Threat Modeling

Keith A. Watson, CISSP, CISA
GLSP April 2011 Meeting

Overview of Module 2

- Assets
- Threats
  - STRIDE threat classification system
- Risk
  - DREAD risk rating system
- Threat Modeling

Objectives for Module 2

1. Understand the difference between threats and vulnerabilities.
2. Apply classification and rating techniques.
3. Understand the basic threat modeling process.

Assets

- Assets are items of value to an organization
  - Can be tangible or intangible
- Examples:
  - Organization's reputation
  - Data
  - Processing capabilities
  - Availability of resources
  - Network

Threats

- A threat is an action that an attacker might take to affect an asset
  - unauthorized access, destruction, disclosure, modification of data, and/or denial of service
  - Includes the means and motivation
- A threat cannot exist without a target asset

Threat Agents

- Anyone or anything that can impact an asset can be a threat agent
  - Malicious attackers
  - Untrained or inept users or system operators
  - Thieves stealing laptops
  - Animals causing power outages
  - Floods, fires, earthquakes, lightning
- Can intentional or unintentional

Motivation and Skills

- Intentional human attackers are typically driven by some motivation
  - Money (sale of sensitive information)
  - Political (work on behalf of a government org)
  - "Hacktivism" or terrorism (promote a cause)
  - Explorers (non-target specific)
- Skill levels vary widely
  - Script kiddie (untrained, low skill level)
  - Professional criminal (trained, high skill level)

Threat Consequences

- The violations of the security of asset by a threat action
  - Disclosure
  - Deception
  - Disruption
  - Usurpation

STRIDE

- **STRIDE** is a classification system for attacker objectives

  Spoofing an identity

  Tampering with data

  Repudiation

  Information Disclosure

  Denial of Service

  Elevation of Privilege

STRIDE

- Some threats can fit into more than one STRIDE category
- What if an attacker:
  - Modifies an employee database
  - Spoofs admin access to a server
  - Sends malicious code to web browser

STRIDE Exercise

- Handout "Exercise #1"
- Part 1: Match STRIDE attacker objective with threat actions
  - There may be more than one objective per action
- Part 2: Match STRIDE attacker objective with security concepts to prevent the attack

- Take 5 minutes

DREAD

- DREAD is a complement to STRIDE
  - Also promoted by Microsoft
- Risk calculation methodology

Damage Potential

Reproducibility

Exploitability

Affected Users

Discoverability

DREAD Values

- Numbers are assigned to represent extent of damage, percentages, ratios, effort, etc.
  - 30% of users affected
  - Exploits works 1 out 100 attempts
  - 50% of users will experience significant damage
- Each DREAD value should be constrained to range of values that are appropriate
  - 1 to 10
  - 1 to 3

Damage Potential

- If a threat exploit occurs, how much damage will be caused?

  0 = Nothing

  5 = Individual user data is compromised or affected

  10 = Complete system or data destruction

Reproducibility

- How easy is it to reproduce the threat exploit?

  0 = Very hard or impossible

  5 = One or two steps required, may need to be an authorized user

  10 = Easily reproduced, without authentication

Exploitability

- What is needed to exploit this threat?

  0 = Advanced programming and networking knowledge, with custom attack tools

  5 = Malware exists on the Internet, or an exploit is easily performed, using available attack tools

  10 = Just a web browser

Affected Users

- How many users will be affected?

  0 = None (0-10%)

  5 = Some users, but not all (%50)

  10 = All users (%100)

- May be dependent on configuration options

Discoverability

- How easy is it to discover this threat?

