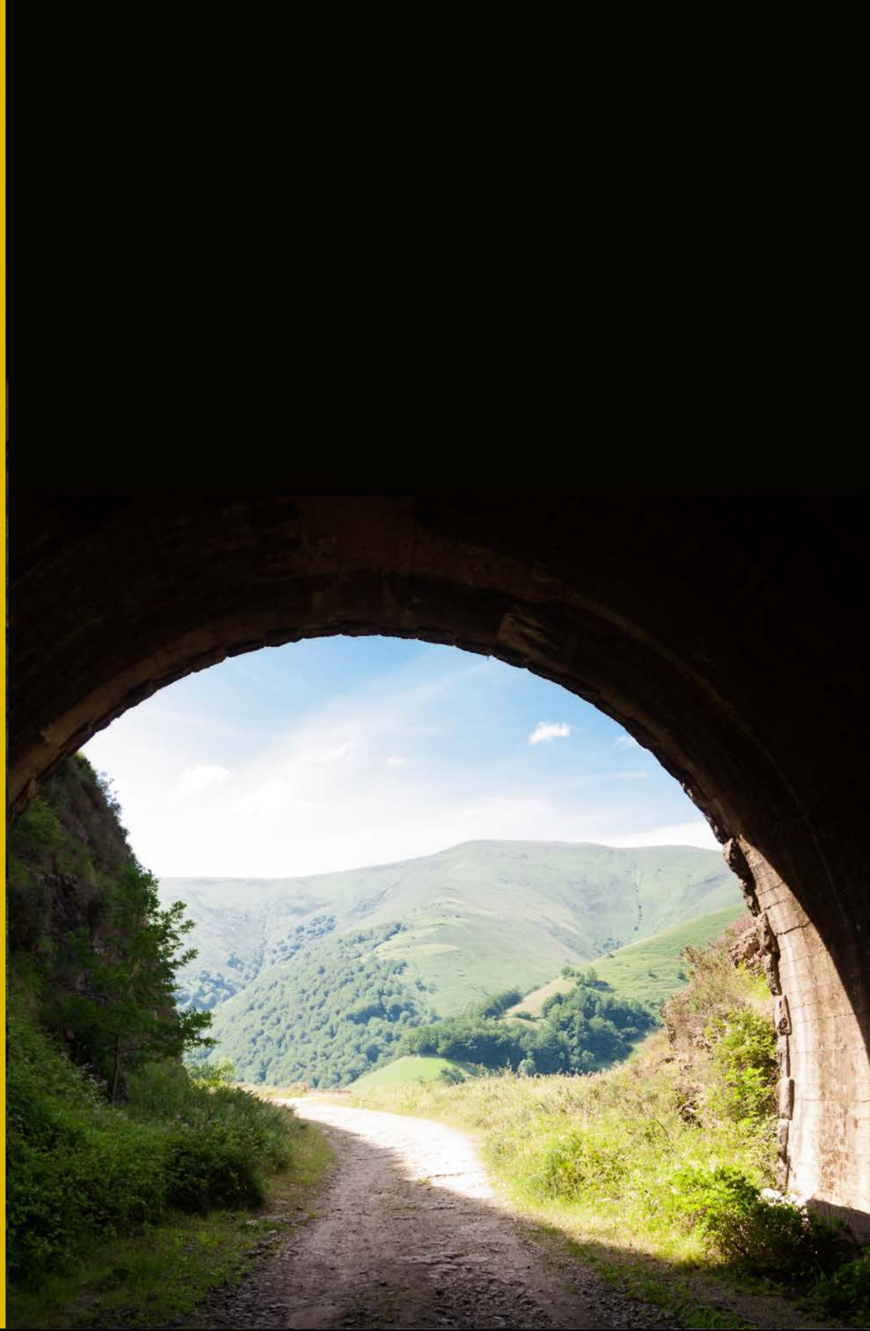
# Advanced Threat Modeling (ATM)

Nancy R. Mead, PhD

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA  15213



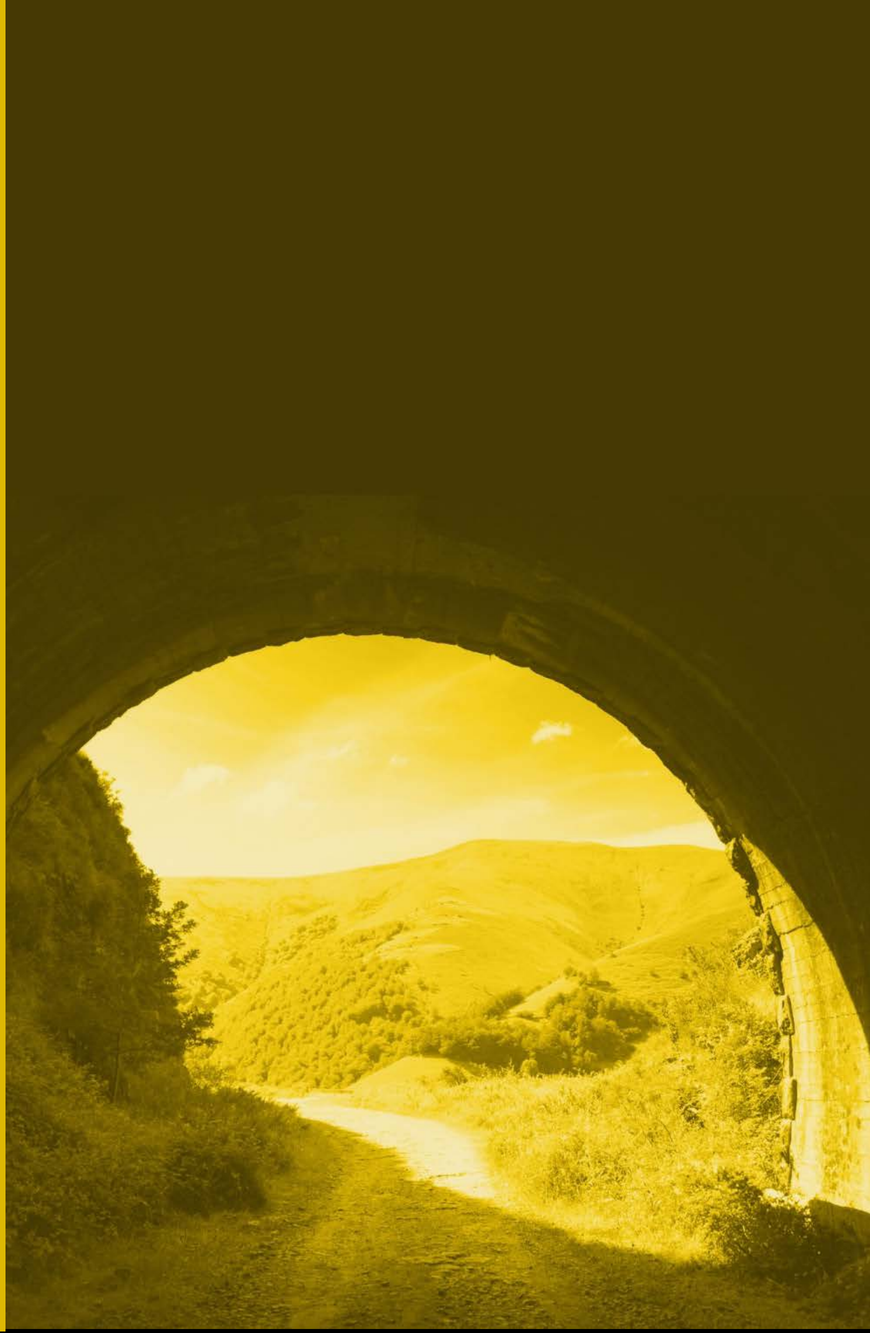
This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

DM17-0724

# ATM Introduction



# Learning Outcomes

After completing this course, students will be able to

- Understand how threat modeling fits into the security development lifecycle
- Apply any of the 4 threat modeling methods discussed to their systems.
- Study and assess new threat modeling methods for applicability to their systems

# Agenda

**What is Threat Modeling?**

**Threat Modeling Methods**

- Security Cards
- Personas and Persona Non Grata
- STRIDE
- Hybrid Threat Modeling Method

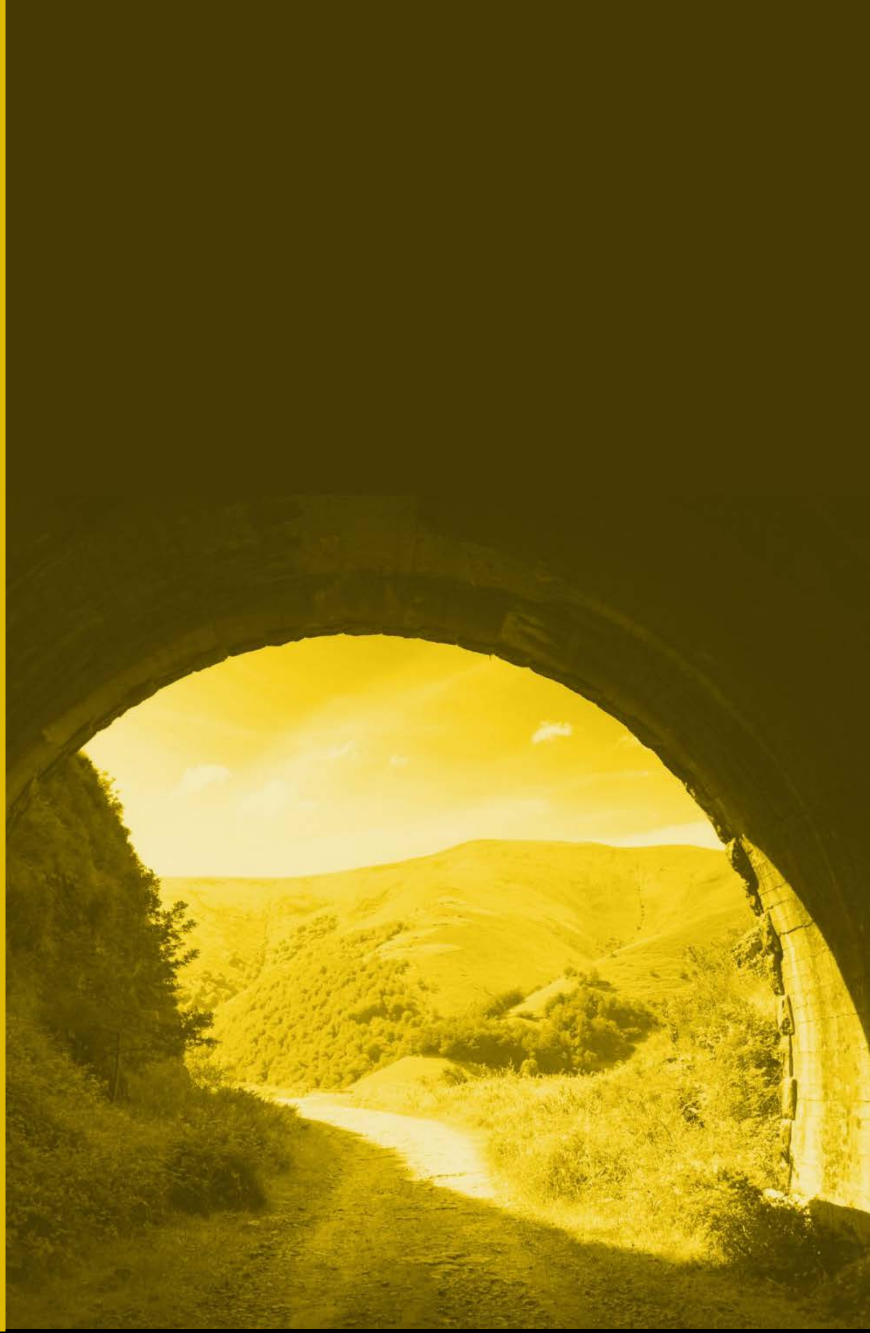
**Case Studies**

- Drone Swarm (UAV)
- Implantable Cardioverter Defibrillator (ICD)
- Aircraft Maintenance

**Research work and other methods**

**Summary**

# What is Threat Modeling?



# Definitions

**Threat modeling** is a process by which potential threats can be identified, enumerated, and prioritized – all from a hypothetical attacker's point of view.

Wikipedia https://en.wikipedia.org/wiki/Threat_model

A **threat modeling method (TMM)** is an approach for creating an abstraction of a software system, aimed at identifying attackers' abilities and goals, and using that abstraction to generate and catalog possible threats that the system must mitigate.

# Who Does Threat Modeling?

Vendors such as Microsoft

- Microsoft developed and uses STRIDE and makes it freely available

Government organizations such as DoD

- Mandated for DoD
- Various methods in use, some are based on NIST standards, some use checklists.

Commercial organizations such as automotive industry, finance, and so on

- Various methods in use, including STRIDE, risk analysis approaches such as OCTAVE, attack trees, etc.

BSIMM identified Attack Models as a Level 1 practice

https://www.bsimm.com/framework/intelligence/attack-models/

# Security is not binary.

- **Not** secure vs. insecure
- Perfect security does **not** exist.
- “Secure against attacker *A* with resources *R* with probability *P* under conditions *C*”

# Security Vulnerability

**Definition:** a weakness which allows an attacker* to bypass security controls

Requires three elements:

- a system susceptibility or flaw,
- attacker access to the flaw, and
- attacker capability to exploit the flaw

Threat modeling can be used to identify potential security vulnerabilities in the software design

* Security Hacker. *Wikipedia*. July 2017. https://en.wikipedia.org/wiki/Security_hacker

# Threat Modeling: Schools of Thought

- What are the artifacts/goals?
- Who does it?
- When in the process?

Is the process driven by:

- adversaries and their goals?
- system assets?
- system design?
- system implementation?

**Specificity?**

**Flexibility?**

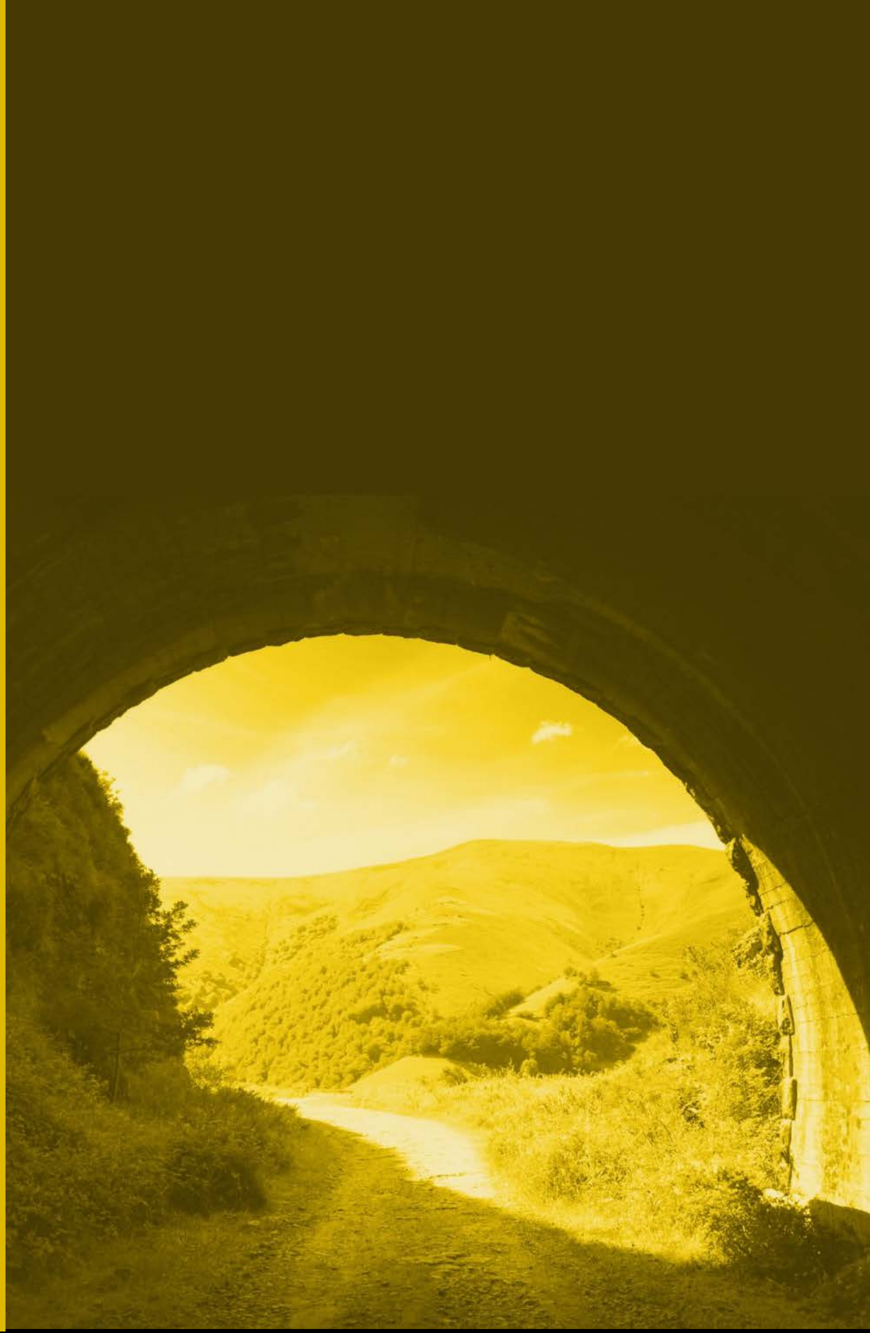
**Automation?**

**Expertise?**

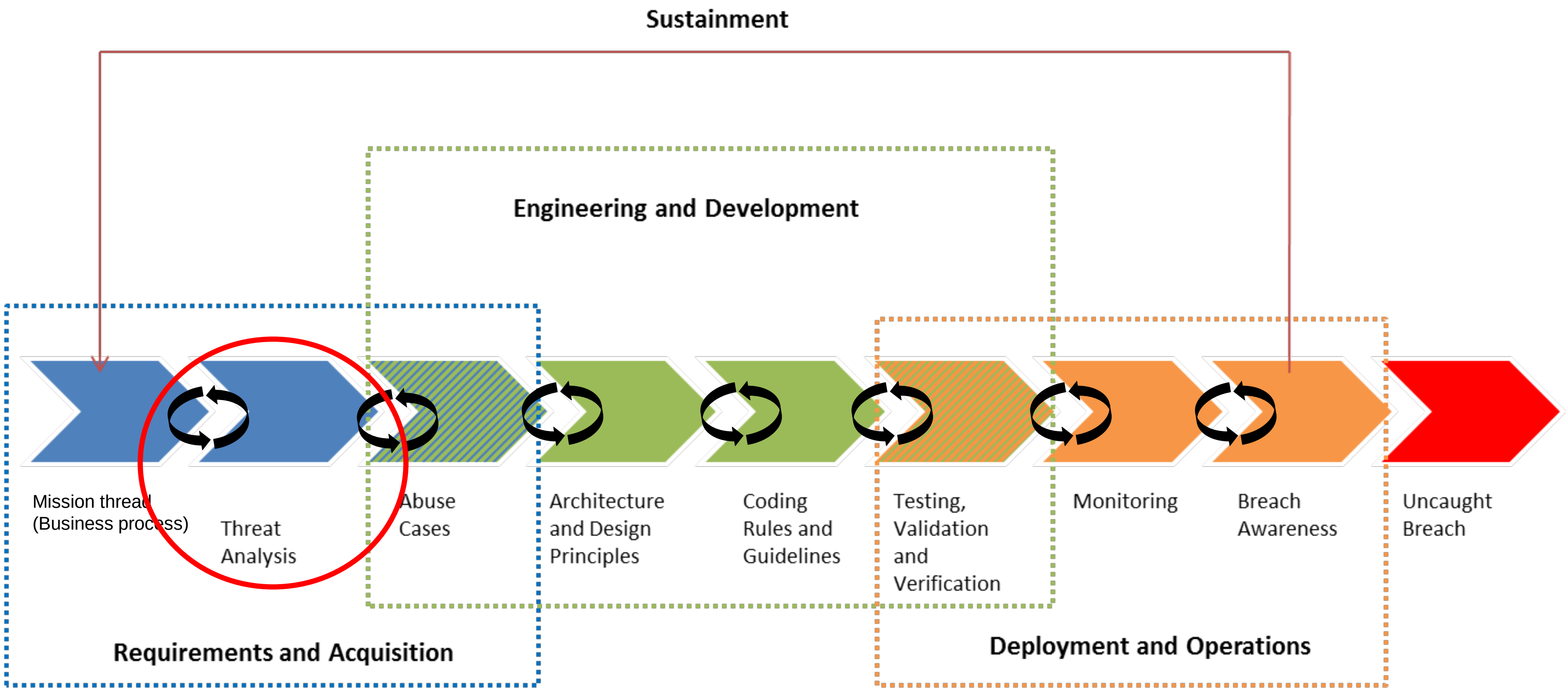
**Creativity?**

**Cost?**

# Threat Modeling in the Lifecycle



# Cybersecurity Is a Lifecycle Challenge


Sustainment
Engineering and Development
Mission thread (Business process)
Threat Analysis
Abuse Cases
Architecture and Design Principles
Coding Rules and Guidelines
Testing, Validation and Verification
Monitoring
Breach Awareness
Uncaught Breach
Requirements and Acquisition
Deployment and Operations

Design Weaknesses

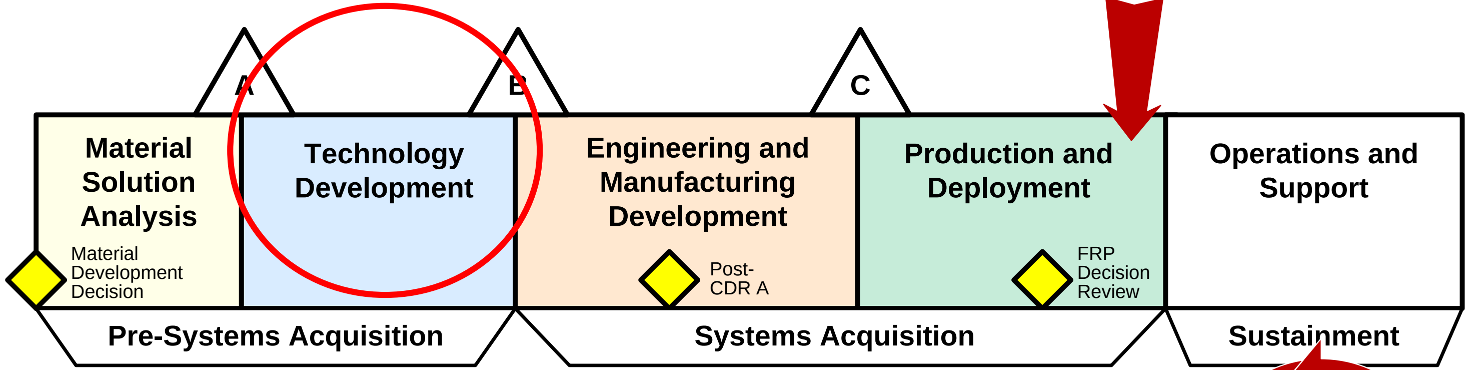
Coding Weaknesses

Implementation Weaknesses

# Software Assurance Landscape: System Lifecycle

**Threat modeling informs the selection of security controls**

Certification and Authorization to Operate


A

B

C

Material Solution Analysis

Material Development Decision

Technology Development

Engineering and Manufacturing Development

Post-CDR A

Production and Deployment

FRP Decision Review

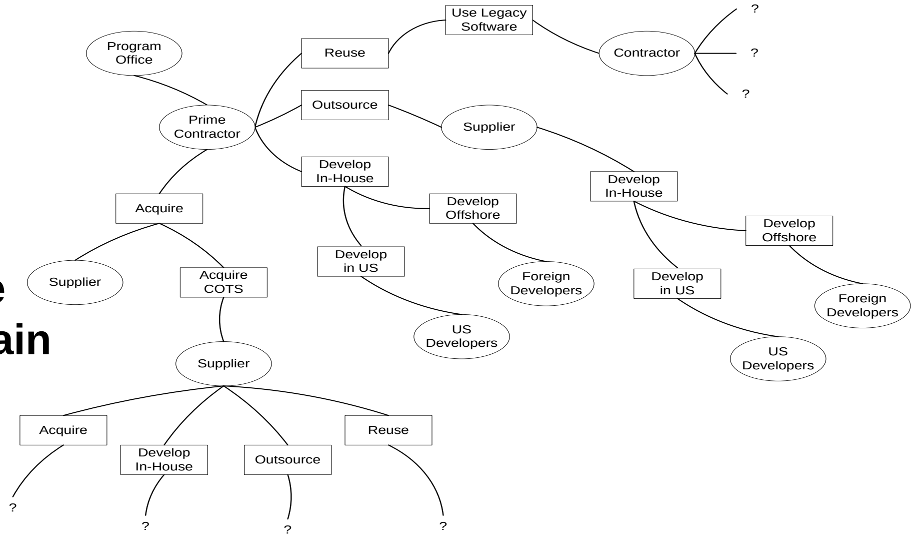
Operations and Support

Pre-Systems Acquisition

Systems Acquisition

Sustainment

**Software Supply Chain**


Program Office

Prime Contractor

Reuse

Use Legacy Software

Contractor

?

