in understanding what sorts of attackers can launch certain attacks
- “Possible” and “Impossible” are one way to assign attributes to the tree

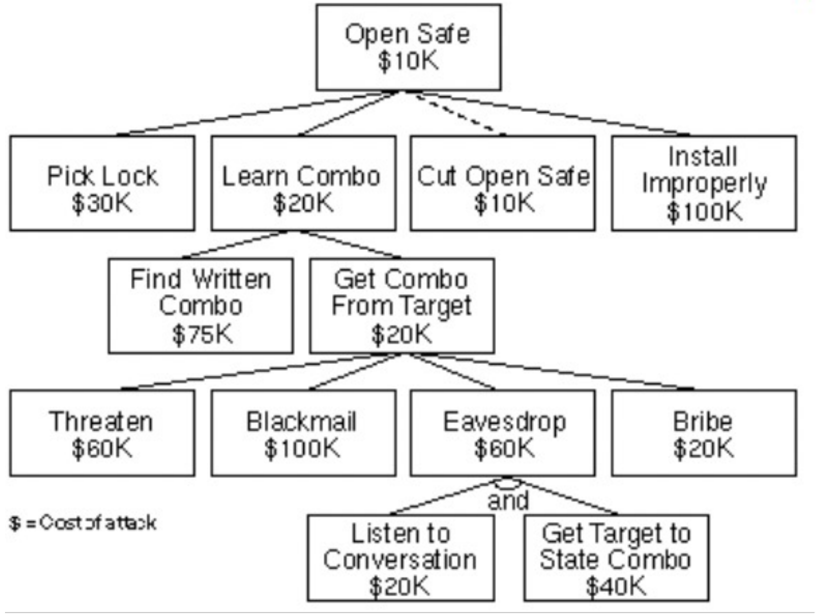


Attributes: Boolean

- “Possible” and “Impossible” are only one way to assign attributes to the tree
- Any Boolean value can be assigned to the leaf nodes and then propagated up the tree structure: AND/OR of the children node values
  - Easy vs. hard
  - Expensive vs. inexpensive
  - Legal vs. illegal
  - Special equipment Vs no special equipment