  0 = Very hard to impossible; requires source code or administrative access

  5 = Can figure it out by guessing or by monitoring network traces

  9 = Details of faults like this are already in the public domain and can be easily discovered using a search engine

  10 = The information is visible in the web browser address bar or in a form

DREAD Risk Calculation

- Risk is calculated as an average of the DREAD values: (D + R + E + A +D) / 5 = Risk Rating

  0 = No risk

  5 = Moderate risk

  10 = Significant risk

DREAD Exercise

- Handout "Exercise #2"
- Read through the application description and threat findings.
- Evaluate the DREAD risk categories and assign a value to each from 0 to 10
- Average each category and compute overall risk

- Take 10 minutes

STRIDE and DREAD Issues

- STRIDE and DREAD are ways of looking at attacks from the attacker's point of view
- Introduced in the first version of Microsoft's Threat Modeling

- Subjective, requires security knowledge
- Focused on attacks and not threats

Architecture Vulnerabilities

- Finding vulnerabilities in the architecture is not possible with a code review
  - Too close to the code to see the larger issues
  - Code reviews come after the code is written
- No software tool available will automatically find architecture vulnerabilities

- How do developers find architecture vulnerabilities?

Implementation Vulnerabilities

- Reviewing code requires time
  - If the app has 1M LOC, it is not possible to review everything in a reasonable time
- There are software tools available to find implementation vulnerabilities
  - They might find a significant number of issues
- How do developers determine where to focus code review and correction efforts?

Threat Modeling

- Building secure software requires an understanding of the threats
- A threat model is a descriptive analysis of risks to the design of software
  - Identifies weaknesses that must be addressed
- Conducted early in the development process
  - Before code is written

Threat Model Advantages

- Threat models provide insight into the application's internal workings
  - Helps new team members understand the app
  - Allows other teams to understand it as well
  - Testers also benefit in development of tests

Threat Model Methodologies

- There are multiple approaches to threat modeling
- Approaches have evolved over time
- Several organizations have different methods
  - Microsoft Secure SDLC
  - OWASP Threat Risk Modeling
  - Calculative Threat Modeling Methodology
  - Trike
  - Consultative Object Risk Analysis System

Threat Model Documentation

- There is no specific requirements for threat modeling documentation
- The format should based on the development team's needs
- The threat model will serve many purposes and has many audiences
  - Architects and developers
  - Software and security testers
  - Outside reviewers

Threat Model Process

- This model based on Swiderski and Snyder's **Threat Modeling** book

1. Understanding the Attacker's View
2. Characterizing the Security of the System
3. Determining Threats

Understanding the Attacker's View

- Analyzing Entry and Exit Points
- Determining the Assets
- Examining Trust Levels

Viewpoints

- Attacker's view
  - Application is a black box
  - Outside view of services available
- Architect and Developer's view
  - Application is a white box
  - Understanding is from the inside out
- Threat models allow the development team to use the attacker's viewpoint in order to address security issues

Entry and Exit Points

- The points at which data or control crosses the boundary of the system
- Entry points represent interactions
  - Junctions with external components
  - Attack points

Data Needed:

1. Numerical ID
2. Name
3. Description
4. Trust Levels

Determining the Assets

- Assets are what the attacker is interested in
  - Need protection from unauthorized use
- Without assets there is no interest in attack

Data Needed:

1. Numerical ID
2. Name
3. Description
4. Trust Levels

Examining the Trust Levels

- Representation of the access rights for external entities
  - Authenticated or remote anonymous user
  - Different trust levels based group association

Data Needed:

1. Numerical ID
2. Name
3. Description

Threat Model Process

1. Understanding the Attacker's View
2. **Characterize the Security of the System**
3. Determining Threats

Characterize the System

- Background information
  - Use scenarios
  - External dependencies
  - External security notes
  - Internal security notes
  - Implementation assumptions
- Modeling the application
  - Data flow diagrams (DFD)

Use Scenarios

- Describe how the system is to be used
  - Or, how it is not to be used
  - Supported and unsupported uses should be listed
- Deals with deployment issues that affect security

Data Needed:

1. Numerical ID
2. Description

External Dependencies

- List requirements for outside components
  - Dependency on behavior or compliance
  - Development has little control

Data Needed:

1. Numerical ID
2. Description
3. External Security Note reference

External Security Note

- Security-relevant information for users that interact with the system

Data Needed:

1. Numerical ID
2. Description

Internal Security Note

- Provide clarification to model readers
- Explain security trade-offs made in the design
  - Business reasons, timeline, costs, etc