?

?

Outsource

Supplier

Develop In-House

Develop Offshore

Foreign Developers

Develop in US

US Developers

Develop In-House

Develop Offshore

Foreign Developers

Develop in US

US Developers

Acquire

Supplier

Acquire COTS

Supplier

Acquire

Develop In-House

Outsource

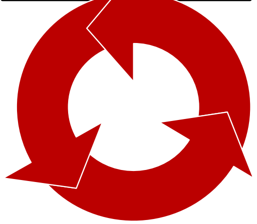
Reuse

?

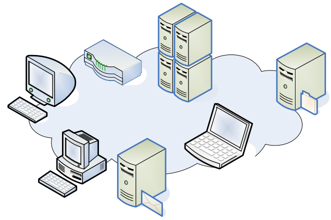
?

?

?

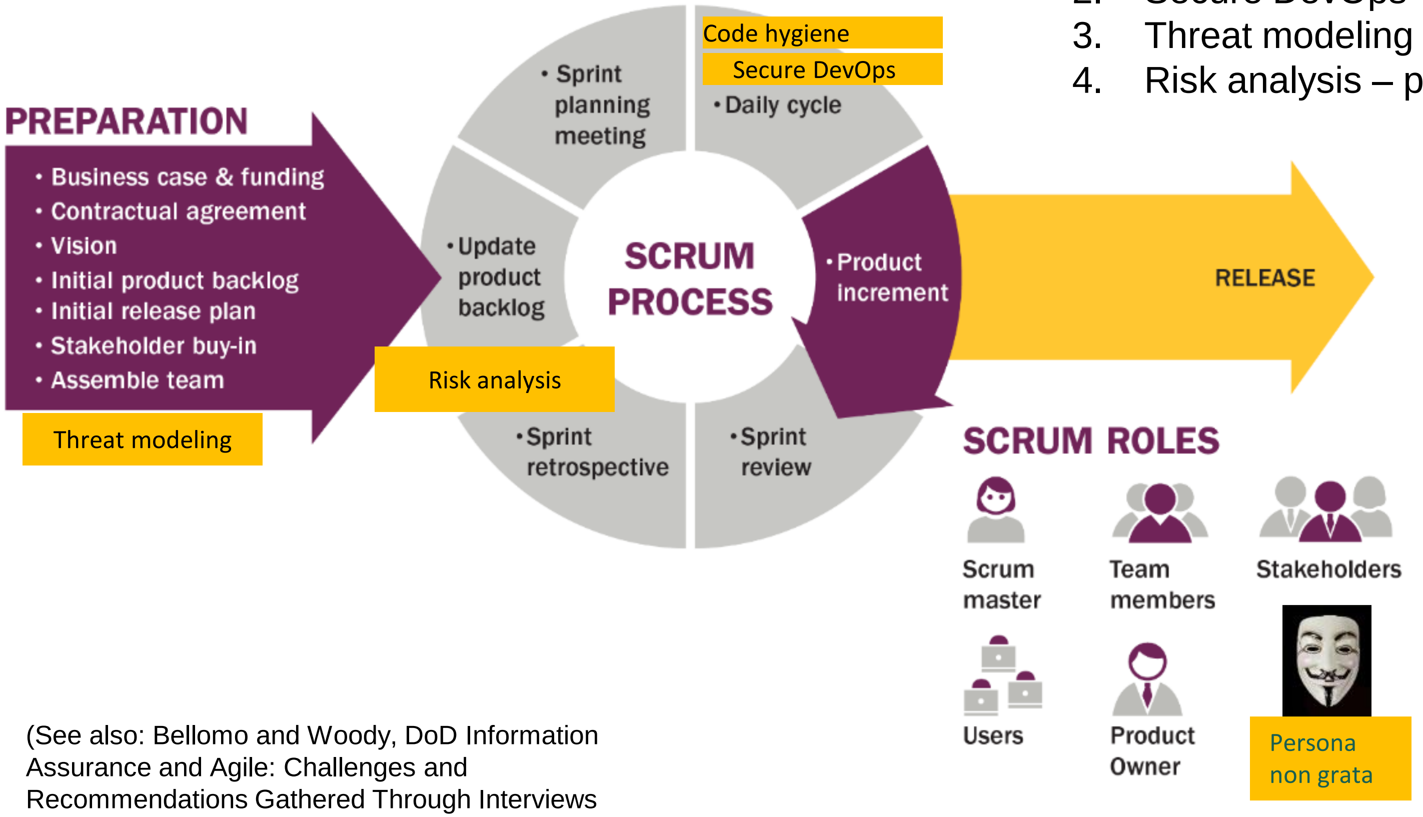


**Software Patch Cycle**



# Integrating security into Agile (Scrum) development

1. Code hygiene – introduce secure coding
2. Secure DevOps – include security tools
3. Threat modeling – represent a new role
4. Risk analysis – prioritize in backlog



PREPARATION

- Business case & funding
- Contractual agreement
- Vision
- Initial product backlog
- Initial release plan
- Stakeholder buy-in
- Assemble team

Threat modeling

SCRUM PROCESS

- Sprint planning meeting
- Daily cycle
- Product increment
- Sprint review
- Sprint retrospective
- Update product backlog

Code hygiene

Secure DevOps

Risk analysis

RELEASE

SCRUM ROLES

Scrum master

Team members

Stakeholders

Users

Product Owner

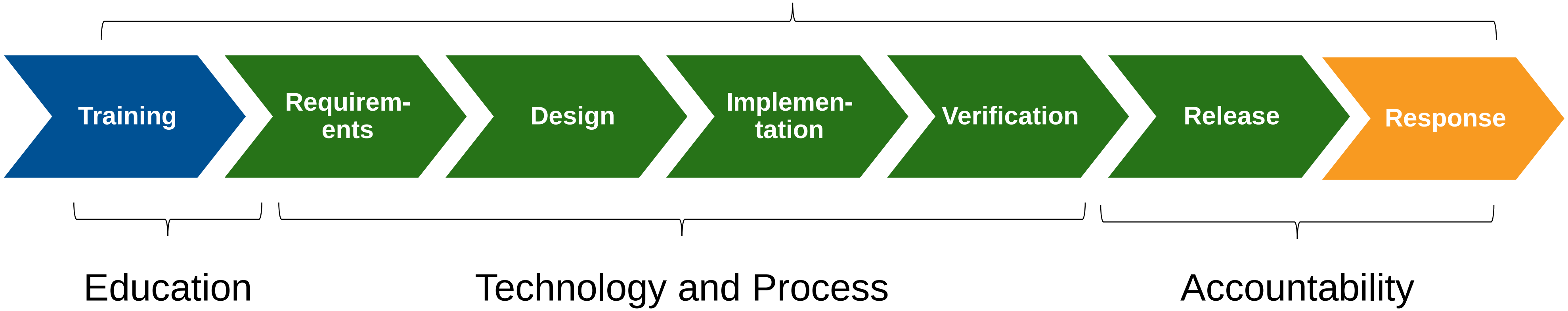
Persona non grata

(See also: Bellomo and Woody, DoD Information Assurance and Agile: Challenges and Recommendations Gathered Through Interviews with Agile Program Managers and DoD Accreditation Reviewers (http://repository.cmu.edu/cgi/viewcontent.cgi?article=1674&context=sei)

# Microsoft Security Development Lifecycle (SDL)

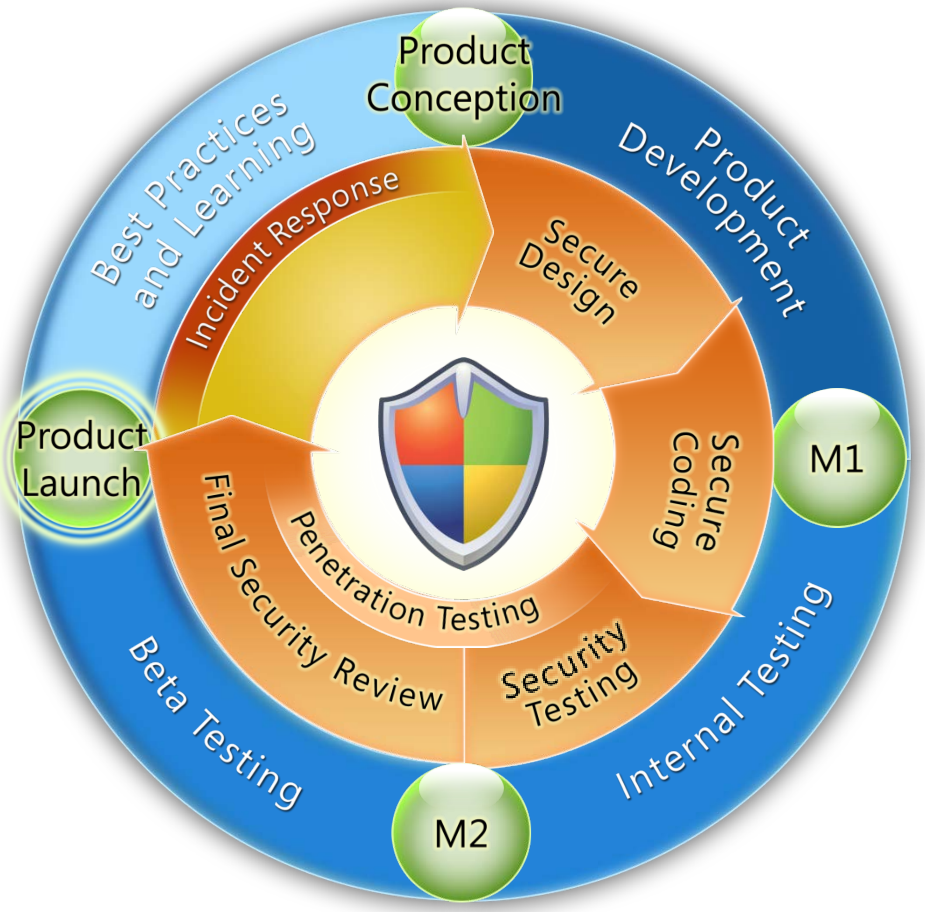
**Delivering secure software requires:**

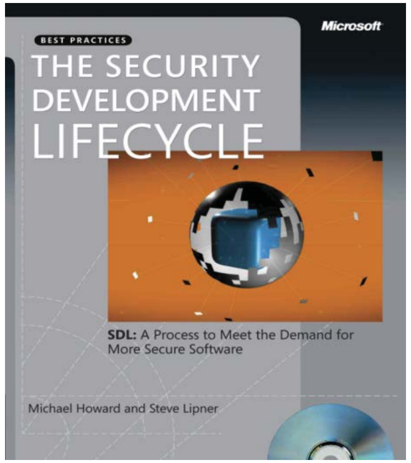
Executive commitment → SDL a mandatory policy at Microsoft since 2004



**Ongoing Process Improvements → 6 month cycle**

# Microsoft Security Development Lifecycle (SDL)




BEST PRACTICES

THE SECURITY DEVELOPMENT LIFECYCLE

Microsoft

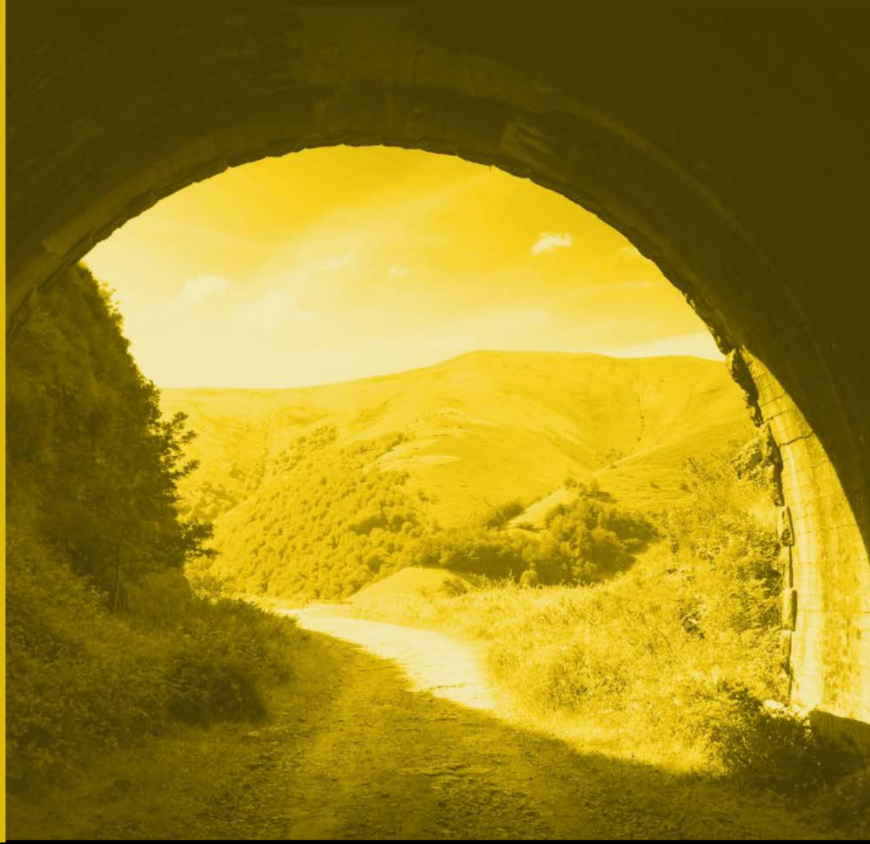
**SDL:** A Process to Meet the Demand for More Secure Software

Michael Howard and Steve Lipner

**SDL Book:**
http://www.microsoft.com/mspress/books/8753.aspx

**Official SDL Web Site:** http://www.microsoft.com/sdl

# Why We Do Threat Modeling: E-voting Case Study

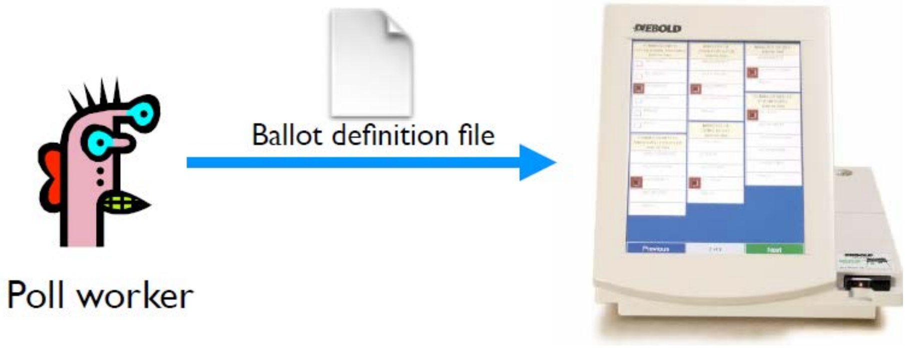


# Example: Electronic Voting

- popular replacement for traditional paper ballots

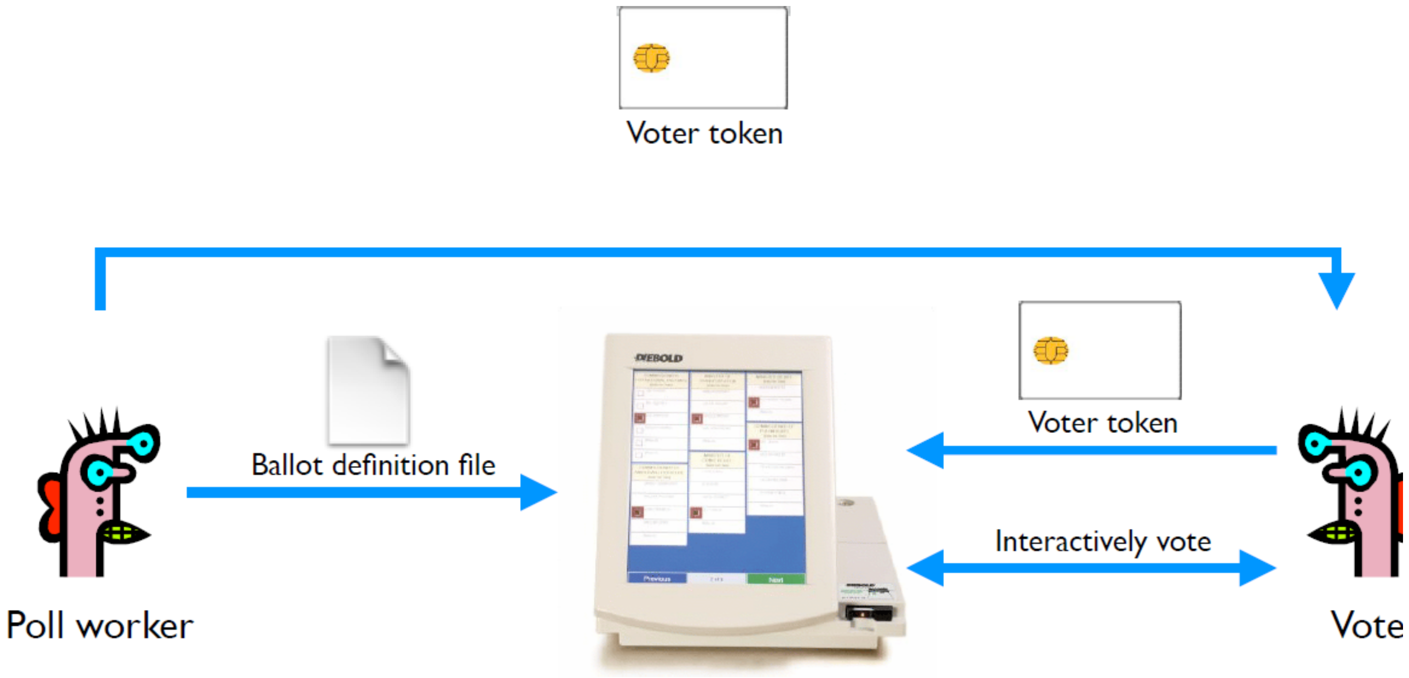
# Pre-Election

Poll workers load “ballot definition files” on voting machine.



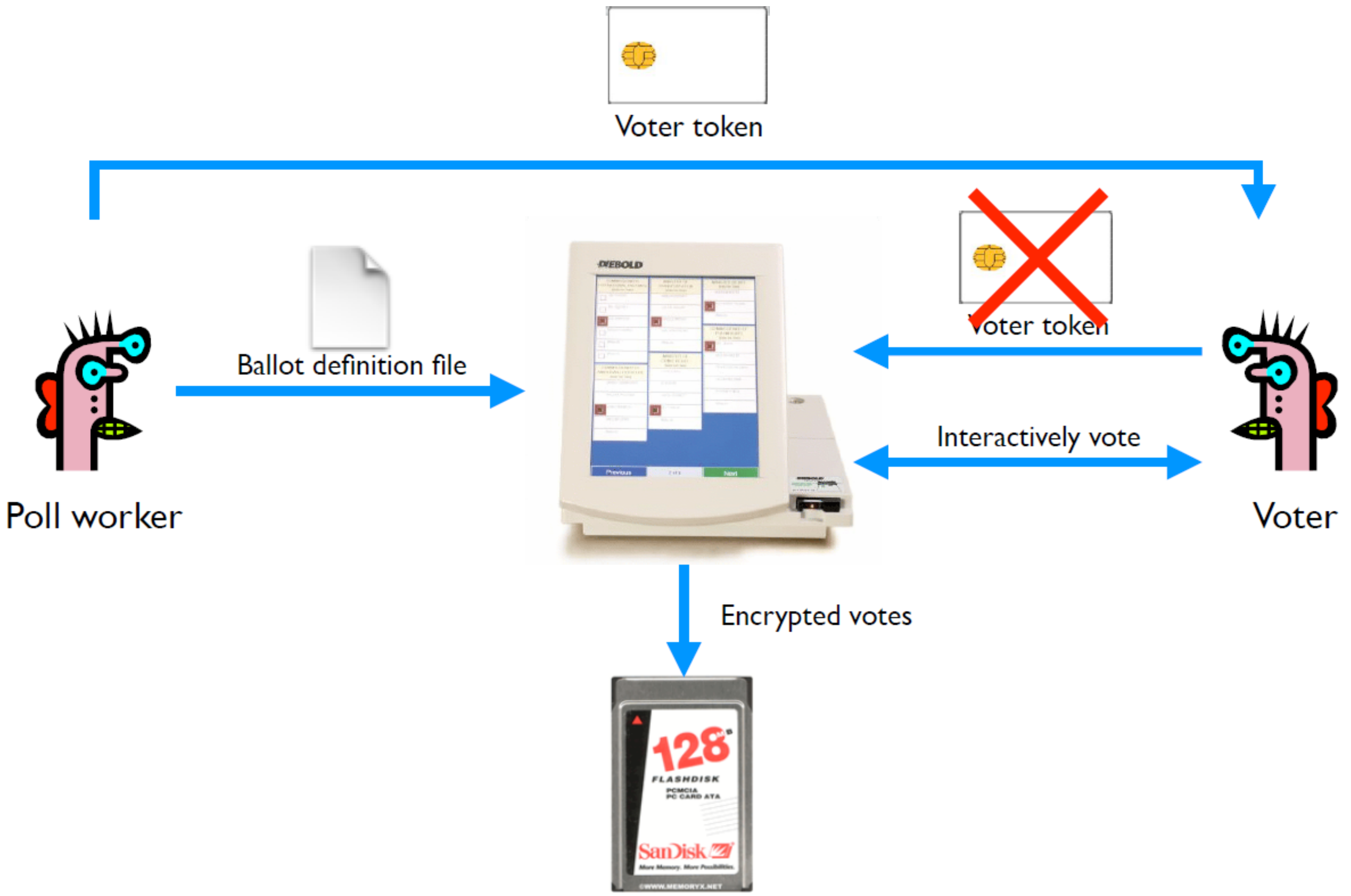
# Active Voting

Voters obtain **single-use** tokens from poll workers. Voters take tokens to **active machines** and vote.



