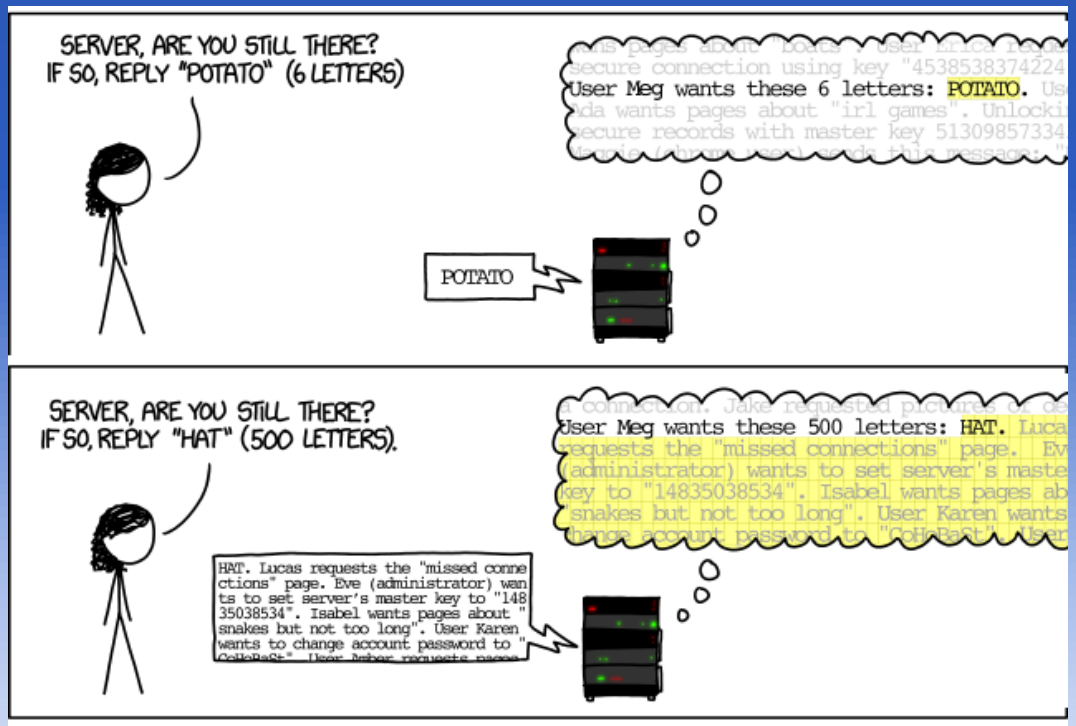
CS 493
Secure Software Systems
Ch 8 Vulnerability mapping



Adapted Heartbleed explanation XKCD

Dr. Williams
Central Connecticut State University

Goals

- Construct use case and misuse case diagrams.
- Identify overlapping security concerns in a use case overview diagram.
- Construct supporting documents in UML with the addition of security concerns.
- Identify and prioritize system vulnerabilities.
- Manage the required documentation to provide a complete business specification of the system.

Use Case Construction and Extension

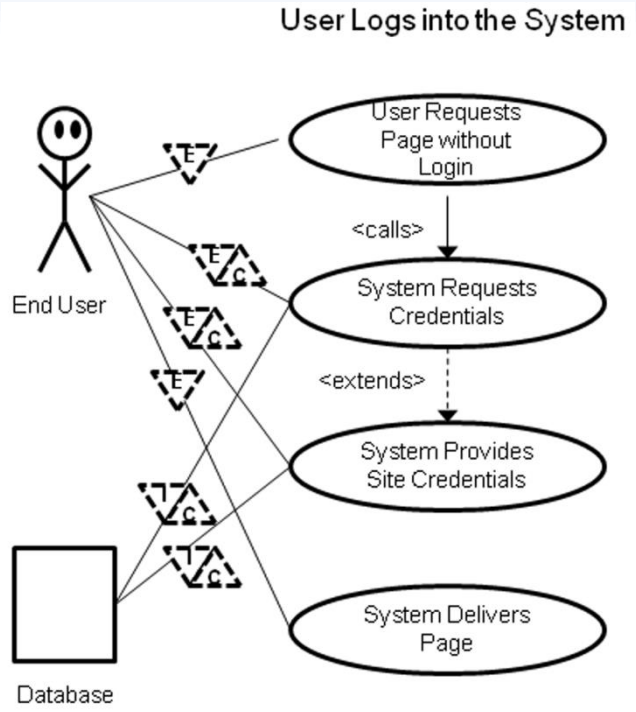
- The first step in moving from a listing of system requirements to an actualized and deployed system is the process of use case mapping.
- A **use case** is a translation of functional requirements into a visual map of activity that details the steps of arriving at a measurable system outcome in more granular and explicit fashion.

Use Case Construction and Extension

Use cases involve three primary components:

1. An **actor** is a person, external system, or entity that plays a role in the performance of the functional task described in the use case.
2. A **procedure** is a step performed to achieve the outcome of the system specified by the functional requirement.
3. An **association** is a relationship between actors and procedures. For actors and procedures, this is represented by a directional arrow specifying the instantiation of the next step in the process for the system.