Data Needed:

1. Numerical ID
2. Description

Implementation Assumptions

- Assumptions about specific features to be implemented

Data Needed:

1. Numerical ID
2. Description

Data Flow Diagrams

- Modeling the application effectively requires understand how data moves through the system
- DFDs provides visual representation
  - Hierarchical in nature
- Other modeling systems can be used
  - UML
  - Flow charts

DFD Process Primitive

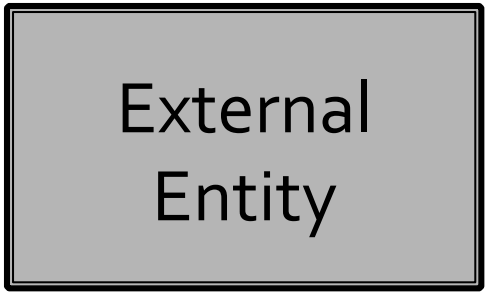
- Process primitive is task that processes data or performs some action based on the data

DFD Complex Process Primitive

- A complex or multiple process contains several sub processes
  - Represents a high-level process in the hierarchy

DFD External Entity Primitive

- External entity is a source or destination of data that exists outside of the system
  - Interacts at an entry point



DFD Data Store Primitive

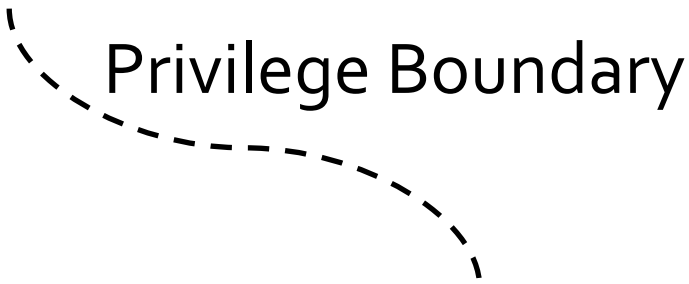
- Repository for data
  - File systems, Files
  - Databases, caches
  - Environment variables, registry

DFD Data Flow Primitive

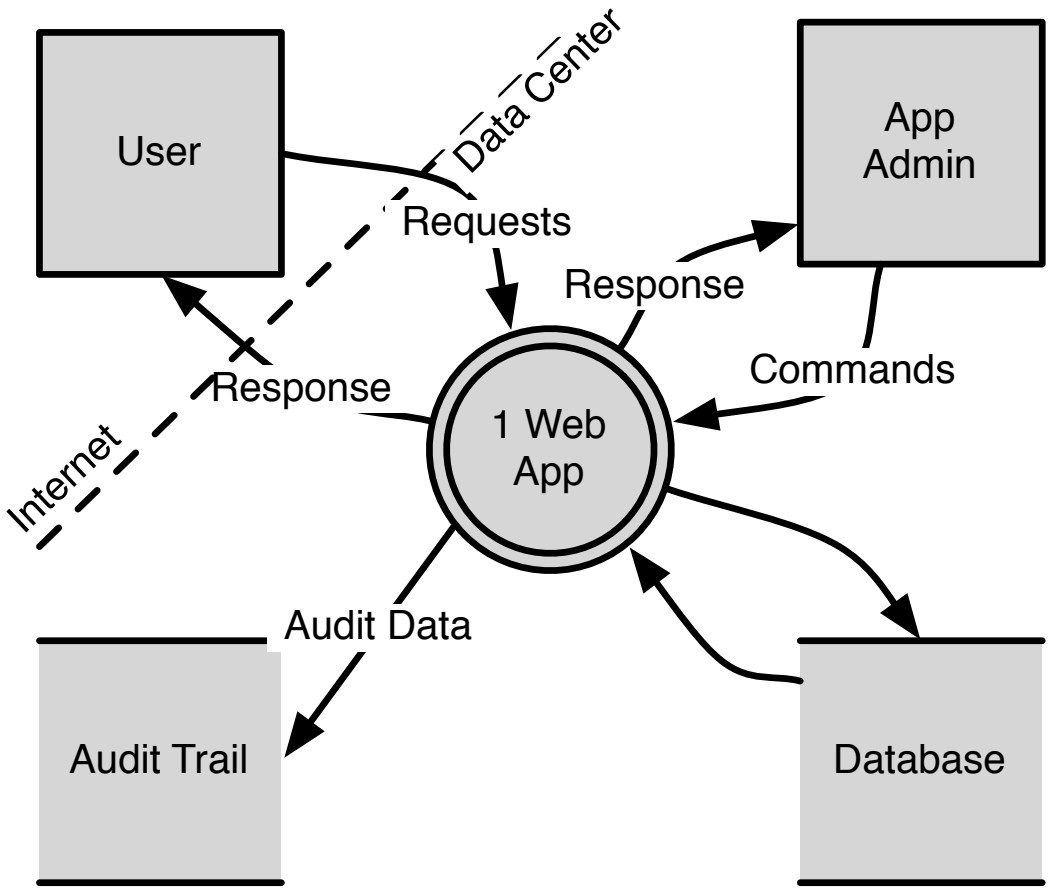
- Data flow represents data transferred between elements