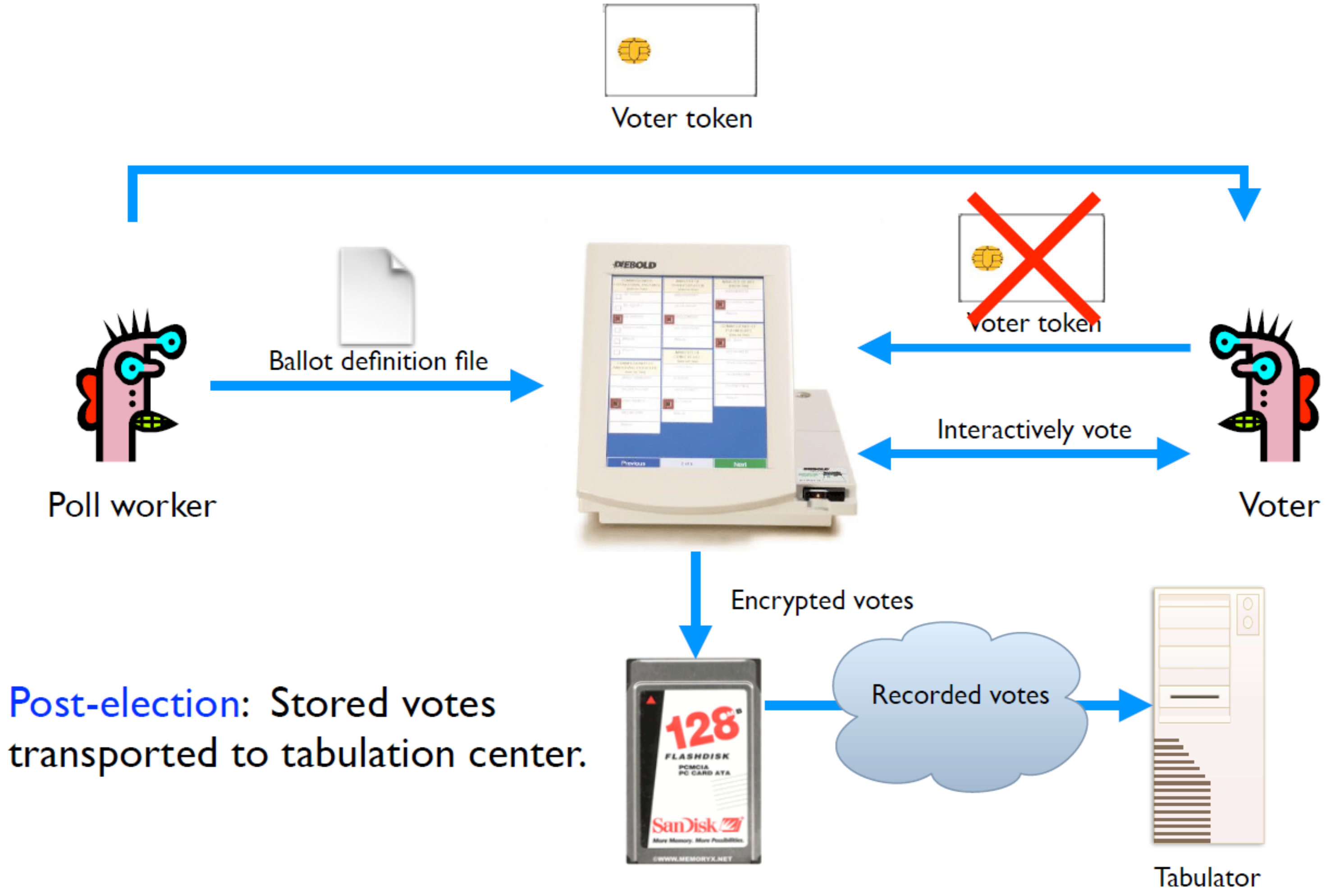
# Active Voting

Votes are encrypted and stored. Voter token is canceled.



# Post-Election

Stored votes are transported to the tabulation center.



Post-election: Stored votes transported to tabulation center.

# Security and E-Voting (Simplified)

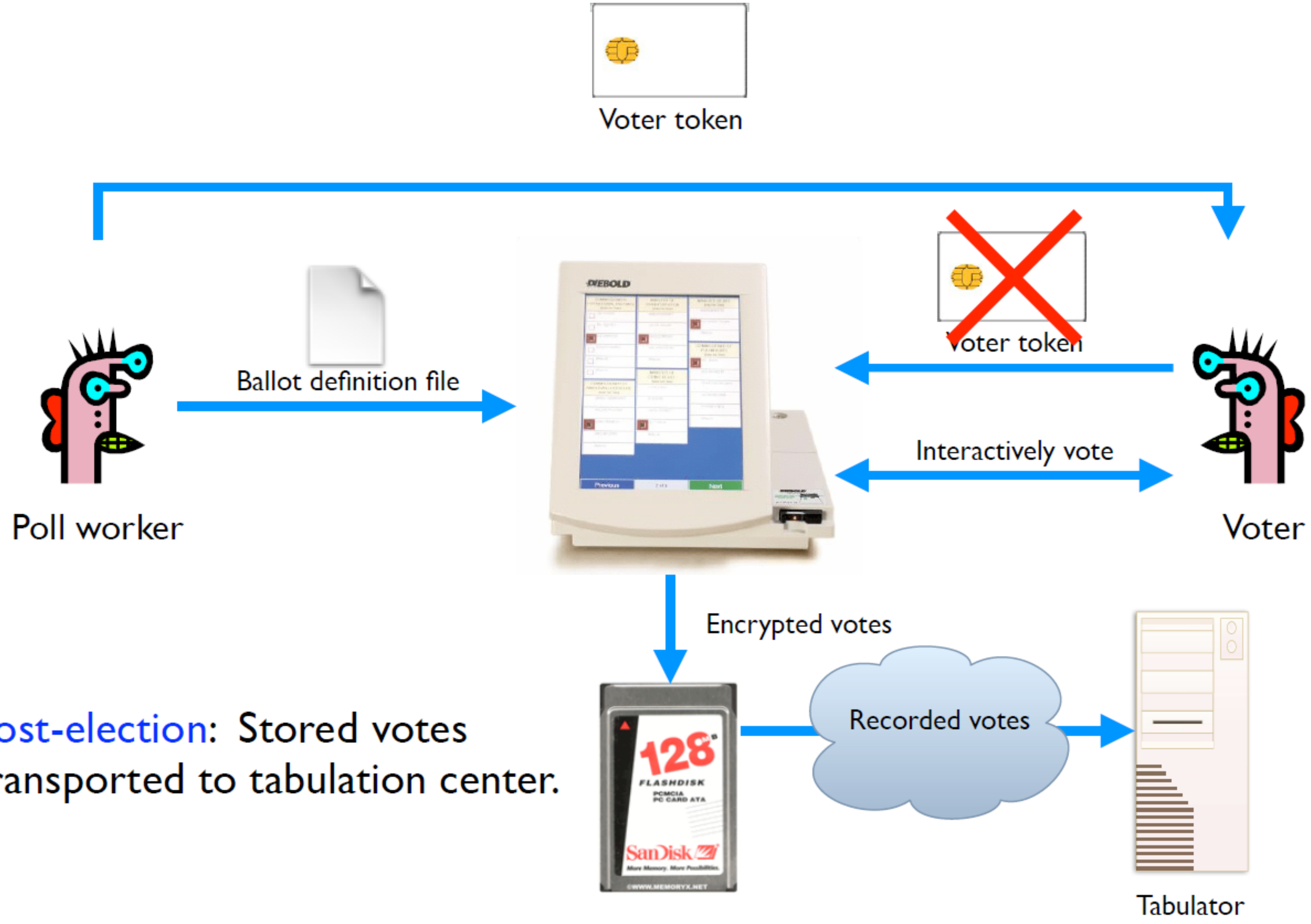
Functionality goals:

- Easy to use
- People should be able to cast votes easily, in their own language or with headphones for accessibility.

Security goals:

- Adversary should not be able to tamper with the election outcome
  - by changing votes
  - by denying voters the right to vote
- Adversary should not be able to figure out how voters vote

# Can you spot any potential issues?



Post-election: Stored votes transported to tabulation center.

# Potential Adversaries

- Voters
- Election officials
- Employees of voting machine manufacturer
  - Software/hardware engineers
  - Maintenance people
- Other engineers
  - Makers of hardware
  - Makers of underlying software or add-on components
  - Makers of compiler
- ...
- Or any combination of the above

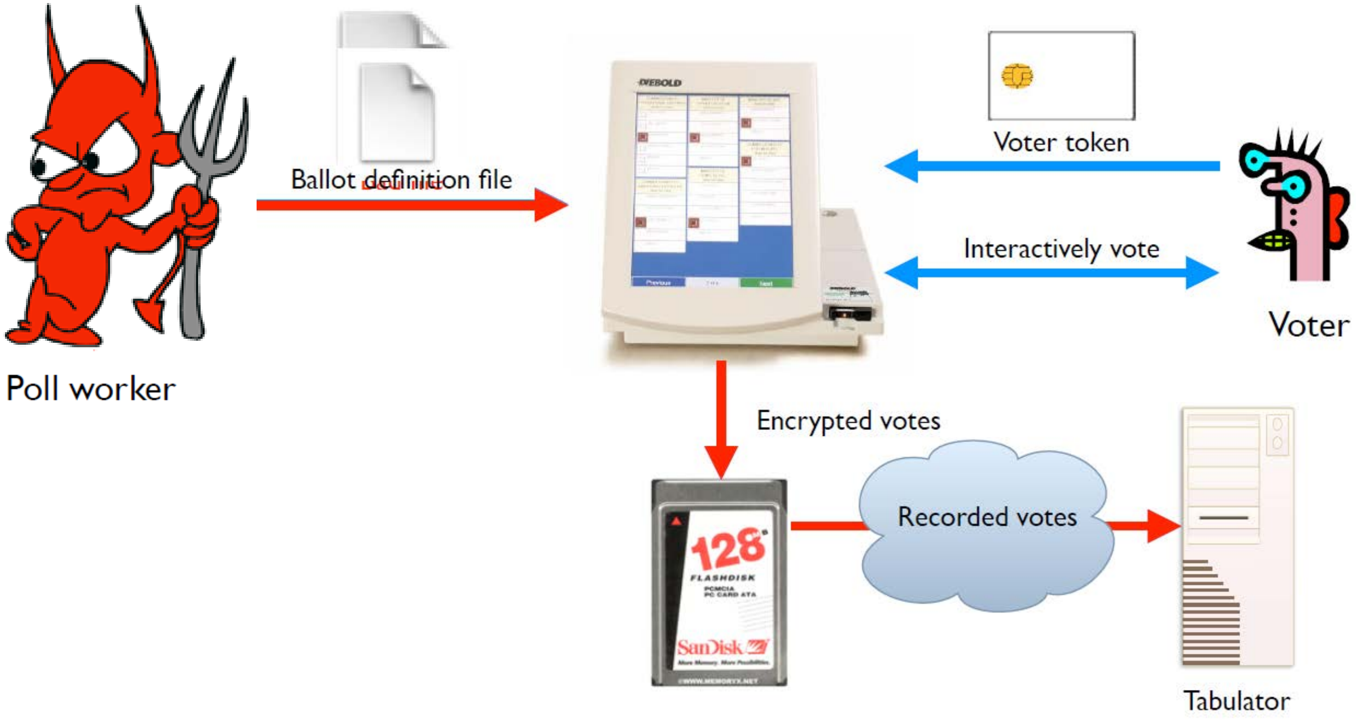
# What software is running?

**Problem:** An **adversary** (e.g., a poll worker, software developer, or company representative) **able to control the software** or the underlying hardware **could do whatever he or she wanted**.

# Other Problems

Ballot **definition files** are **not authenticated.**

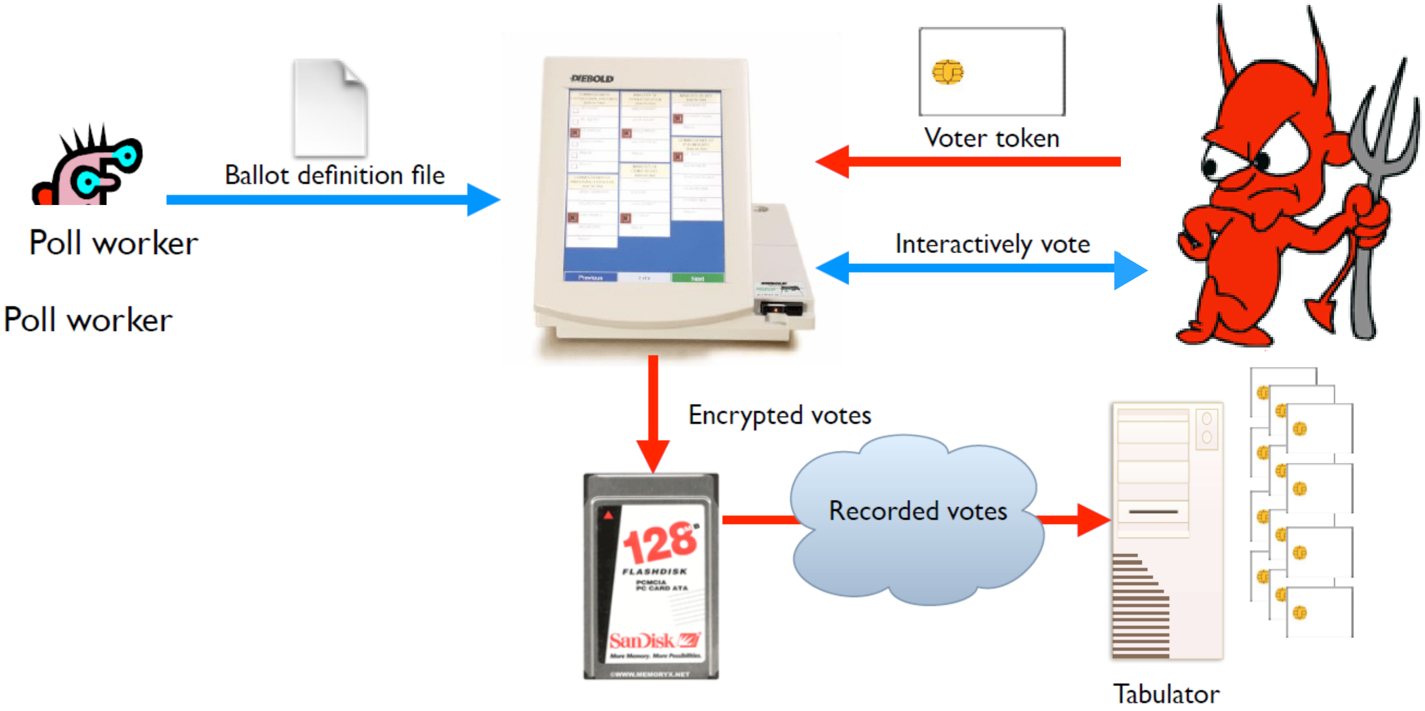
**Example attack:** A **malicious poll worker** could modify ballot definition files so that **votes cast for "Mickey Mouse"** are **recorded for "Donald Duck."**



# Other Problems

Smartcards can perform cryptographic operations. But there is **no authentication from voter token to terminal**.

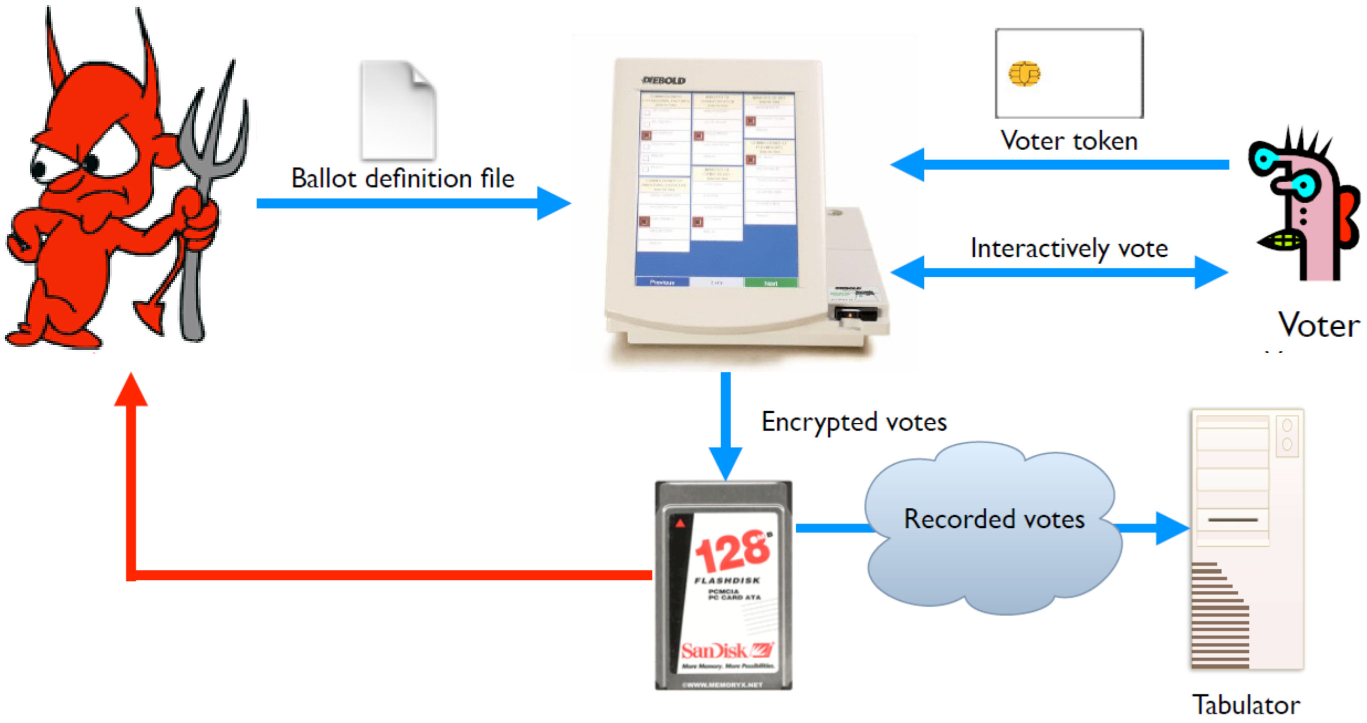
**Example attack:** A regular voter could make his or her own voter token and **vote multiple times**.



# Other Problems

**Encryption key** (“F2654hD4”) **hard-coded** into the software since (at least) 1998. Votes stored in the order cast.

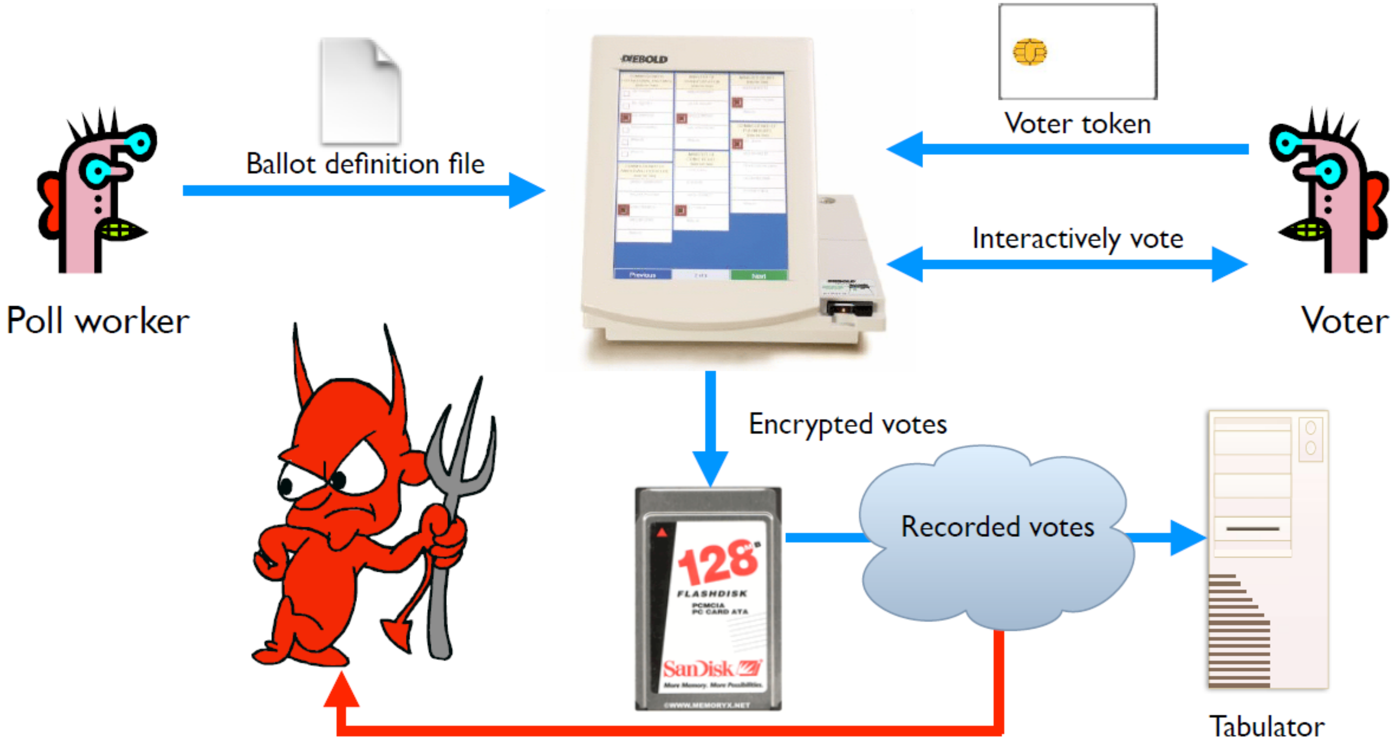
**Example attack:** A poll worker could **determine how voters vote**.



# Other Problems

When votes transmitted to tabulator over the Internet or a dialup connection, they are **decrypted first**; the cleartext results are sent to the tabulator.