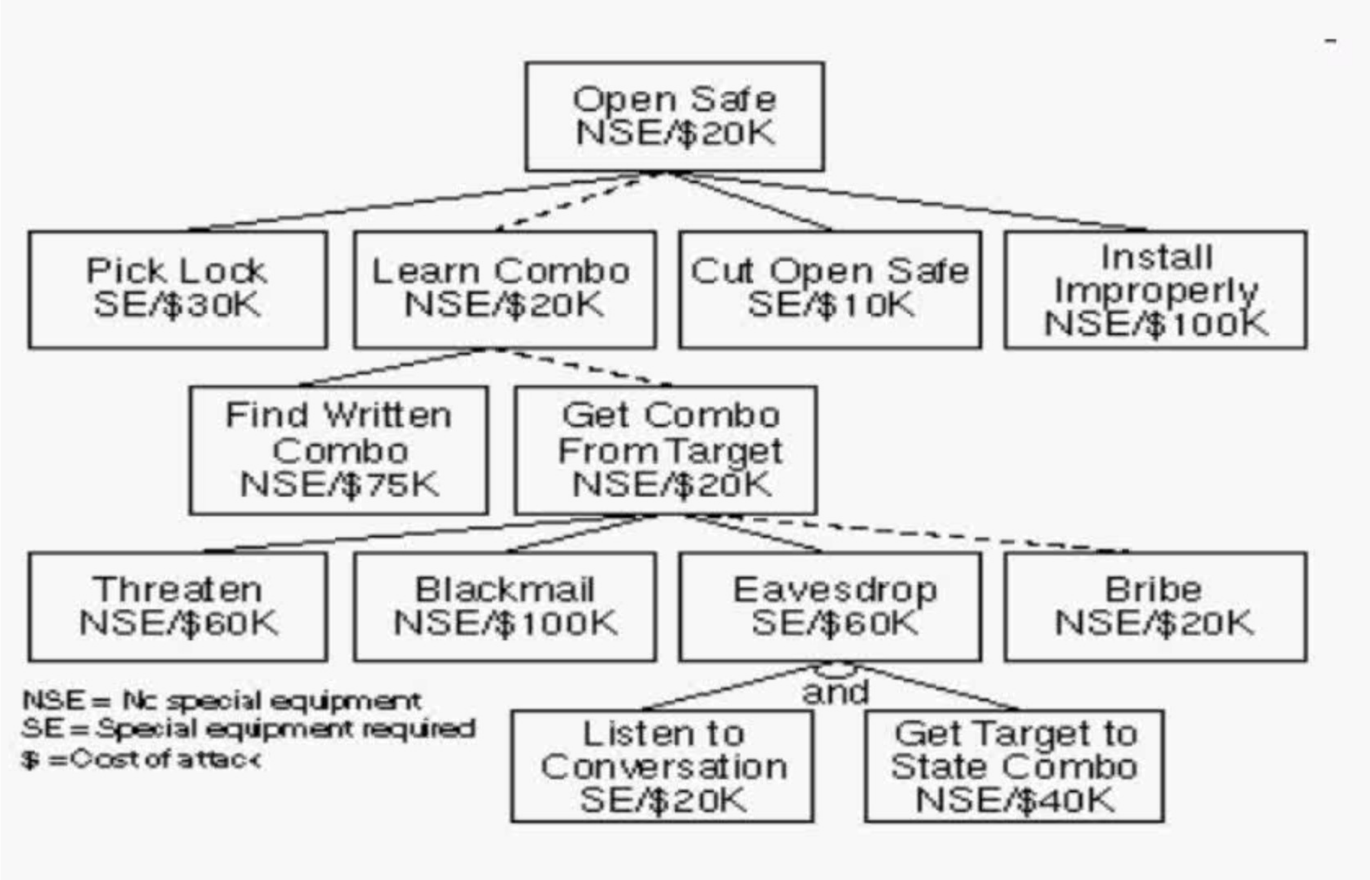
Attributes: Continuous

- Expensive vs. Inexpensive is fine, but good to say the amount, e.g.
- Continuous values can also be assigned to the nodes of the attack tree, and can be propagated up the tree
  - OR nodes have the value of their cheapest child
  - AND nodes have the value of the sum of their children



Combination of attributes

- Cheapest Attack with no Special Equipment



Example 1

- Example: Given a battlefield communications system. The related CIA asset is the __________ of the system, and the impact of a failure is __________.

- Example: Given a battlefield communications system. The related CIA asset is the availability and integrity of the system, and the impact of a failure is loss of life.

Example 2

- Example: Given a system that uses personal information such as name, SSN, etc. The related CIA asset at risk is the __________ of that information, and the impact of a compromise is the potential for __________.

- Example: Given a system that uses personal information such as name, SSN, etc. The related CIA asset at risk is the confidentiality of that information, and the impact of a compromise is the potential for identify theft.

Risk Assessment

- Assessment: measures of the impact of an event, and the probability of an event (threat agent exploiting a vulnerability)
- Quantitative (objective) and Qualitative (subjective) approaches both used.
- Quantitative approach:
  - Compute expected monetary value (impact) of loss for all “events”
  - Compute the probability of each type of expected loss
- Qualitative approach: use Low, Medium, High; ratings; other categorical scales

Risk Management

- Once you have risk computed for each threat you can prioritize them and for each do one of the following:
  - Accept the risk - The risk is so low or so costly to mitigate that it is worth accepting.
  - Transfer the risk - Transfer the risk to somebody else via insurance, warnings etc.
  - Reduce the risk - Remove the system component or feature associated with the risk if the feature is not worth the risk.
  - Mitigate the risk - Reduce the risk with countermeasures.
- The understanding of risks leads to policies, specifications and requirements.
- Appropriate security mechanisms are then developed and implemented, and then deployed