DFD Privilege Boundary

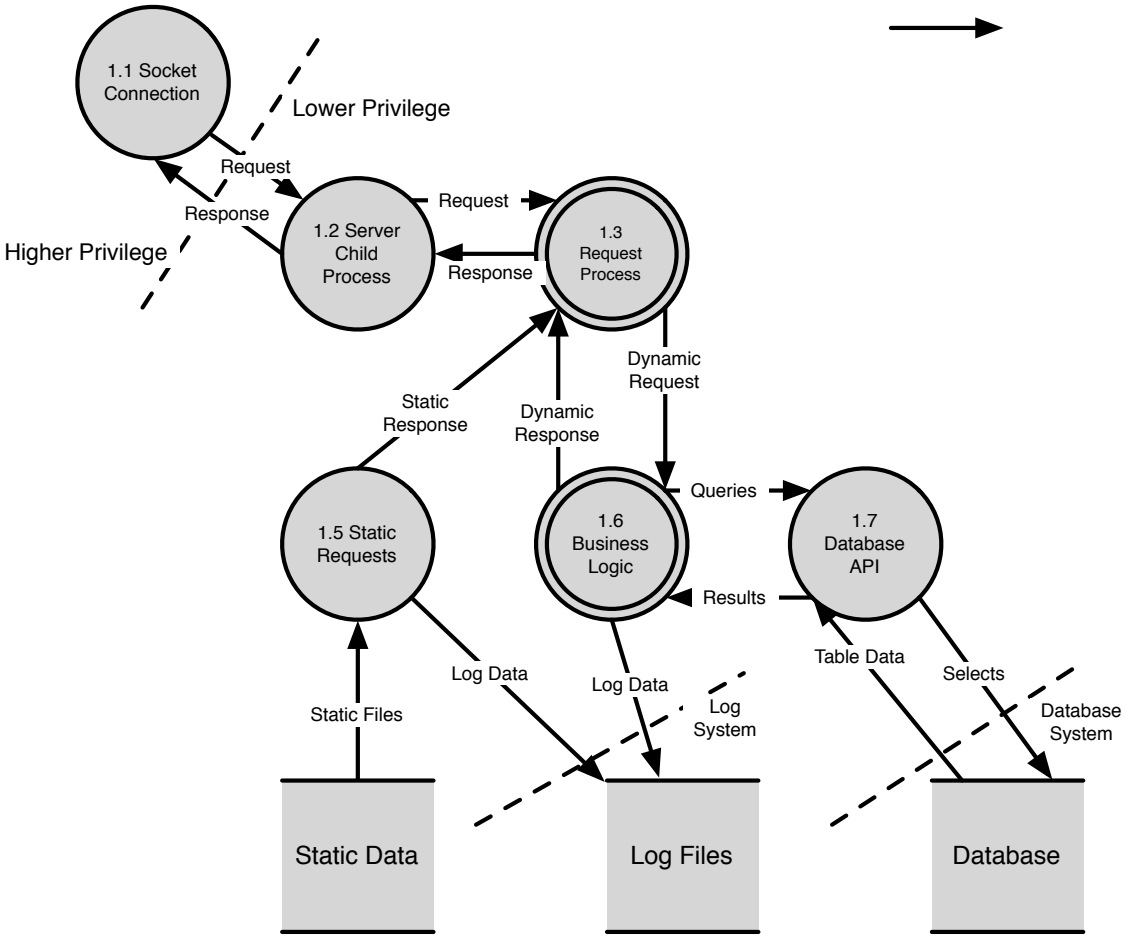
- Defines a boundary between nodes with a different privilege level