**Example attack:** A sophisticated outsider could determine how voters vote.



# Security Not Just for PCs

mobile sensing platforms

RFID

EEG Gaming

large displays

ambient displays

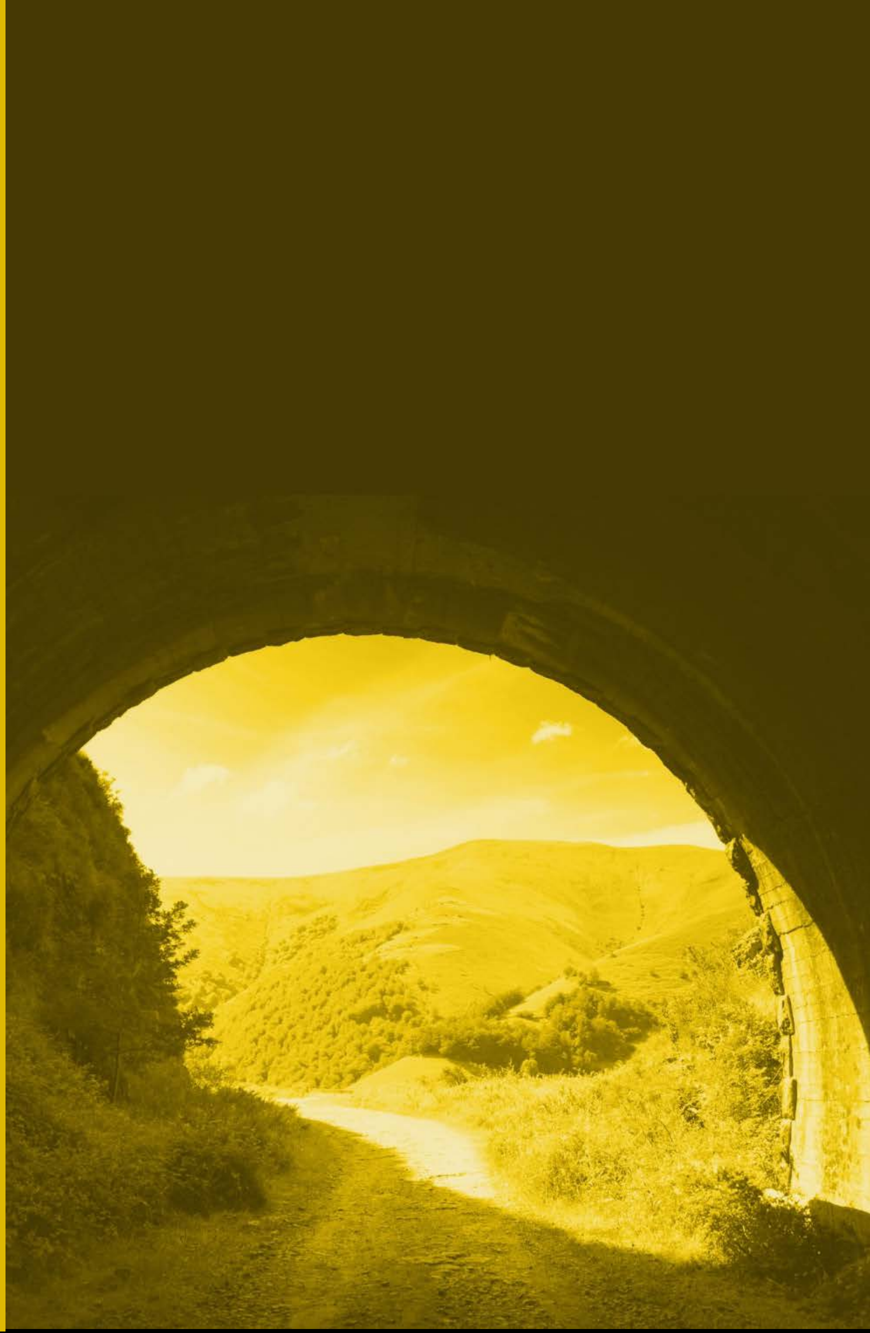
smart phones

wearables

health displays

Source: securitycards.cs.washington.edu

# Threat Modeling Methods: Security Cards



This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

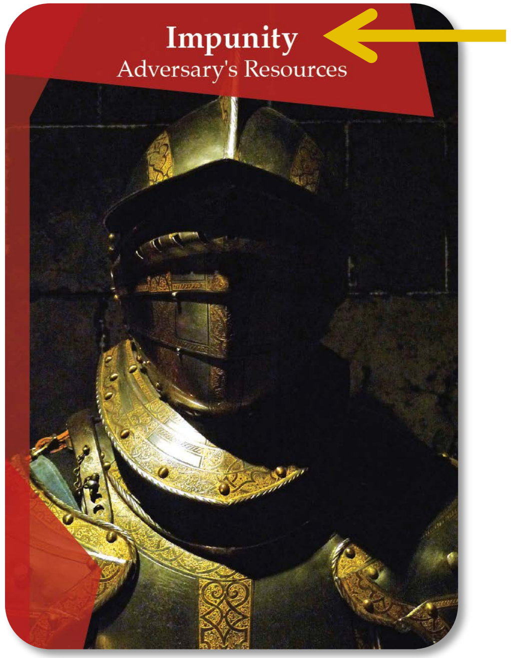
The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

DM17-0724

# Security Cards: A Threat Brainstorming Toolkit

Purpose: To facilitate the broad exploration of potential security and privacy threats to a system: the "security mindset"

# Example Card (Front)



Card topic

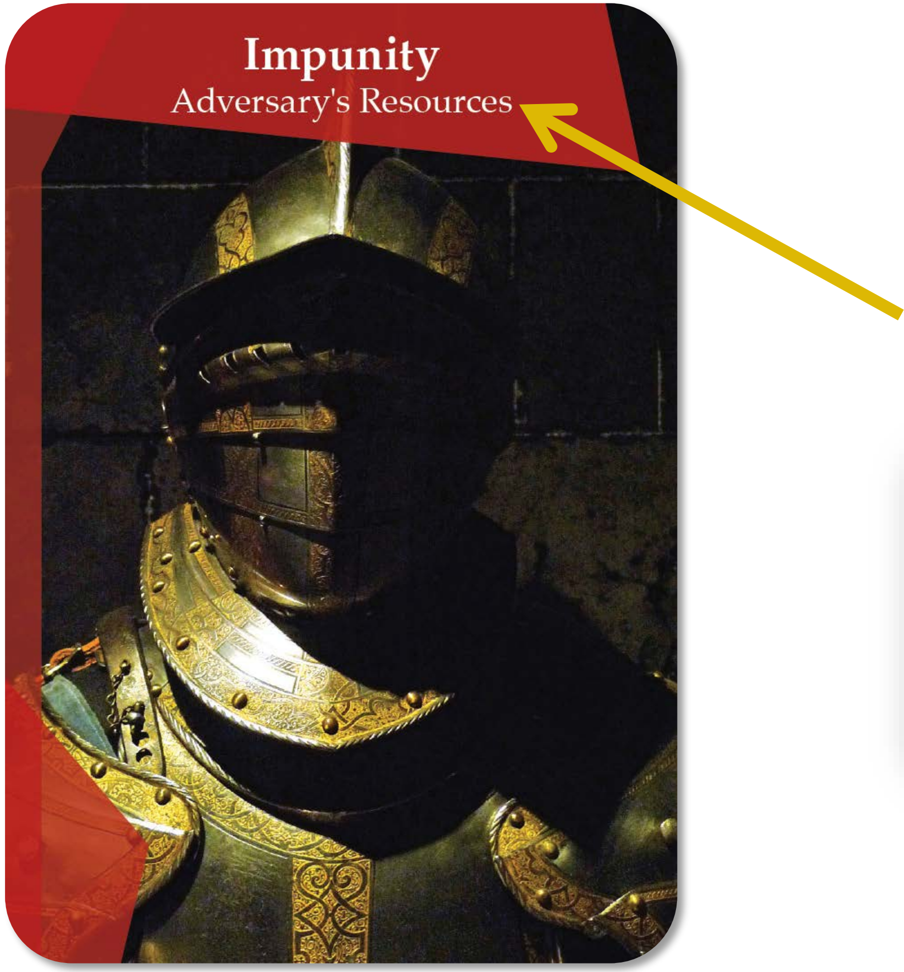
# Example Card (Front)

Impunity
Adversary's Resources

Card topic

Card dimension

# Example Card (Front)



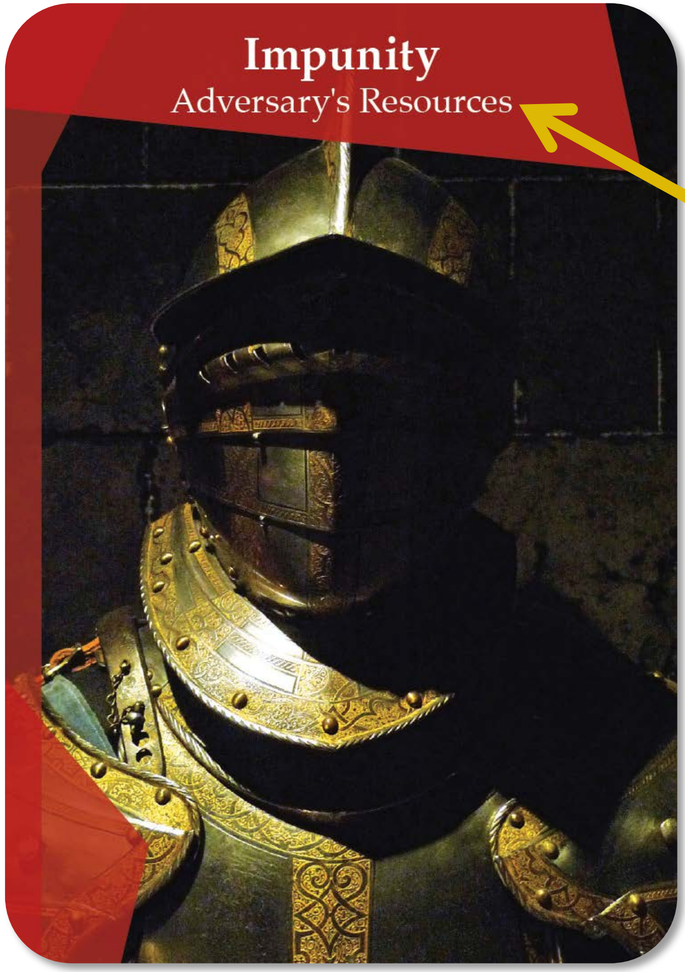
Card topic

Card dimension

Diplomacy or Warfare
Adversary's Motivations
Inside Capabilities
Adversary's Resources
Notice
Employees
Only
Indirect Attack
Adversary's Methods
Emotional Wellbeing
Human Impact

Adversary's Motivations
Adversary's Resources
Adversary's Methods

# Example Card (Front)



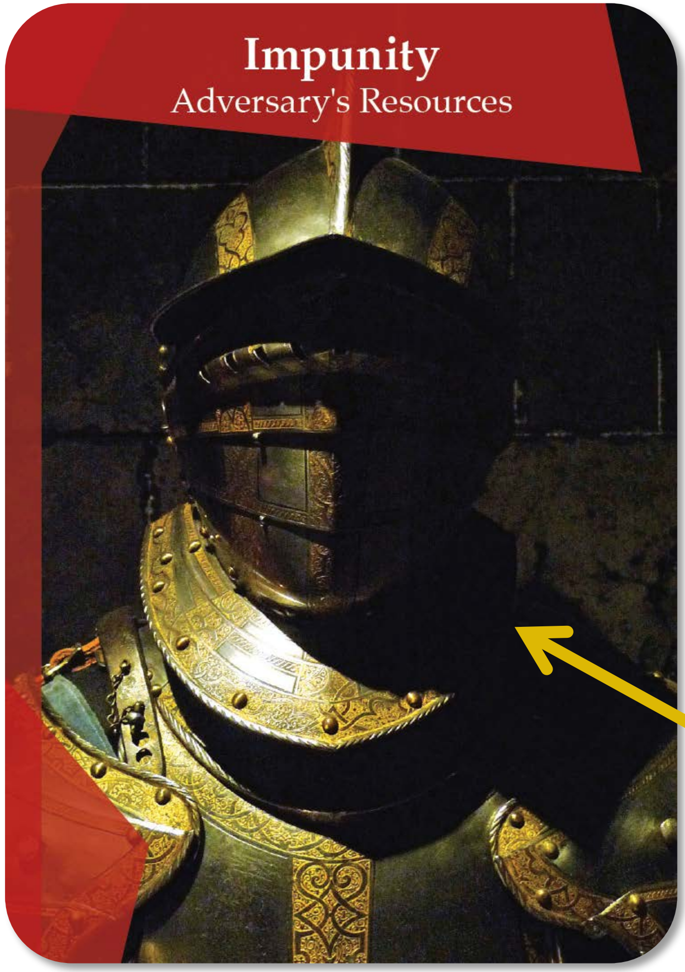
Card topic

Card dimension

Diplomacy or Warfare
Adversary's Motivations
Inside Capabilities
Adversary's Resources
Notice
Employees Only
Indirect Attack
Adversary's Methods
Emotional Wellbeing
Human Impact

Human
Impact

# Example Card (Front)



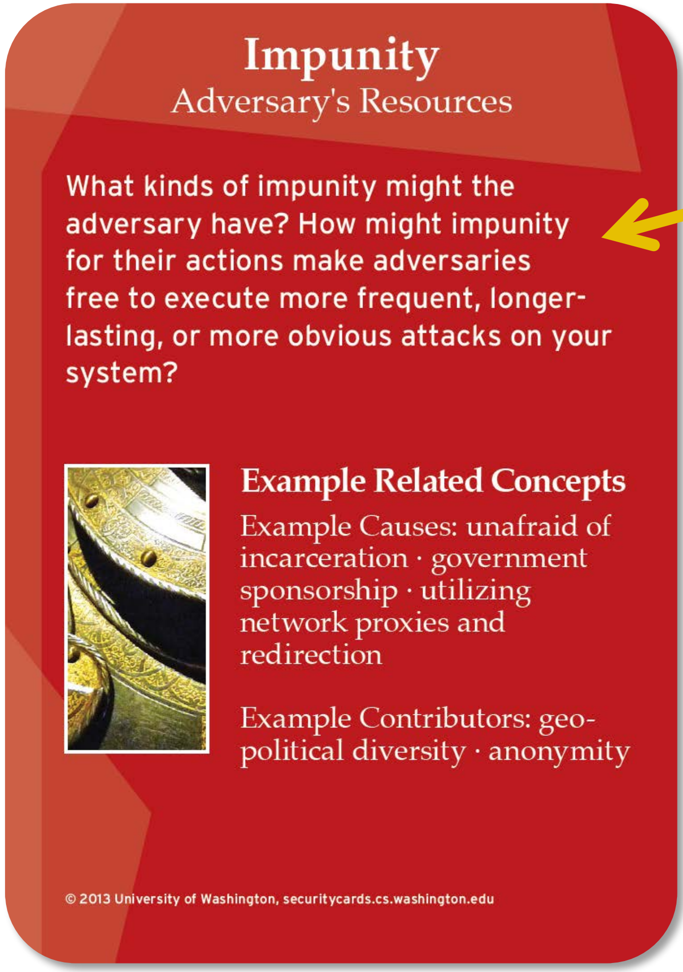
Card topic

Card dimension

Diplomacy or Warfare
Adversary's Motivations
Inside Capabilities
Adversary's Resources
Notice
Employees
Only
Indirect Attack
Adversary's Methods
Emotional Wellbeing
Human Impact

Photograph

# Example Card (Back)



## Impunity

Adversary's Resources

What kinds of impunity might the adversary have? How might impunity for their actions make adversaries free to execute more frequent, longer-lasting, or more obvious attacks on your system?

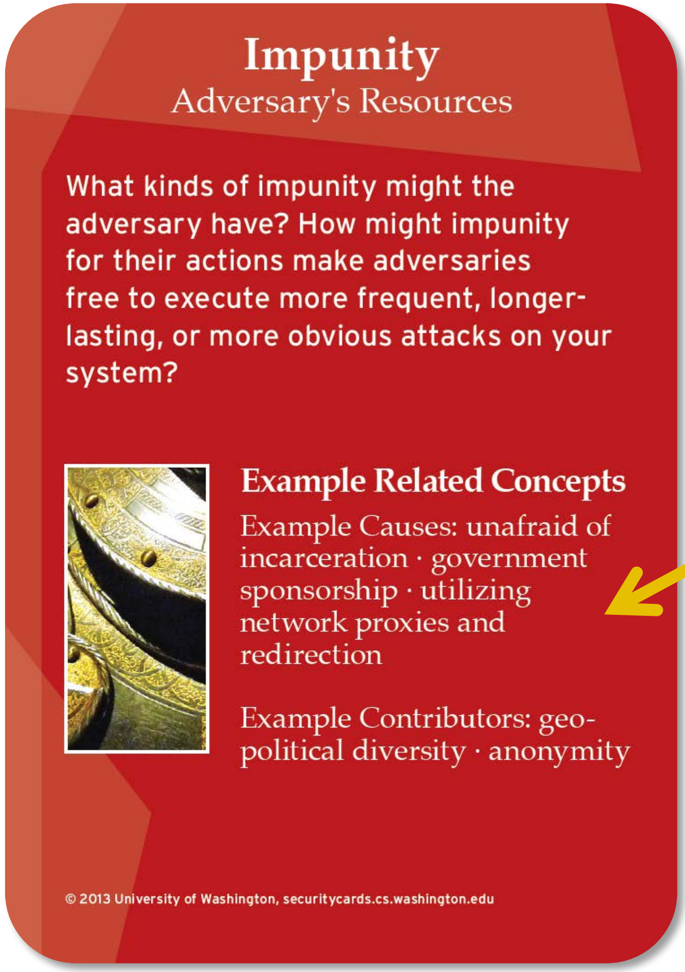
### Example Related Concepts

Example Causes: unafraid of incarceration · government sponsorship · utilizing network proxies and redirection

Example Contributors: geo-political diversity · anonymity

Questions for clarification and to jumpstart thinking

# Example Card (Back)



## Impunity

Adversary's Resources

What kinds of impunity might the adversary have? How might impunity for their actions make adversaries free to execute more frequent, longer-lasting, or more obvious attacks on your system?

### Example Related Concepts

Example Causes: unafraid of incarceration · government sponsorship · utilizing network proxies and redirection

Example Contributors: geo-political diversity · anonymity

Questions for clarification and to jumpstart thinking

Illustrative examples

# Security Card Aspects

### Human Impact

- the biosphere
- emotional well-being
- financial well-being
- personal data
- physical well-being
- relationships
- societal well-being
- unusual impacts

### Adversary’s Resources

- expertise
- a future world
- impunity
- inside capabilities
- inside knowledge
- money
- power and influence
- time
- tools
- unusual resources

### Adversary’s Motivations

- access or convenience
- curiosity or boredom
- desire or obsession
- diplomacy or warfare
- malice or revenge
- money
- politics
- protection
- religion
- self-promotion
- world view

### Adversary’s Methods

- attack cover-up
- indirect attack
- manipulation or coercion
- multi-phase attack
- physical attack
- processes
- technological attack
- unusual methods

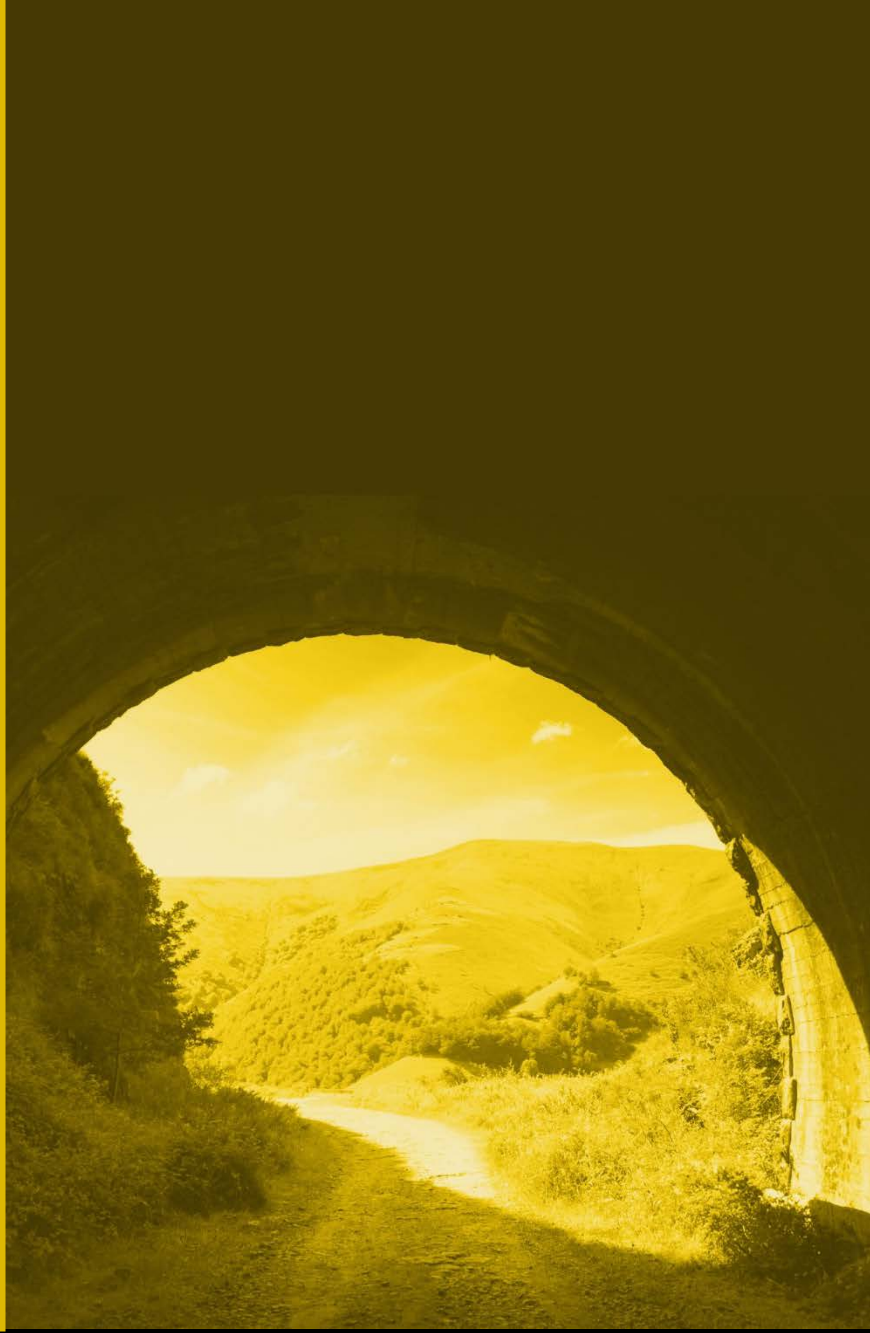
# Security Cards Exercise Instructions

- This exercise will use the Unmanned Autonomous Vehicle (UAV) or Drone scenario.

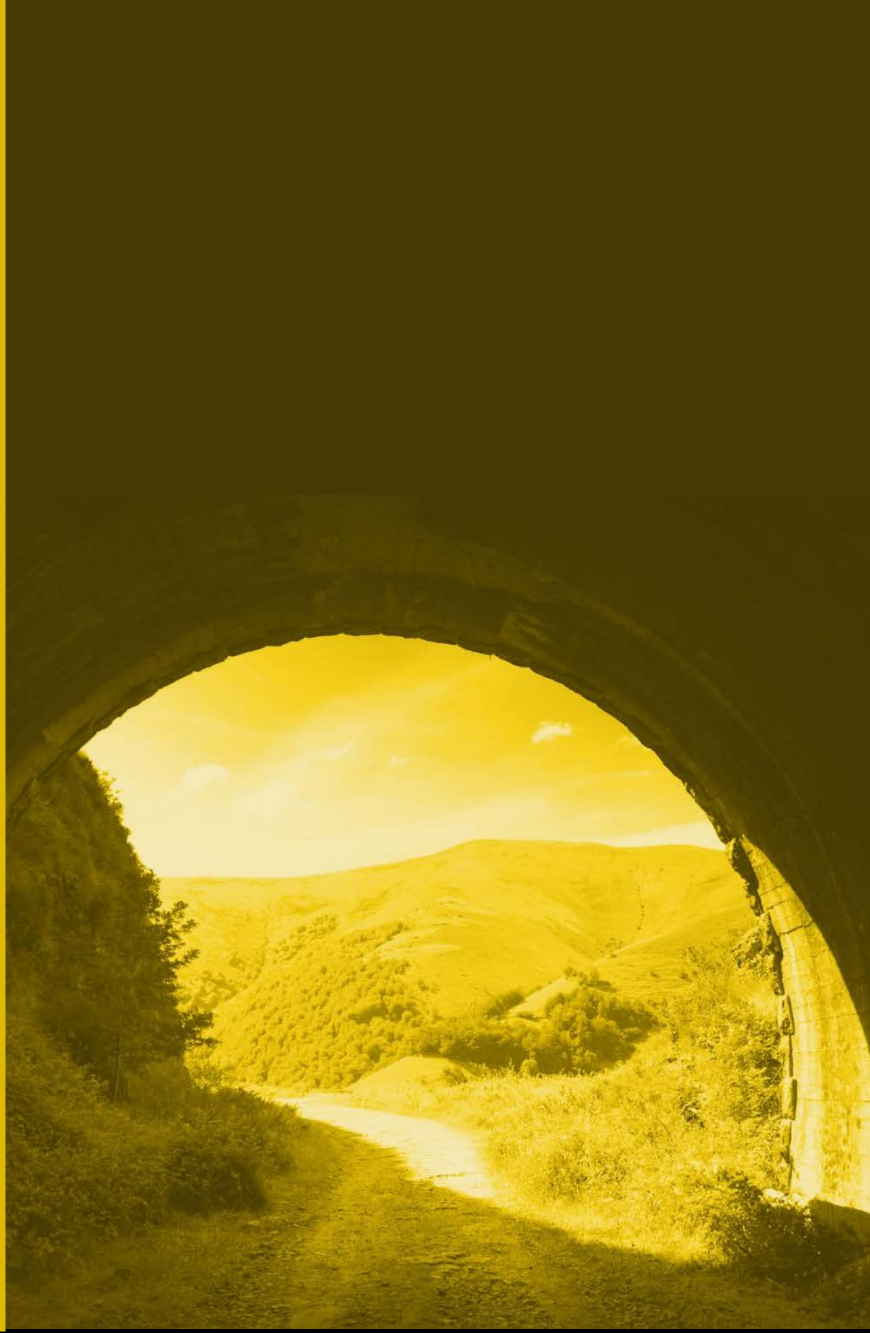
- Read the instructions that have been provided with the scenario.

- Use the security cards that have been provided with your course materials.

# Threat Modeling Methods: Personas and Persona non Grata



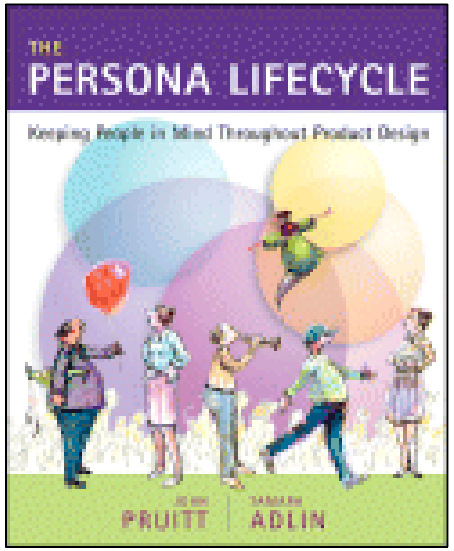
# Personas



# What is a persona?

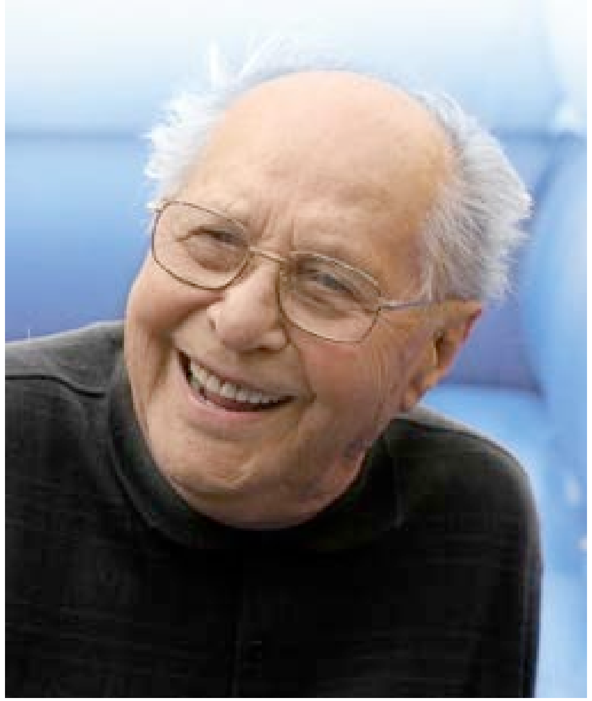
“Personas are detailed descriptions of imaginary people constructed out of well-understood, highly specified data about real people”

— John Pruitt & Tamara Adlin



J. Pruitt, T. Adlin. The Persona Lifecycle: Keeping People in Mind Throughout Product Design. Morgan Kaufman, 2006.

# Example Persona



Thomas is 76 years old, a retired accountant and he enjoys spending time with his grand children. During his retirement, he enjoys reading newspapers, working in his garden and staying in touch with friends. He is a free spirit and enjoys exploration and technology, but only when it doesn’t get in his way.

# Why create personas?

Personas…

- Guide developer decisions about features and how people interact with those features
- Help developers keep users and other stakeholders in mind during development
- Supplement (but cannot replace) developer access to stakeholders during iterations

# Originating Personas from Data

Survey stakeholders

- Good for large N > 1000 individuals to identify segments

Conduct interviews and focus groups

- Good for 4-5 people, hand-picked as representatives

Analyze research articles for facts about stakeholders

- Pew Research Center, Gartner Research, etc.

Read blogs and newspaper articles about similar technologies

# Creating Personas

Identify important categories of stakeholder

- Roles describe the kind of work people do, or their relationship in time to the product
- Goals describe what the users hope to achieve
- Segments describe shared demographic, attitudes or behaviors of your users

# User Roles and Goals

How to describe a role?

- Defined by tasks, job descriptions, responsibilities
- Occupation (shopper, assistant, manager)
- Sub-divide by status: new shopper, repeat customer

What do they care about? How do they feel?

- Defined by their goals
- Behavior (“only browsing”, “get it done”, “max sales”)
- Life phases (adolescence, parenthood, retirement)

# User Segments

Can we segment our users by demographics?

- Age ranges
- Gender
- Income level

What about attitudes or behaviors?

- Physically active, always moving, can’t slow down
- Likes routine, avoids uncertainty, structured
- Telecommuter, works from home, free spirit
- Experienced, technically minded, geek

# The Persona Skeleton

Girl, age 10-13

Computer use at school

- Has access to shared computer in school lab
- Has one computer-related assignment per week
- Finds computer use at school ‘boring’

Internet use at home

- Shares home computer with family
- Uses Internet to play games and do school work
- Enjoys home use of computer

Interests

- Likes to use computers to chat with friends
- Participates in music and art classes

# The Persona Sketch

Annie is 10 and she just started 5th grade, which is very cool. She has computer lab once a week and she likes it a lot. She rushes to finish her homework so she has time in the evening to chat with friends. She thinks she's a computer pro; her mom's been coming to her for help with their family computer for years now.

# Partitioning the stakeholders into personas

Use a data-driven approach, whenever possible

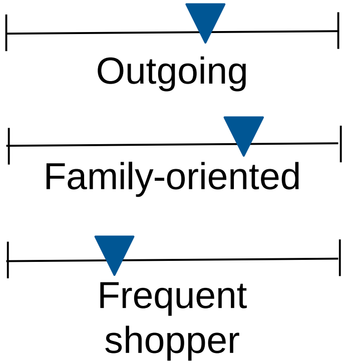
- Data collected using surveys or focus groups
- Data reported in research studies
- Data inferred using affinity diagrams

Diversify your selections

- The common case (most users)
- The extremes (rare, but demanding users)

# Example Personas

**Name**: Aarnav
**Age**: 28
**Job**: Financial Analyst

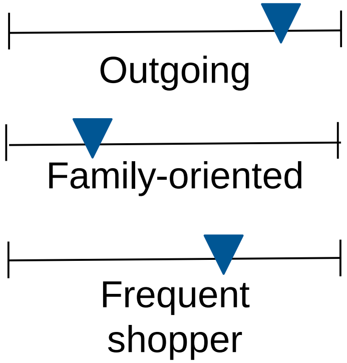


Outgoing

Family-oriented

Frequent shopper

**Name**: Kayaan
**Age**: 22
**Job**: Student



Outgoing

Family-oriented

Frequent shopper

Aarnav enjoys planning his future. He looks for friends who inspire him to improve his life and who help others. Quote: “The ignorant are the blind”

Kayaan enjoys meeting new people. He likes being the center of attention and will seek ways to get reactions from his friends. Quote: “I love my hair!”

# Example Scenario

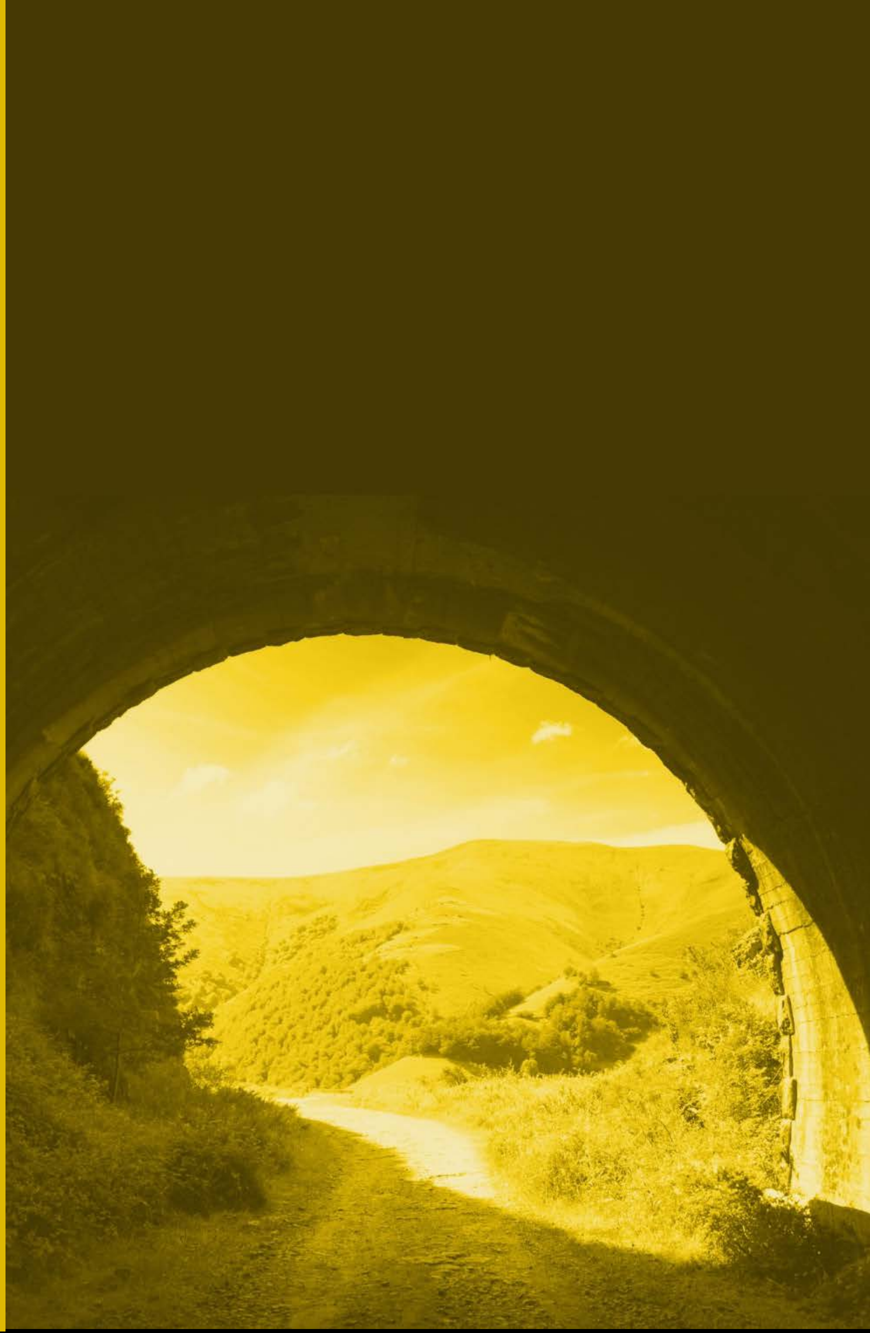
Scenarios illustrate a “thread of execution” through the existing or envisioned system

Example:

“Today is election day, and Aarnav is ready to vote. He arrives at the voting location and shows his identification to the station attendant, who then checks Aarnav’s name against the roster of registered voters. Aarnav is given a virtual ballot to activate the voting machine. He enters the voting booth, closes the curtain, enters the ballot into the machine, and casts his vote. Upon leaving, he returns the ballot to attendant.”

Provided by Ole Villadsen, Carnegie Mellon University

# Persona Non Grata



# From Persona to Persona non Grata (PNG)

PnG? An unacceptable or unwelcome person

- Help developers think about how requirements/features can be intentionally or unintentionally abused
- Useful for understanding—and defending against—malicious users.

Steps:

- Begin with consideration of attackers and their motivations and abilities
- Shift to attackers’ targets and attack mechanisms

Cleland-Huang, 2014; Shull & Mead, 2016

# Persona non Grata:  Background

PnG concept developed at DePaul University, based on Human Computer Interaction (HCI) theory & practice

Similar research on Negative User Stories & Negative Roles for agile software development at Concordia University (Montreal)

- In agile methods, software requirements often originate with user stories or use cases

PnG and Negative User Stories build on the idea in Persona development of the “Negative Persona” or “Anti-Persona”

- Negative Personas are types of users the product is designed not to serve, ranging from tech-savvy early adopters to criminals

Kamthan & Shahmir, 2016; Cooper et. al., 2014; Shull & Mead, 2016

# Persona non Grata:  Background (cont'd)

**PnGs** (and Negative User Stories) developed through techniques such as brainstorming and mind-mapping

**Personas** normally developed using ethnographic methods e.g. interviews and surveys

**Bottom Line:** A move away from checklists, regulations, and misuse cases to thinking about the negative user i.e. attacker/ abuser

Kamthan & Shahmir, 2016

# Developing a PnG

1. **Motivations:** What is the PnG's motivations? Monetary gain? Revenge? Recognition? Laughs?
2. **Goals:** How will the PnG fulfill their motivation i.e. what do they want to do and how do they plan to get away with it?
3. **Skills:** What abilities do they have to achieve their goal? What other assets do they have e.g. access to infrastructure, relationships to those who have skills?
4. **Misuse cases:** What are the misuse cases the PnG can follow to achieve their goals?

# Example Persona non Grata: Mike

**Description:** Mike worked as a contractor installing SCADA radio-controlled sewage equipment for a municipal authority. After leaving the contractor, Mike applied for a job with the municipality but was rebuffed. Feeling bitter and rejected, Mike decides to get even with the municipality and his former employer.

**Goals:** Cause raw sewage to leak into local parks and rivers and make the events appear as malfunctions. Create a public backlash against the contractor and municipality.

”Mike” is based on the true story of Vitek Boden, who was convicted of causing the release of sewage in Maroochy Shire Council in Queensland, Australia in 2000 after hacking the associated SCADA system. See Abrams & Weiss, 2008

# Example Persona non Grata: Mike (cont'd)

**Skills**:  Extensive knowledge of SCADA equipment, including control computers, relevant programs, and radio communication protocols; access to specialized equipment.

**Misuse cases**:

- Steal control computer and radio equipment from his former employer
- Using the stolen computer, construct a fake pumping control station from which to send radio signals
- Gain remote access to SCADA system and disable alarms at pumping stations
- Issue radio commands (using stolen radio equipment) to instruct pumping stations to release sewage

Abrams & Weiss, 2008

# PnG in Practice

Initial Results from DoD Threat Modeling Working Group

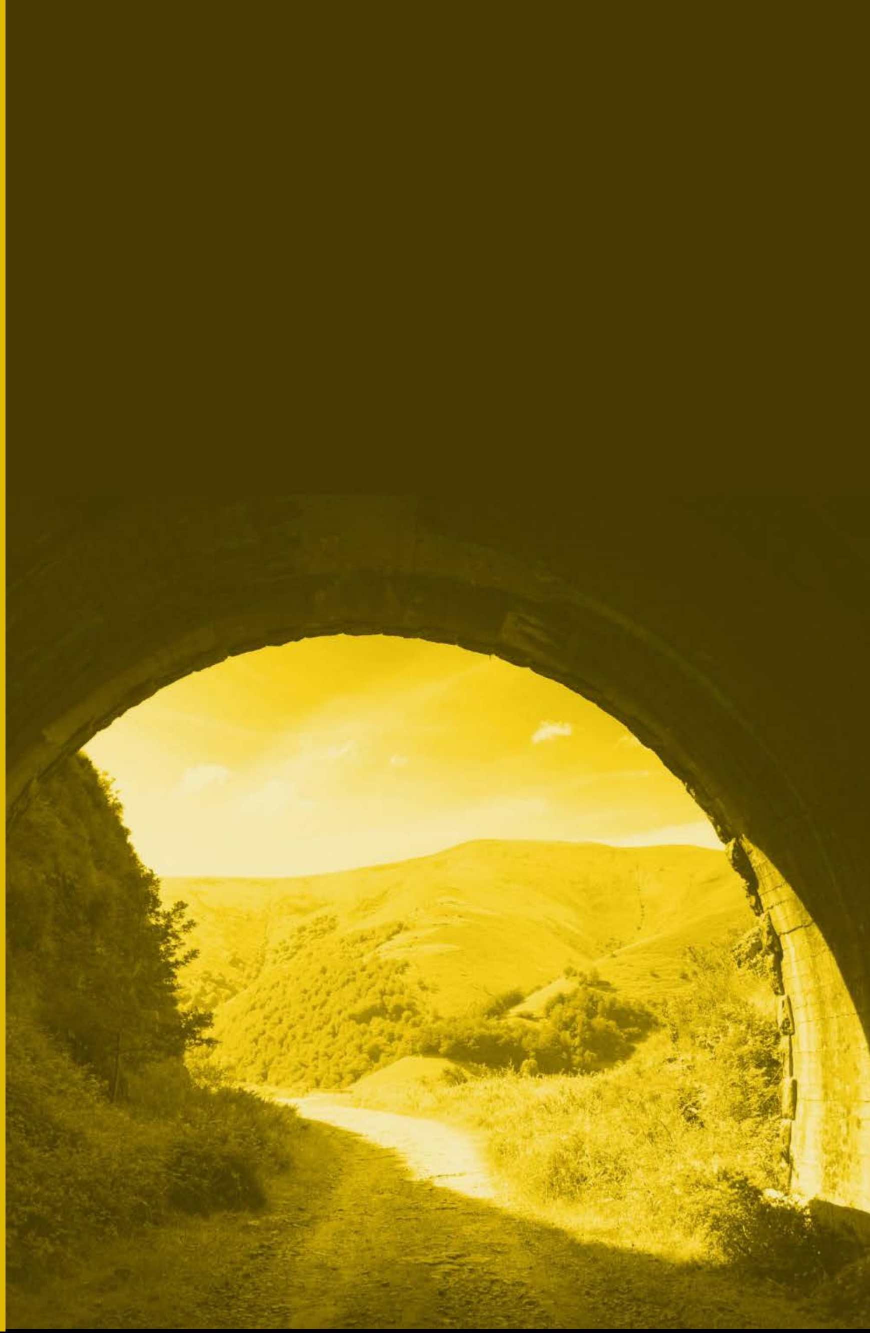
- Useful technique to identify a potential threat with a high degree of confidence
- Not ideal for gaining a comprehensive view of all potential threats.
- Compared to other threat modeling methods, research participants applying PnG yielded fewer false positives
- But they consistently produced only a subset of threat types (which were nonetheless reproduced uniformly across teams)

Shull & Mead, 2016

# Persona-non-grata Exercise

- You’ll be doing an exercise that involves an Implantable Cardioverter Defibrillator (ICD). It summarizes the use of Security Cards and introduces the idea of using personas for threat modeling instead of using Security Cards.
- First you’ll review examples that have been provided for you.  Then you’ll practice creating your own PnGs
- Go ahead and turn to the resources that have been provided to do the exercise.

# Threat Modeling Methods: STRIDE



# STRIDE Threat Model

## STRIDE Threat Categories:

| Threat | Security Property |
|---|---|
| Spoofing | Authentication |
| Tampering | Integrity |
| Repudiation | Non-repudiation |
| Information Disclosure | Confidentiality |
| Denial of Service | Availability |
| Elevation of Privilege | Authorization |

Invented in 1999 by Kohnfelder & Garg; implemented at Microsoft and widely adopted

Typical implementation:

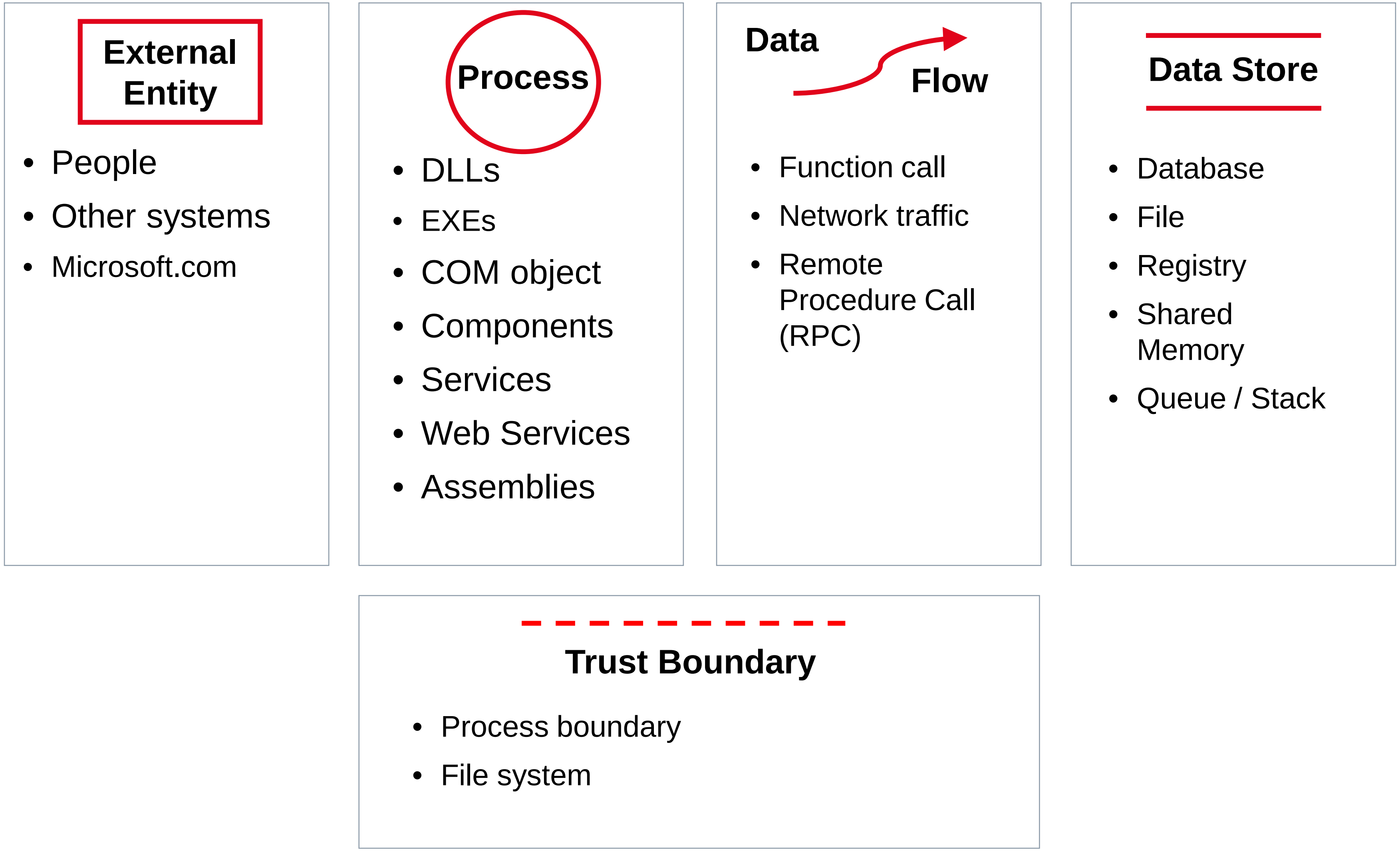
- Model system w/ Data Flow Diagrams (DFD)
- Map the DFD to **Threat Categories**
- Determine the threats (via threat trees)
- Document the threats and steps for prevention

Can be implemented manually or through free SDL Threat Modeling Tool

Considered relatively easy to implement but…time-consuming and prone to different results based on implementer

*Scandariato et. al, 2015; Hernan et. al., 2006*

# Diagram Elements: Examples



### External Entity

- People
- Other systems
- Microsoft.com

### Process

- DLLs
- EXEs
- COM object
- Components
- Services
- Web Services
- Assemblies

### Data Flow

- Function call
- Network traffic
- Remote Procedure Call (RPC)