DFD Level-o Example



DFD Level-1 Example



Exploring DFD Levels

- Each complex process can be broken down and explored to many more detailed levels
  - More information is not always helpful
  - Significant time investment with too many levels
  - Avoid "Analysis Paralysis"

Threat Model Process

1. Understanding the Attacker's View
2. Characterize the Security of the System
3. **Determining Threats**

Determining Threats

- Identifying threats to the system
- Investigating threats with threat trees
- Mitigating vulnerabilities

Identifying Threats

- Identify attacker goals might try to achieve
  - Acquire an asset
  - Misuse an asset
- Use the system's assets as a starting point
  - Look at high-level attack goals for each asset
- Can the asset be modified?
  - By a non-privileged user?
  - Without detection or auditing?

Identifying Threats

Data Needed:

1. Numerical ID
2. Name
3. Description
4. STRIDE Classification
5. Entry Points
6. Assets
7. Mitigation

Determining Threats using STRIDE

- Remember STRIDE defines attacker objectives

  Spoofing an identity

  Tampering with data

  Repudiation

  Information Disclosure

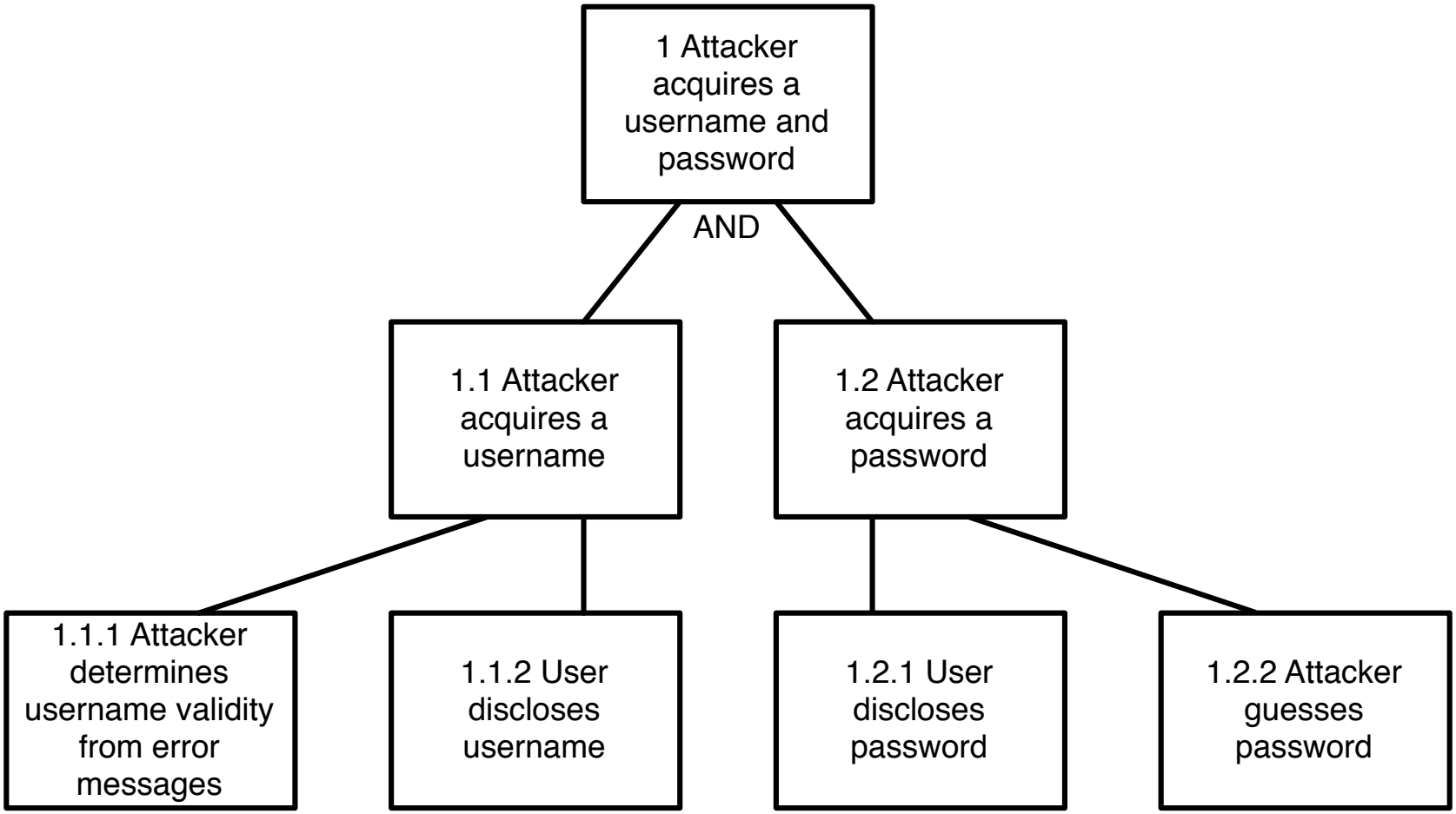
  Denial of Service

  Elevation of Privilege

Threat trees

- Threats trees show how a threat might be accomplished
  - Attack Paths show the route for a threat
  - Consists of a "root threat" and mitigated and unmitigated conditions
- Can be shown visually in a tree structure
- Can be written in an outline form

Threat Tree Example



Threat Tree Example

1. Attacker acquires a username and password
   - 1.1 (AND) Attacker acquires username
     - 1.1.1 Attacker determines username validity from error messages
     - 1.1.2 User discloses username
   - 1.2 (AND) Attacker acquires password
     - 1.2.1 User discloses password