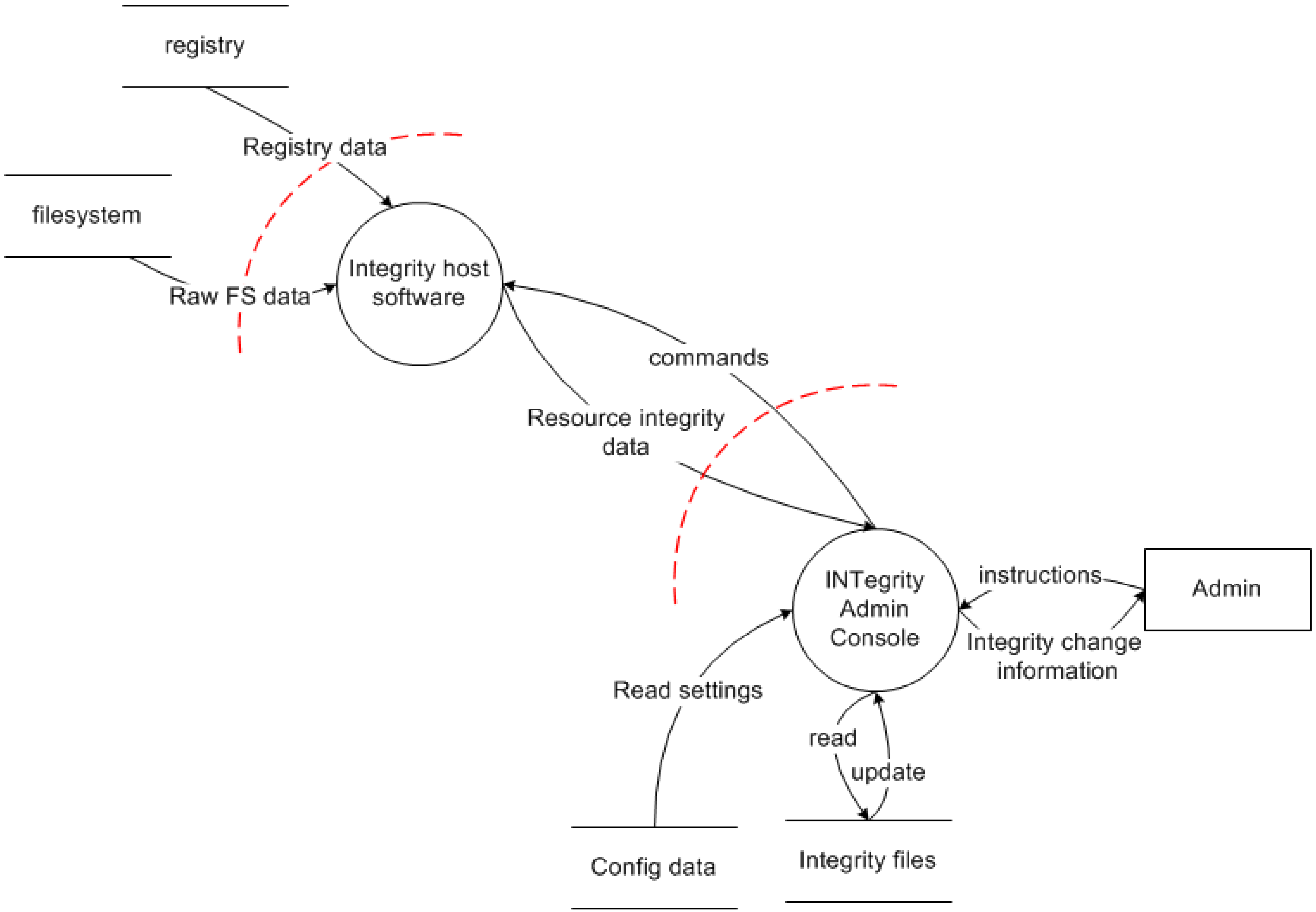
### Data Store

- Database
- File
- Registry
- Shared Memory
- Queue / Stack

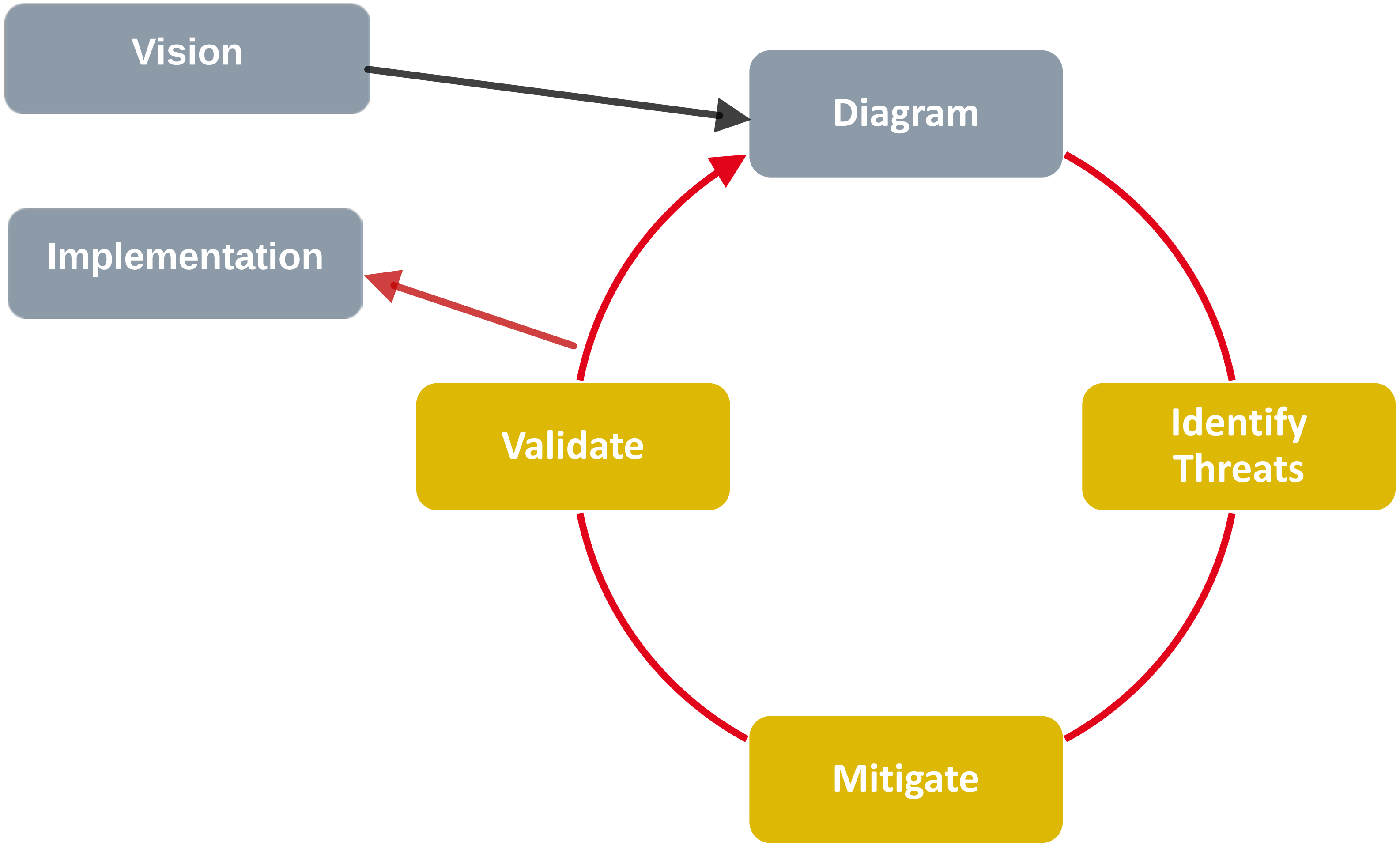
### Trust Boundary

- Process boundary
- File system

# Level 1 Diagram



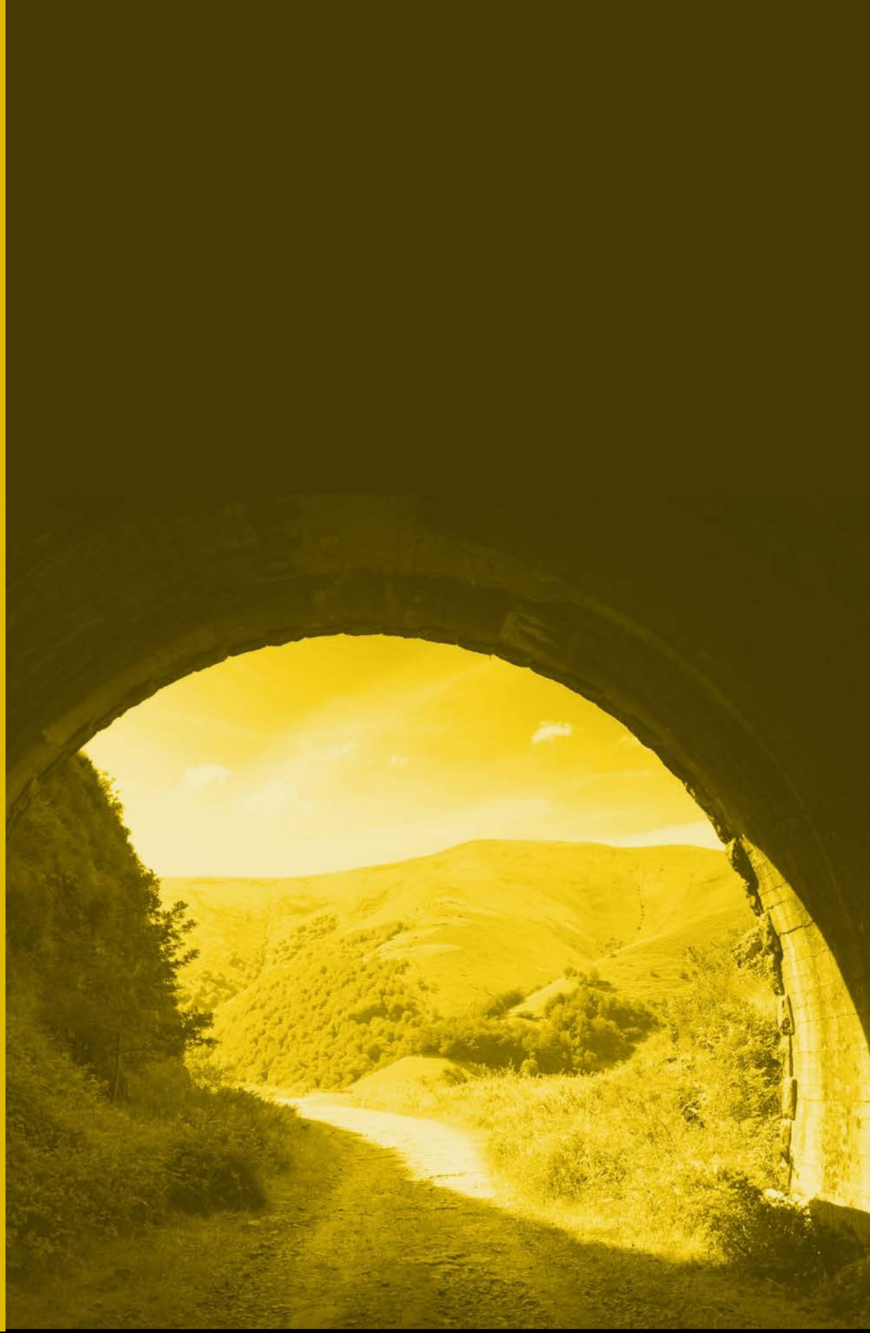
# STRIDE Process (described in SQUARE module)



# STRIDE Exercise Instructions

- For this exercise we will be using the aircraft service scenario in your handouts.
- Follow the exercise instructions in the handout.  If you are able to install the STRIDE tool and use the SimpleStride add-on, it will make the exercise much easier
- You can use these slides and also the slides on STRIDE in the SQUARE course to help.

# SEI’s Threat Modeling Research



# SEI’s threat modeling research

Focus on early lifecycle activities (e.g. requirements engineering, design), independent of lifecycle model

Evaluate competing threat-modeling methods (TMMs) to

- identify and test principles regarding which TMMs yield the most efficacy
- provide evidence about the conditions under which different TMMs are most effective.
- In short, allow reasoning about the confidence to be had in threat modeling results.

# Why we did Cyber Threat Modeling research

## What is threat modeling?

Threat modeling is an activity for creating an abstraction of a software system—aimed at identifying attackers' abilities, motivations, and goals—and using it to generate and catalog possible threats.

> "…engineers have not had sufficient training nor been encouraged to have a mind-set that considers how an adversary might thwart their system… the R&D community has not given engineers the tools they need."
>
> – Greg Shannon, SEI/CERT Chief Scientist, *IEEE Institute*, March 2015

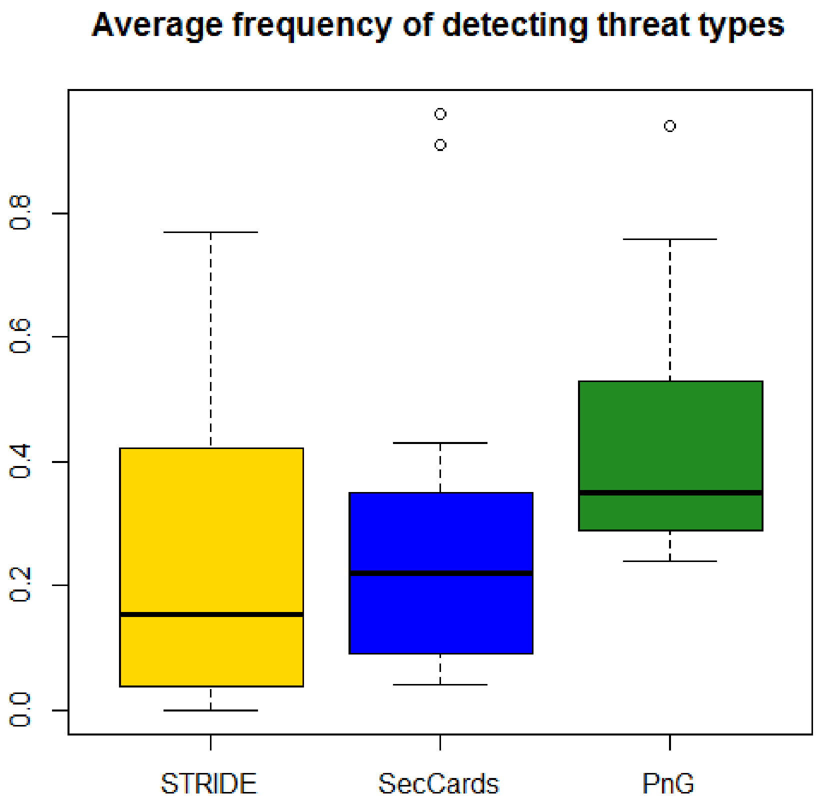
## State of practice

## Goals o research

- identify and test principles regarding which TMMs yield the most efficacy
- provide evidence about the conditions under which different TMMs are most effective

Ultimately, the goal is to improve TMM effectiveness by incorporating the best parts of competing TMMs.

# Results: How frequently is a given threat type reported?



**STRIDE (13 teams)** **Sec.Cards (23 teams)** **PnG (17 teams)**

Comparison of different TMMs applied to the same testbed highlights additional tradeoffs:
If we know that a TMM was able to find a given threat, how confident can we be that it would be reported by a team?

- STRIDE: Great variability.
- Security Cards: Able to find the most threat types but also substantial variability across teams.
- PnG: Was the most focused TMM, but showed the most consistent behavior across teams.

No single TMM led to teams reporting a majority of the valid threats.

# Desirable Threat Modeling Characteristics

Desirable Characteristics for Threat Modeling Method

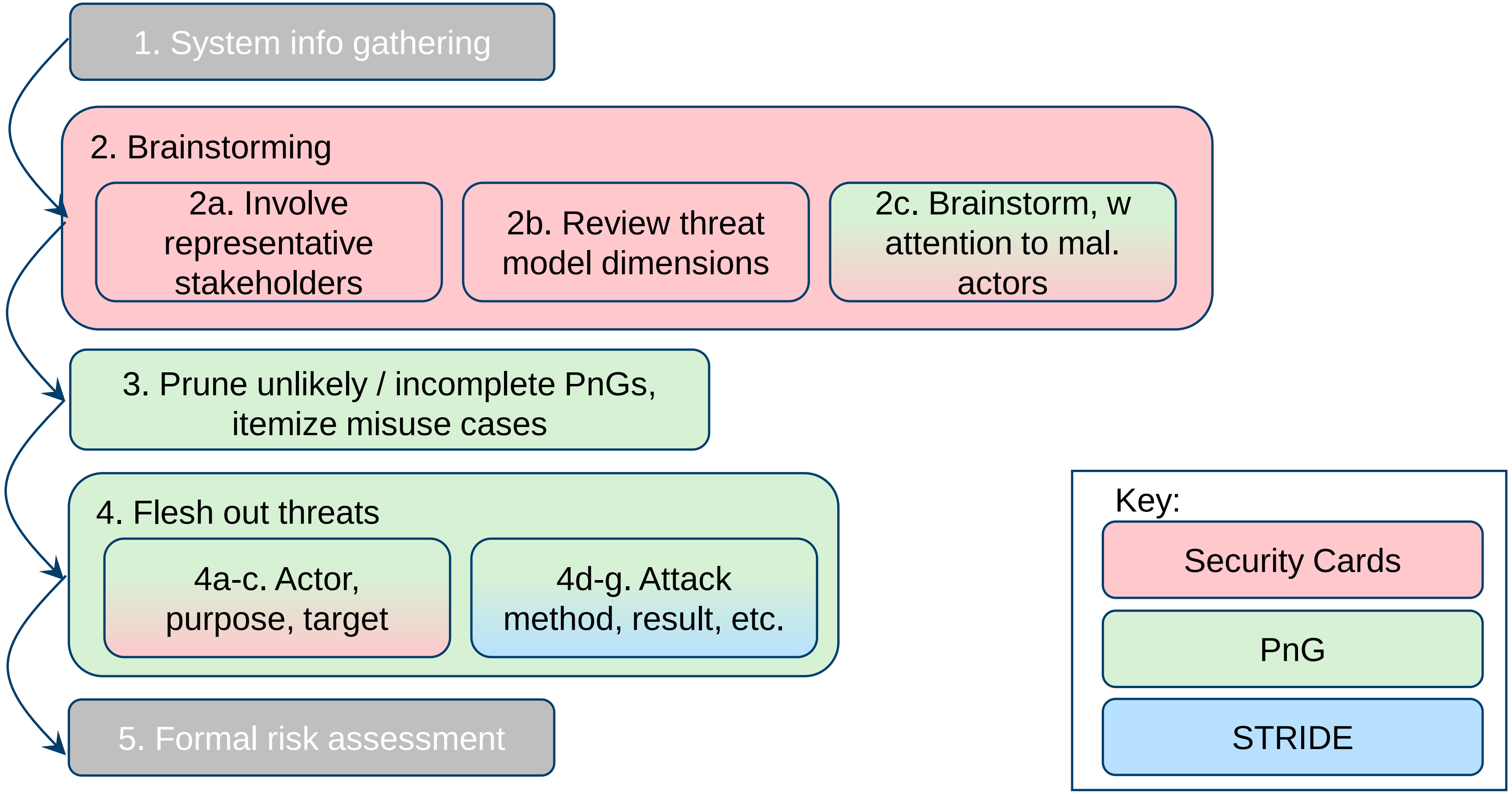
- Minimal false positives
- Minimal overlooked threats
- Consistent results regardless of who is doing the threat modeling
- Cost-effective (doesn’t waste time)
- Empirical evidence to support its efficacy

Other considerations

- Has tool support
- Suggests a prioritization scheme
- Easy to learn, intuitive
- Encourages thinking outside the box
- Can be used by non-experts, or conversely, optimal for experts.
- Clearly superior for specific types of systems

One reference, in addition to our own thinking:
http://threatmodeler.com/successful-threat-modeling/

# Initial Hybrid Threat Modeling Method (hTMM)



1. System info gathering

2. Brainstorming

2a. Involve representative stakeholders

2b. Review threat model dimensions

2c. Brainstorm, w attention to mal. actors

3. Prune unlikely / incomplete PnGs, itemize misuse cases

4. Flesh out threats

4a-c. Actor, purpose, target

4d-g. Attack method, result, etc.

5. Formal risk assessment

Key:

- 🟥 Security Cards
- 🟩 PnG
- 🟦 STRIDE

# Initial Hybrid Threat Modeling Method (hTMM)

1. Identify the system you will be threat modeling. Execute steps 1-3 of SQUARE or a similar security requirements method.
   a. Agree on definitions
   b. Identify a business goal for the system, assets and security goals
   c. Gather as many artifacts as feasible.
2. Apply security cards in the following way, as suggested by the developers (http://securitycards.cs.washington.edu/)
   a. *Distribute the Security Cards to participants either in advance or at the start of the activity.* Include representatives of at least the three following groups of stakeholders: System users / purchasers, system engineers/developers, and cybersecurity experts. You may find that within each of those categories, there are multiple distinct perspectives that need to be represented. Other relevant stakeholders can be included as well.
      i. System users/purchasers would include those purchasing or acquiring the system, end users, and other groups with a vested interest in the system. For example, in a scientific research organization, stakeholders could include the scientists conducting research, the executive directors of the organization, human resources, and information technologists managing the system. Each would have their own ideas about assets that need to be protected and potential attackers.
      ii. Cybersecurity experts could be part of a separate specialized team or integrated into the project team. They could include roles such as system administrators, penetration testers or ethical hackers, threat modelers, security analysts, and so on.
      iii. The engineer/development team members could range from systems engineers, requirements analysts, architects, developers, testers, and so on.
   b. Have the participants look over the cards along all four dimensions: Human Impact, Adversary's Motivations, Adversary's Resources, and Adversary's Methods. Read at least one card from each dimension, front and back.
   c. Use the cards to support a brainstorming session. Consider each dimension independently and sort the cards within that dimension in order of how relevant and risky it is for the system overall. Discuss as a team what orderings are identified. It's important to be inclusive, so do not exclude ideas that seem unlikely or illogical at this point in time. As you conduct your brainstorming exercise, record the following:
      i. If your system is compromised, what assets, both human and system, could be impacted?
      ii. Who are the Personae non Gratae (https://www.infoq.com/articles/personae-non-gratae) who might reasonably attack your system and why? What are their names/job titles/roles? Describe them in some detail.
         1. What are their goals
         2. What resources and skills might the PnG have?
      iii. In what ways could the system be attacked?
         1. For each attack vector, have you identified a PnG (or could you add a PnG) capable of utilizing that vector?
3. Once this data has been collected, you have enough information to prune those PnGs that are unlikely or for which no realistic attack vectors could be identified. Once this has been done, you are in a position to:
   a. Itemize their misuse cases. This expands on HOW the adversary attacks the system. The misuse cases provide the supporting detailed information on how the attack takes place.

# Initial Hybrid Threat Modeling Method (hTMM)

4. Summarize the results from the above steps, utilizing tool support, as follows (https://resources.sei.cmu.edu/asset_files/Presentation/2016_017_001_474200.pdf):
    a. Actor (PnG): who or what instigates the attack? (2.c.ii)
    b. Purpose: what is the actor's goal or intent? (2.c.ii)
    c. Target: what asset is the target? (2.c.i)
    d. Action: What action does the actor perform or attempt to perform? Here you should consider both the resources and the skills of the actor. You will also be describing HOW the actor might attack your system and its expansion into misuse cases. (2.c.iii, and 3.a)
    e. Result of the action: What happens as a result of the action? What assets are compromised? What goal has the actor achieved?
    f. Impact: What is the severity of the result (high, medium, or low)
    g. Threat type: (e.g. denial of service, spoofing)
5. Once this is done, you can continue with a formal risk assessment method, using these results, and the additional steps of a security requirements method such as SQUARE, perhaps tailoring the method to eliminate steps you have already accounted for in the threat modeling exercise.

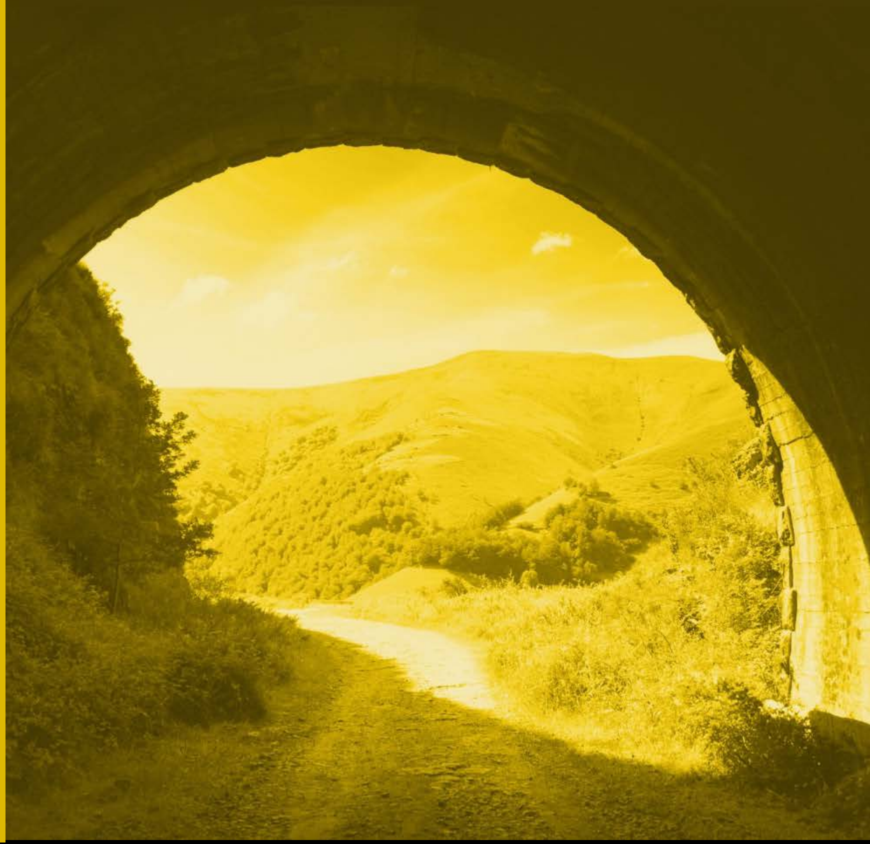
*Measurement considerations:*

a. *Intend to collect data on number and types of issues that come from each stakeholder type so we could have some evidence about what each contributes to the overall threat model.*
b. *Focus on understanding efficiency, using testbeds: How many items get generated in Step 2, then how many are dropped vs refined in Step 3? With some work we could also map those to an oracle dataset so that we could see if the ones that got filtered were actually related to real threats, or if the ones that get refined in PnG were false positives that weren't worth the effort.*

# hTMM Exercise Instructions

- This exercise will revisit the Unmanned Autonomous Vehicle (UAV) or Drone scenario.
- Read the new hTMM instructions that have been provided with the scenario.
- Use the security cards that have been provided with your course materials, along with the PnG concepts.
- When you are done, think about the results you achieved with the hTMM compared to the other methods.

# Other Threat Modeling Methods



# VAST: Visual, Simple, and Agile Threat Modeling

Developed by A. Arguwal, basis for first commercially available threat modeling tool, ThreatModeler.

- Very limited openly available documentation.

Intended for large/medium organizations while supporting prevailing Agile methodology

Designed for consistent output regardless of implementer

Implemented by DevOps teams during SDLC

Divides threat models into two categories:

- Application models: Creates process flow diagrams that focus on specific application
- Operational models: Provides end-to-end data flow diagrams that incorporate application interactions

*MyAppSecurity, 2016*

# Trike Threat Model

Developed in 2003 to improve perceived deficiencies of STRIDE

Designed for security auditing from a risk management perspective

Models threats from defensive perspective i.e. not from attacker’s

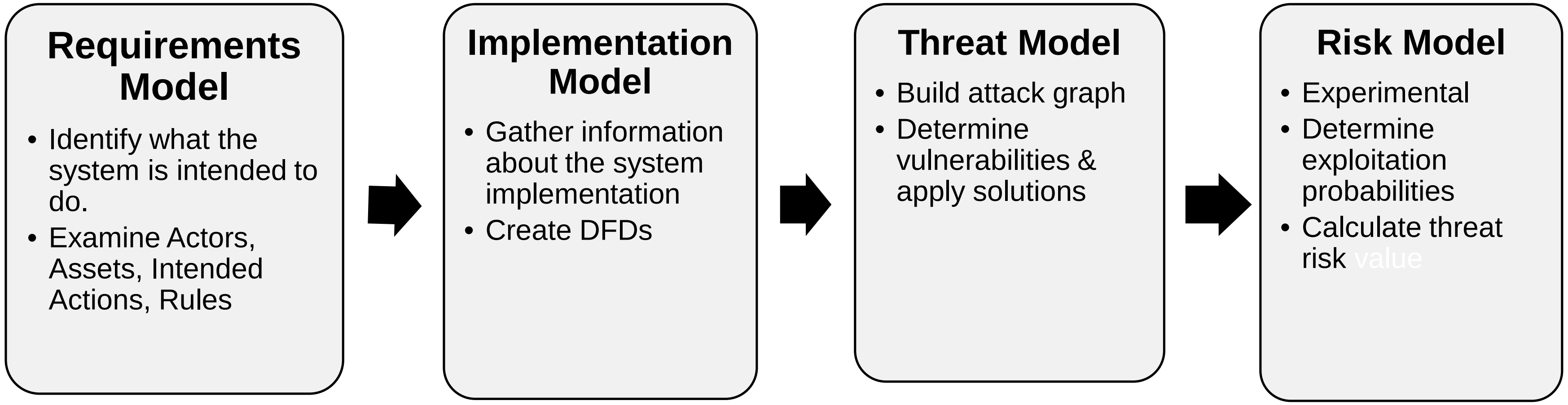
Aims to generate threat models in reliable and repeatable manner through automation of most functions

- Methodology as well as a tool/spreadsheet (www.octotrike.org)
- Multiple, independent implementers should reach same results

Limitations: Difficult to model systems with more than 12 actors and 12 assets; limited documentation and unclear level of tool support

*Saitta et. al, 2005; Larcom, 2012; Dsouza, 2016*

# Trike Implementation



### Requirements Model

- Identify what the system is intended to do.
- Examine Actors, Assets, Intended Actions, Rules

### Implementation Model

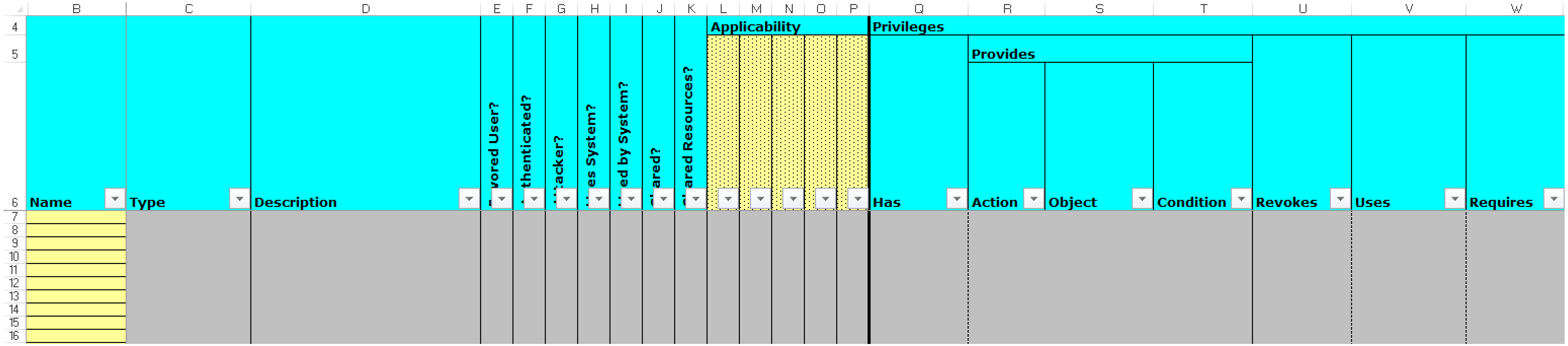
- Gather information about the system implementation
- Create DFDs

### Threat Model

- Build attack graph
- Determine vulnerabilities & apply solutions

### Risk Model

- Experimental
- Determine exploitation probabilities
- Calculate threat risk value

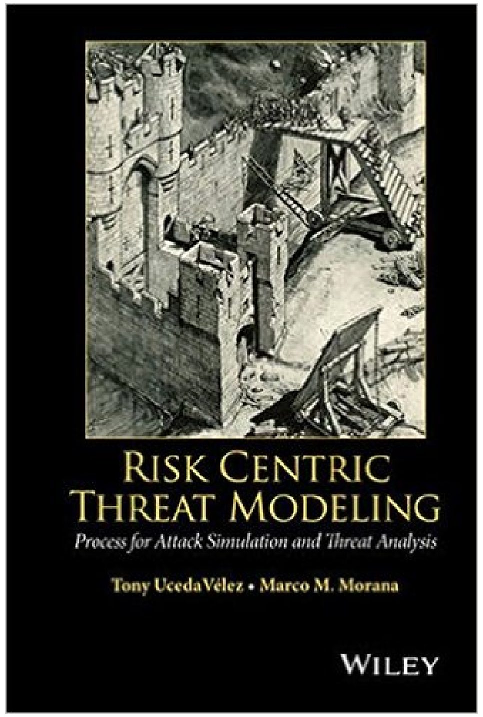


| | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 4 | | | | | | | | | | | Applicability | | | | | Privileges | | | | | | |
| 5 | | | | | | | | | | | | | | | | | Provides | | | | | |
| 6 | Name | Type | Description | vored User? | thenticated? | acker? | es System? | ed by System? | ared? | ared Resources? | | | | | | Has | Action | Object | Condition | Revokes | Uses | Requires |
| 7 | | | | | | | | | | | | | | | | | | | | | | |
| 8 | | | | | | | | | | | | | | | | | | | | | | |
| 9 | | | | | | | | | | | | | | | | | | | | | | |
| 10 | | | | | | | | | | | | | | | | | | | | | | |
| 11 | | | | | | | | | | | | | | | | | | | | | | |
| 12 | | | | | | | | | | | | | | | | | | | | | | |
| 13 | | | | | | | | | | | | | | | | | | | | | | |
| 14 | | | | | | | | | | | | | | | | | | | | | | |
| 15 | | | | | | | | | | | | | | | | | | | | | | |
| 16 | | | | | | | | | | | | | | | | | | | | | | |

Trike Spreadsheet Implementation v.1.05.07 (Actors sheet)

*Saitta et. al, 2005*

# PASTA: Process for Attack Simulation & Threat Analysis



Risk Centric Threat Modeling
Process for Attack Simulation and Threat Analysis
Tony UcedaVélez • Marco M. Morana
Wiley

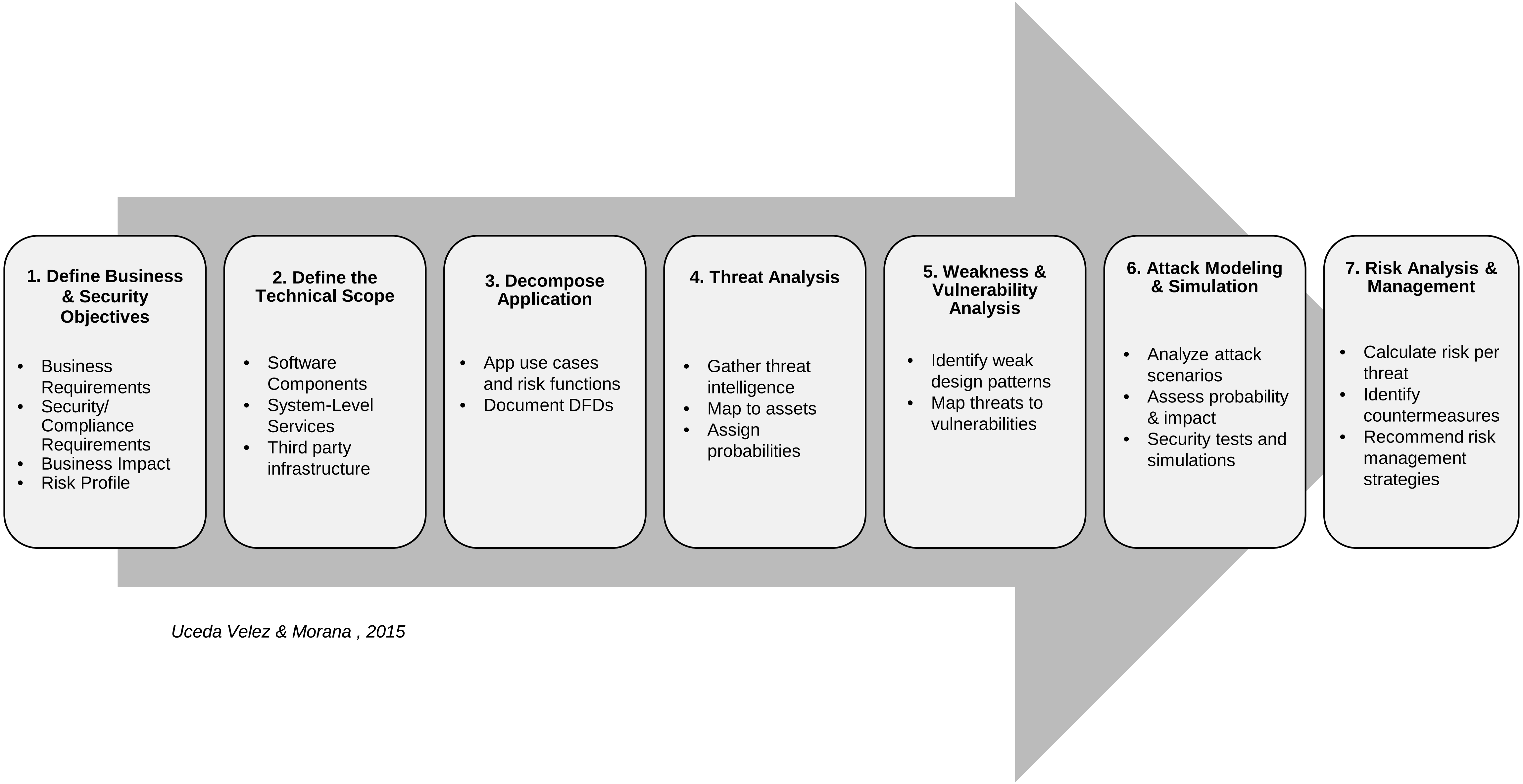
Developed around 2012 by Morana and UcedaVelez; published book in 2015

Seven stage process that yields impact of threat to application and business

- Process begins with defining business objectives & security requirements
- Application is then decomposed into use cases and DFDs
- Threat trees and abuse cases are then introduced
- Final step includes risk and business impact

*Dsouza, 2016*

# PASTA Process



**1. Define Business & Security Objectives**

- Business Requirements
- Security/ Compliance Requirements
- Business Impact
- Risk Profile

**2. Define the Technical Scope**

- Software Components
- System-Level Services
- Third party infrastructure

**3. Decompose Application**

- App use cases and risk functions
- Document DFDs

**4. Threat Analysis**

- Gather threat intelligence
- Map to assets
- Assign probabilities

**5. Weakness & Vulnerability Analysis**

- Identify weak design patterns
- Map threats to vulnerabilities

**6. Attack Modeling & Simulation**

- Analyze attack scenarios
- Assess probability & impact
- Security tests and simulations

**7. Risk Analysis & Management**

- Calculate risk per threat
- Identify countermeasures
- Recommend risk management strategies

*Uceda Velez & Morana , 2015*

# Summary

# Revisiting Our Course Objectives

After completing this course, students will be able to

- Understand how threat modeling fits into the security development lifecycle
- Apply any of the 4 threat modeling methods discussed to their systems.
- Study and assess new threat modeling methods for applicability to their systems

# Additional Practice

- Try applying any of Security Cards, PnG, STRIDE, or hTMM to any of the scenarios.
- Try applying any of these methods to a modest system or subsystem that you are working on now.
- Spend some time investigating commercially available threat  modeling tools.

# References

See the handouts for a short list of references on software security and threat modeling.  Many more references are available in the literature.

# Teamwork

This course is an outgrowth of a threat modeling research project involving several organizations. Hence, in addition to SEI sources, we have included material from:

- Carnegie Mellon University
- DePaul University
- Microsoft
- Notre Dame University
- University of Washington

# Backup Slides for Additional Study

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

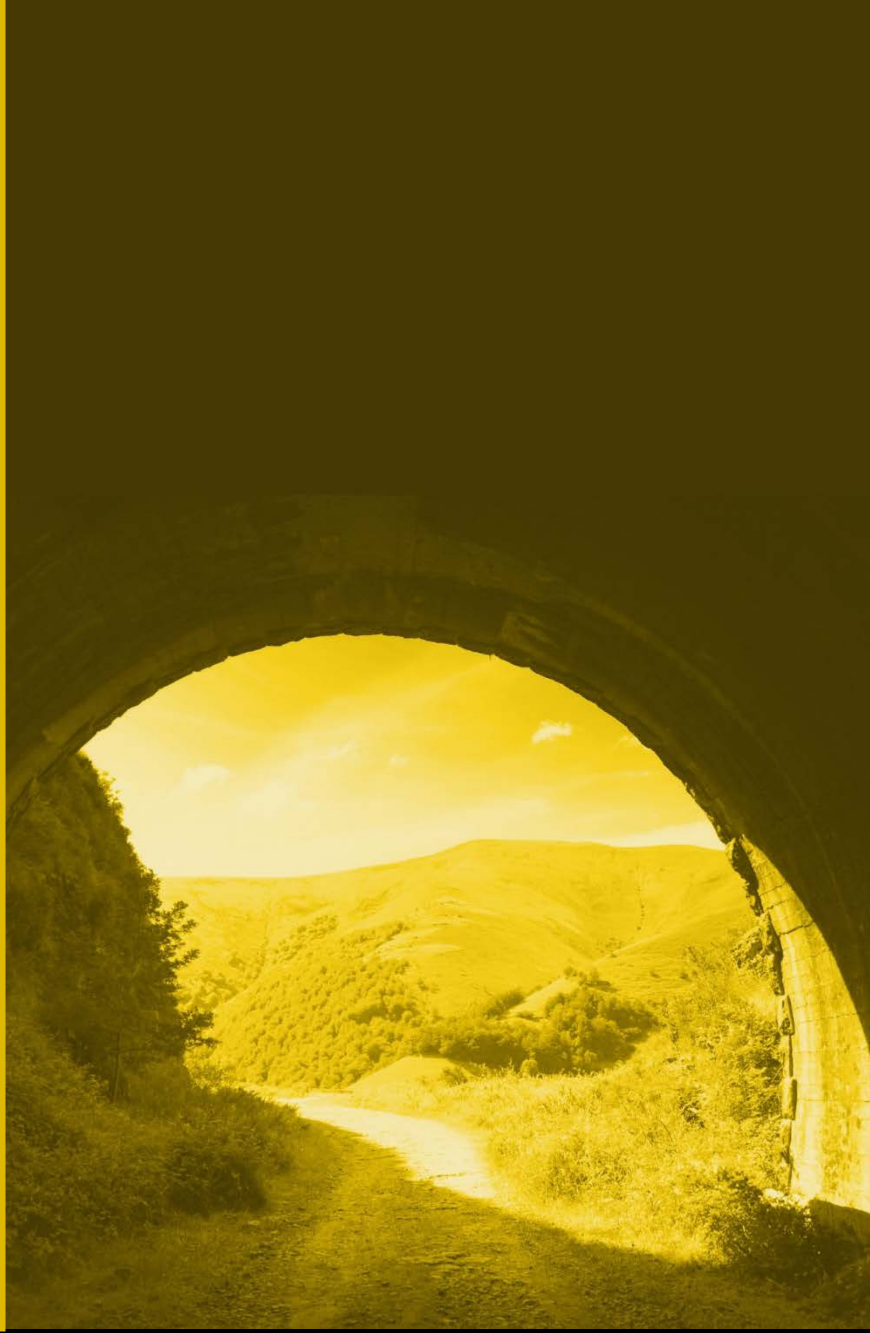
The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

Although the rights granted by contract do not require course attendance to use this material for U.S. Government purposes, the SEI recommends attendance to ensure proper understanding.