     - 1.2.2 Attacker guesses password

Threat Trees

- Useful for security testing
  - Focus on specific attack paths from the tree
- Show that unmitigated conditions are vulnerabilities
  - Attack paths with one or more unmitigated conditions

Data Needed:

1. Threat ID
2. Tree Nodes

Mitigating Vulnerabilities

- Remember that a threat is not a vulnerability
  - Source of confusion
  - A Vulnerability is weakness that would allow an attacker to take some advantage over an asset
  - A Threat is what an attacker might try to do

Mitigating Vulnerabilities

- Using the threat tree, identify attack paths with unmitigated conditions
  - These are the vulnerabilities that must be addressed
  - Vulnerabilities are the useful outcome from model
- Vulnerabilities should be documented outside of the threat model and referenced
  - Use the application's bug tracking system

Mitigating Vulnerabilities

Data Needed:

1. Numerical ID
2. Name
3. Description
4. DREAD Rating
5. Corresponding Threat
6. Assigned Bug ID

Threat Model Process Review

1. Understanding the Attacker's View
   - Goals, data flow, entry points, assets, trust levels

   Output: Detailed information on app and access
2. Characterize the Security of the System
   - Background info for data flow and processing

   Output: Data Flow Diagram and Threat tree
3. Determining Threats
   - Identifying threats and vulnerabilities

   Output: Vulnerabilities to be mitigated

Threat Modeling Exercise

- Handout "Exercise #3"
- Work in groups of two
- Look for "**Instructions**" sections
  - Fill in table data
  - Draw missing primitives in the DFD
- Take 15 minutes

Review

- Assets, Threats, Vulnerabilities
- STRIDE threat classifications
- DREAD risk ratings
- Threat Modeling
  - Attacker's Perspective
  - Understand the system
  - Find the vulnerabilities