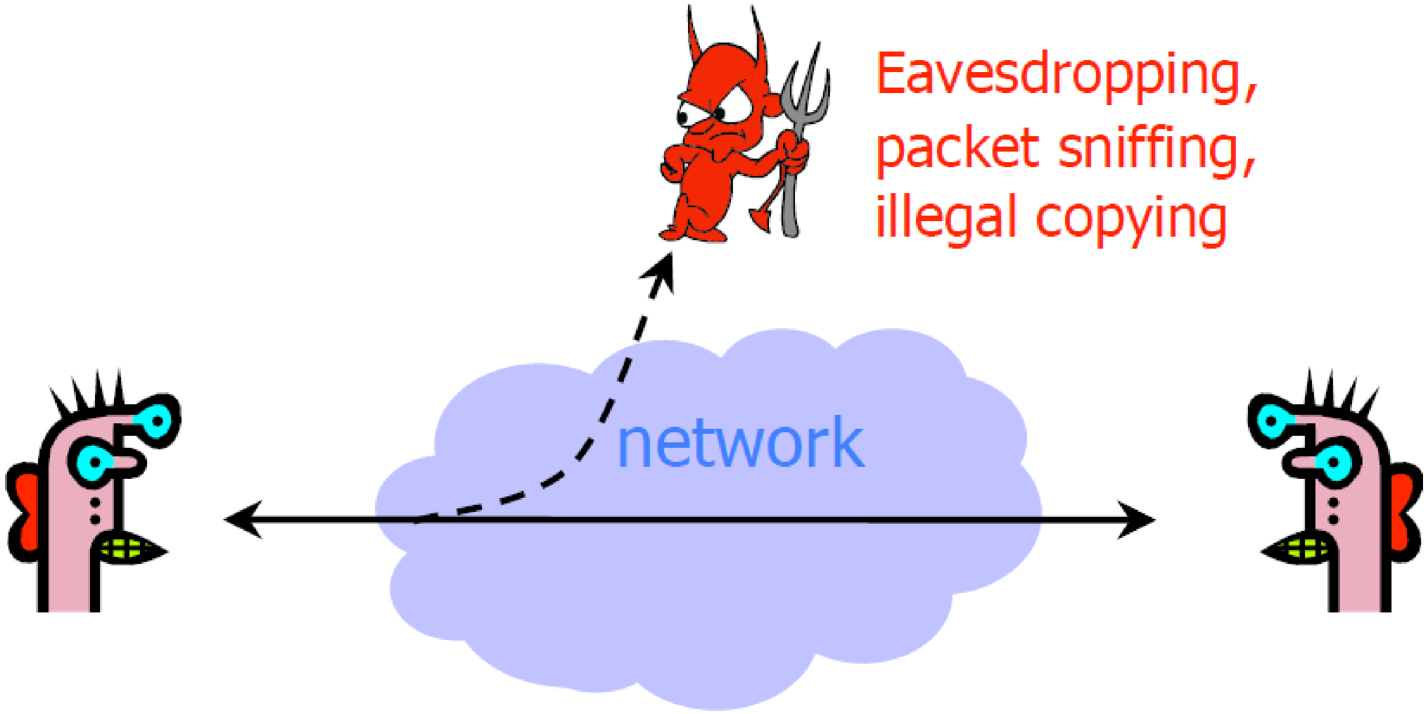
DM17-0724

# Security Goals



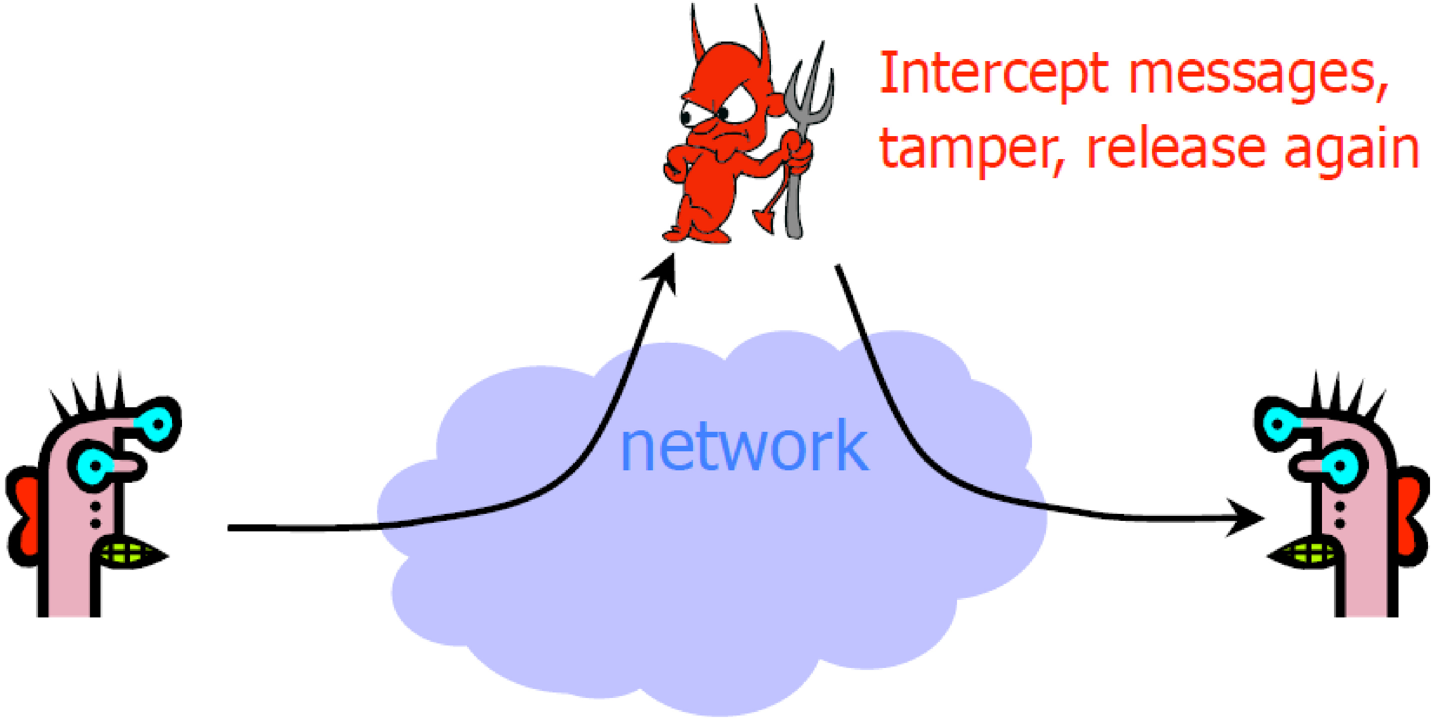
# Confidentiality (Privacy)

Confidentiality is **concealment of information**.



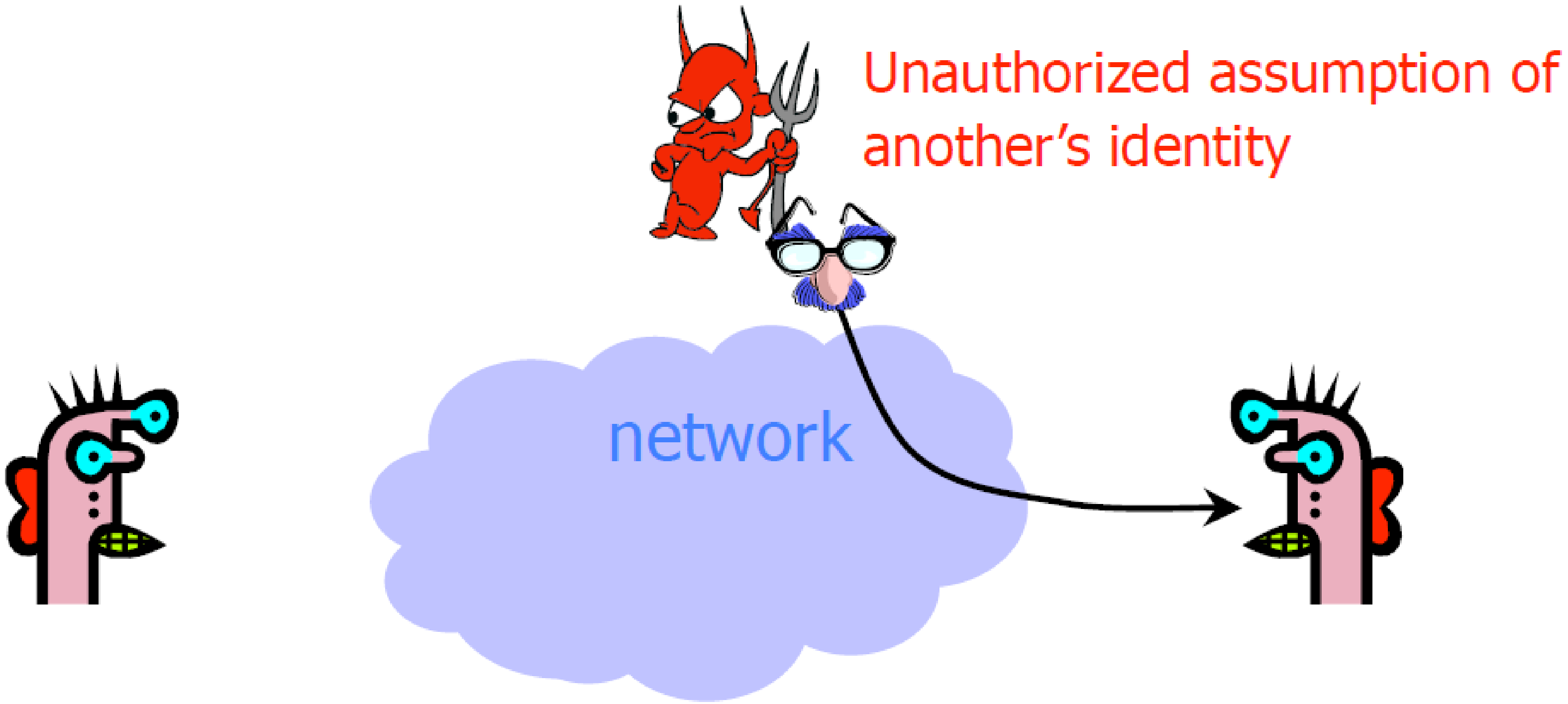
# Authenticity/Integrity (1)

Authenticity/integrity is **prevention of unauthorized changes**.



# Authenticity/Integrity (2)

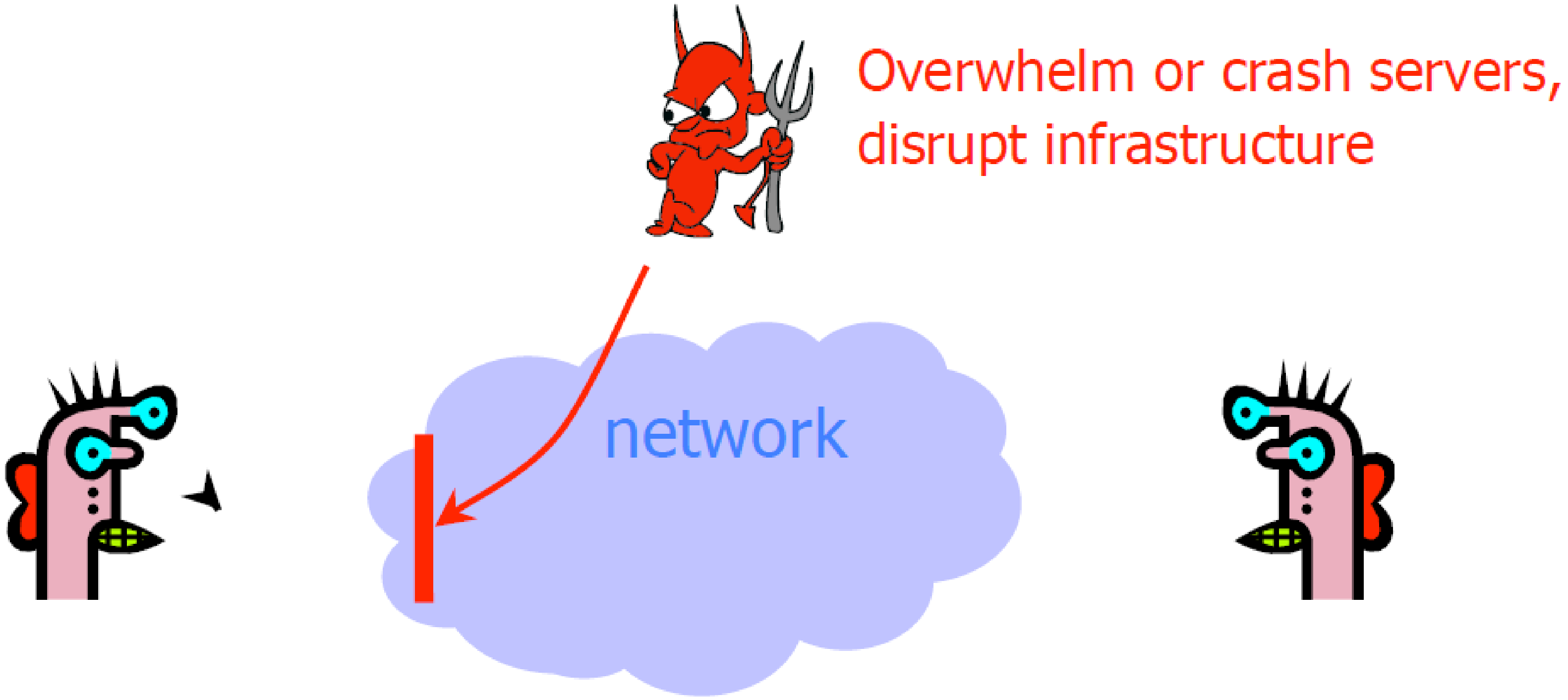
Authenticity/integrity is **identification and assurance of origin of information**.



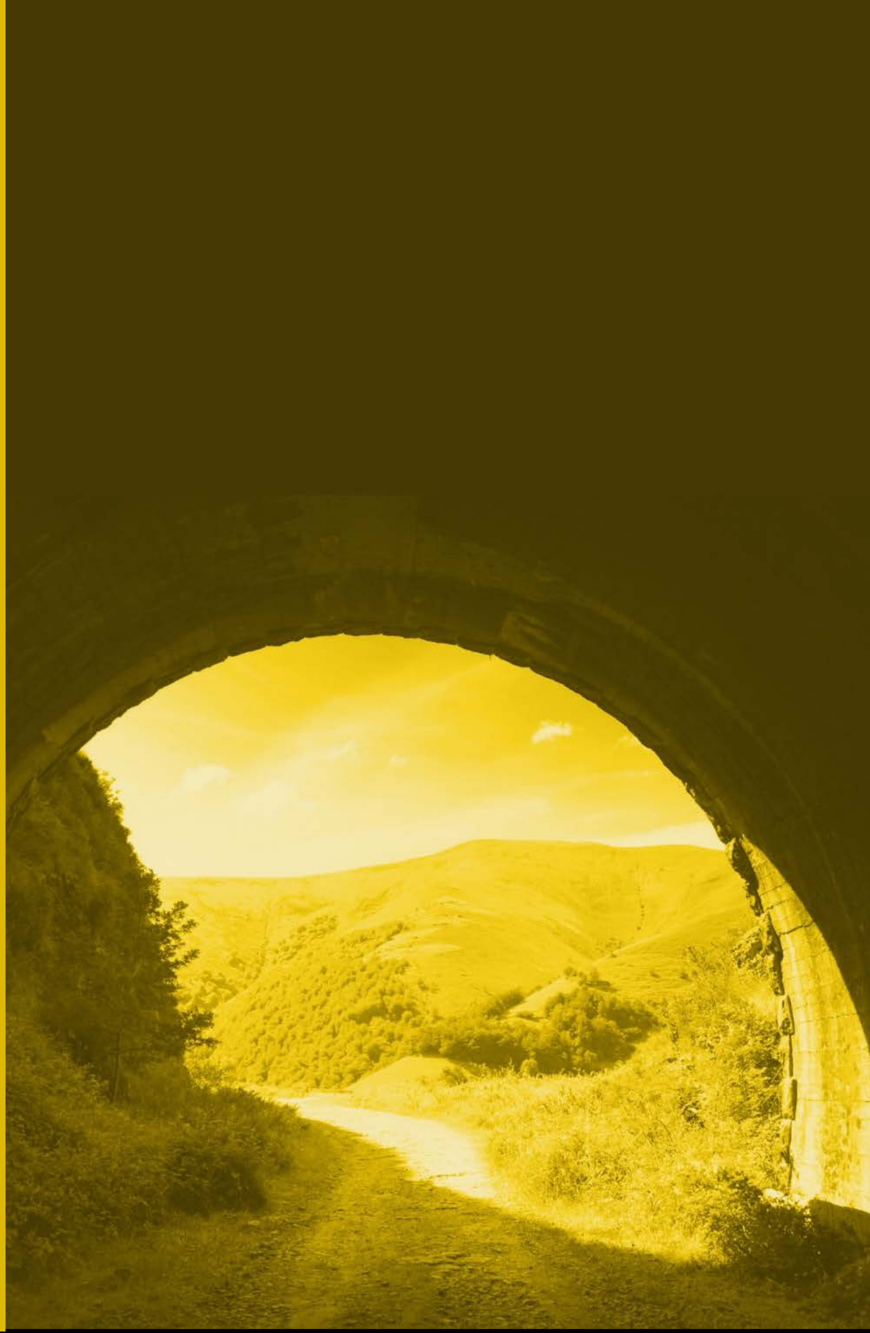
# Availability

Availability is **ability to use information or resources desired**.

.



# Security of a System



# How Systems Fail

Systems may fail for many reasons.

- **Reliability** deals with accidental failures.
- **Usability** deals with problems arising from operating mistakes made by users.
- **Security** deals with intentional failures created by intelligent parties.
    - Security is about computing in the presence of an adversary.
    - Security, reliability, and usability are all related.

# Challenges: What is "security"?

**What does security mean?**

- Often the hardest part of building a secure system is figuring out what security means.
- What are the assets to protect?
- What are the threats to those assets?
- Who are the adversaries, and what are their resources?
- What is the security policy?

**Perfect security does not exist!**

- Security is not a binary property.
- Security is about risk management.

Current events, security reviews, the game, and other discussions are designed to exercise our thinking about these issues.

# From Policy to Implementation

After you’ve figured out what security means to your application, there are still challenges:

**Requirements bugs**

- Incorrect or problematic goals

**Design bugs**

- Poor use of cryptography
- Poor sources of randomness
- ...

Don’t forget the users! They are a critical component!

**Implementation bugs**

- Buffer overflow attacks
- ...

**Is the system usable?**

# Many Participants

**Many parties involved**

- System developers
- Companies deploying the system
- The end users
- The adversaries (possibly one of the above)

**Different parties have different goals.**

- System developers and companies may wish to optimize cost.
- End users may desire security, privacy, and usability.
  - **True?**
- But the relationship between these goals is quite complex. (Will customers choose not to buy the product if it is not secure?)

# Related Issues

- Do consumers actually care about security?
- Security is expensive to implement.
- Plenty of legacy software
- Easier to write “insecure” code
- Some languages (like C) are unsafe.

# Approaches to Security

- Prevention
  - Stop an attack
- Detection
  - Detect an ongoing or past attack
- Response
  - Respond to attacks
- The threat of a response may be enough to deter some attackers.

# Whole System Is Critical

Securing a system involves a **whole-system view**.

- cryptography
- implementation
- people
- physical security
- everything in between

This is because “security is only as strong as the weakest link,” and security can fail in many places.

- No reason to attack the strongest part of a system if you can walk right around it.
- (Still important to strengthen more than the weakest link)

# Analyzing the Security of a System

First thing: Summarize the system as clearly and concisely as possible.

- **Critical step.** If you can’t summarize the system clearly and concisely, how can you analyze its security?
- Summary can be hierarchical

Next steps:

- Identify the assets: What do you wish to protect?
- Identify the adversaries and threats.
- Identify vulnerabilities: Weaknesses in the system.
- Calculate the risks.

# Assets

Need to know what you are protecting!

- Data and information: Data for running and planning your business, design documents, data about your customers, data about your identity, software
- Reputation, brand name
- Responsiveness
- Personal safety
- Physical resources: Laptops, servers, routers, PDAs, phones, ...

Assets should have an associated value (e.g., cost to replace hardware, cost to reputation, how important to business operation).

# Adversaries

- National governments
- Organized crime
- Terrorists
- Thieves
- Business competitors
- Your supplier
- Your consumer
- Newspapers
- Your family members (parents, children)
- Your friends
- Your ex-friends
- ...

# Threats

Threats are actions by adversaries who try to exploit vulnerabilities to damage assets.

- Spoofing identities: Attacker pretends to be someone else.
- Tampering with data: Change outcome of election.
- Crash machines: Attacker makes voting machines unavailable on election day.
- Elevation of privilege: Regular voter becomes admin.

Specific threats depend on environmental conditions, enforcement mechanisms, etc.

- You must have a clear, simple, accurate understanding of how the system works!

# Several Ways to Classify Threats

**By damage done to the assets**

- Confidentiality, Integrity, Availability

**By the source of attacks**

- (Type of) insider
- (Type of) outsider
- Local attacker
- Remote attacker
- Attacker resources

**By the actions**

- Interception
- Interruption
- Modification
- Fabrication

# Vulnerabilities

**Weaknesses of a system that could be exploited to cause damage**

- Accounts with system privileges where the default password has not been changed (Diebold: 1111)
- Programs with unnecessary privileges
- Programs with implementation flaws
- Problems with cryptography
- Weak firewall configurations that allow access to vulnerable services
- ...

**Sources for vulnerability updates: CERT, SANS, Bugtraq, the news, ...**

# Risk Analyses: Lots of Options (None of Them Great)

**Quantitative risk analysis**

- Example: $\boxed{\text{Risk}} = \boxed{\text{Asset}} \times \boxed{\text{Threat} \times \text{Vulnerability}}$
- Monetary value to assets
- Threats and vulnerabilities are probabilities
- (Yes: Difficult to assign these costs and probabilities)

**Qualitative risk analysis**

- Assets: Critical, very important, important, not important
- Vulnerabilities: Very likely, likely, unlikely, very unlikely
- Threats: Very likely, likely, unlikely, very unlikely

# Helpful Tables

| Asset | Confidentiality | Integrity | Availability |
|---|---|---|---|
| Reputation | | | |
| Responsiveness | | | |
| Data | | | |
| Personal Safety | | | |
| … | | | |

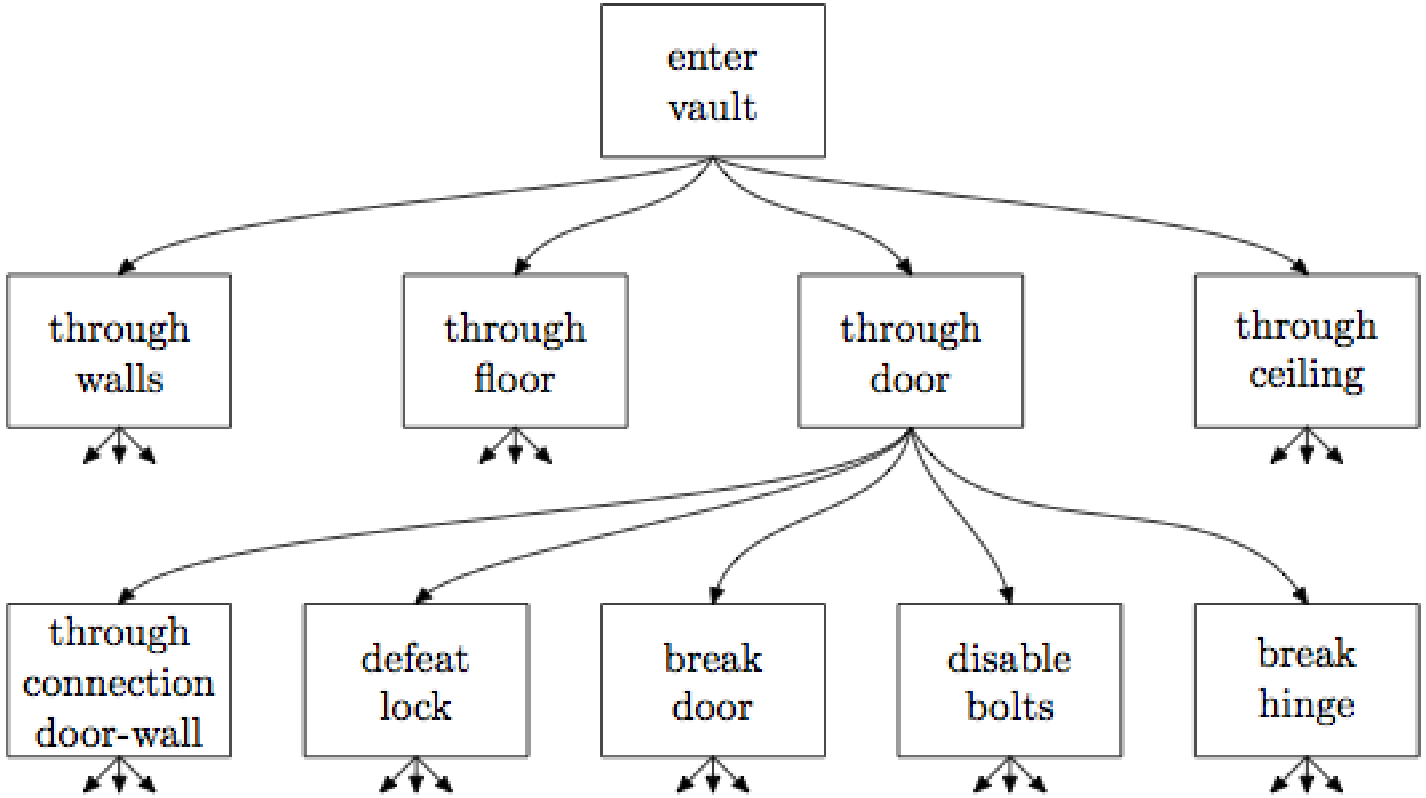
# Helpful Tables

| | Voter | Election official | … |
|---|---|---|---|
| Privacy of vote | | | |
| Integrity of vote | | | |
| Availability of voting system | | | |
| Confidence in election | | | |
| … | | | |

# Helpful Tables

| | Create new voter cards | Decrypt voting record | … |
|---|---|---|---|
| Privacy of vote | | | |
| Integrity of vote | | | |
| Availability of voting system | | | |
| Confidence in election | | | |
| … | | | |

# Attack Trees


enter vault
through walls
through floor
through door
through ceiling
through connection door-wall
defeat lock
break door
disable bolts
break hinge