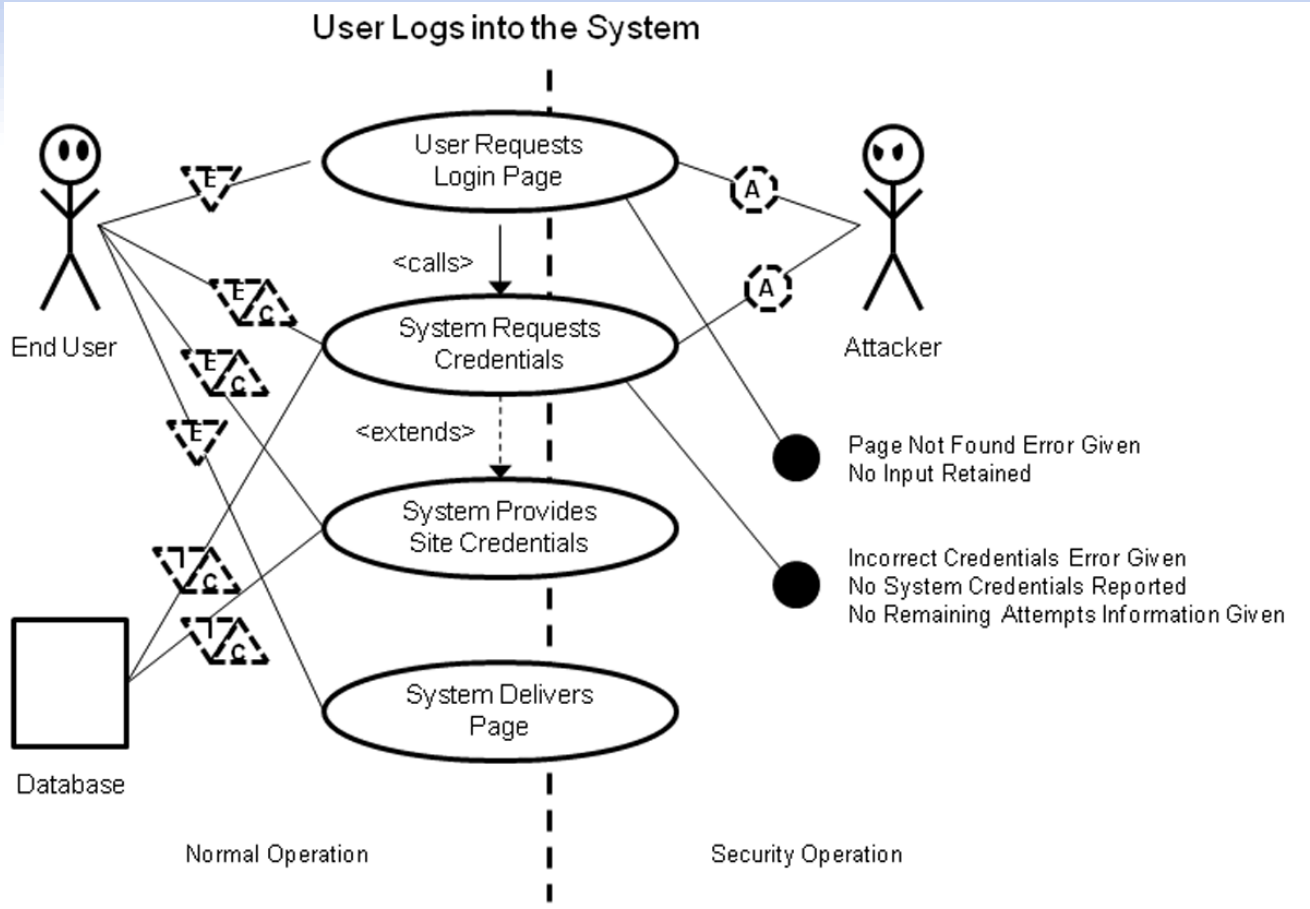
Sample Use Case



Managing Misuse

- This is an attempt to elicit security requirements by considering what a malicious actor could do within the context of the system.
- **Misuse Management Method (MMM)** is done by first, keeping all of the actors in your use case to the left of the procedures and keeping the procedures in the middle of the diagram. Draw a dotted line down through the middle of your procedures to separate the normal operating case from the diagram for malicious attacks on the system. The left side will retain the use case properties and information. The right side will contain the analysis of security needs and the procedural extensions necessary to round out the functionality in the use case to manage attacks. (The next slide has a sample)

Misuse Management Method (MMM)



Off the Map

- After the individual use cases have been completed and the misuse information has been added to the diagrams, it is possible to start mapping the entire system.
- The easiest way to do this is to create a new **use case overview diagram** and add the central functionality defined in each of the individual use cases.

In class analysis

Break into groups and analyze the following requirements:

x. Users can register for a free account

17. Free accounts must still register contact information.

- **Create use case (x & 17 as one)**
- **Notate external, internal, confidential**
- **Apply Misuse Management Method (MMM)**

In class analysis

Break into groups and analyze the following requirements:

18. Registered users can search for a question they are allowed to answer.

19. Registered users can vote A or B on a question they are allowed to answer.

- **Create use cases**
- **Notate external, internal, confidential**
- **Apply Misuse Management Method (MMM)**

In class analysis

Break into groups and analyze the following requirements:

24. Users can answer a question when they are allowed to access it by the user who posted the question.

26. Search results will already be filtered by the user's permissions when they are returned.

- **Create use cases**
- **Notate external, internal, confidential**
- **Apply Misuse Management Method (MMM)**

Sequence Diagrams and Class Analysis

- A **sequence diagram** is a detailed breakdown of the communication that will occur between actors and system objects or components.
- A sequence diagram is most compatible with object-oriented systems, because it allows relatively straightforward mapping from use cases by applying object models to the instantiation.
- A **class** is a template for behavior and variable usage; an **object** is an instantiation of this template.

Sequence Diagrams and Class Analysis

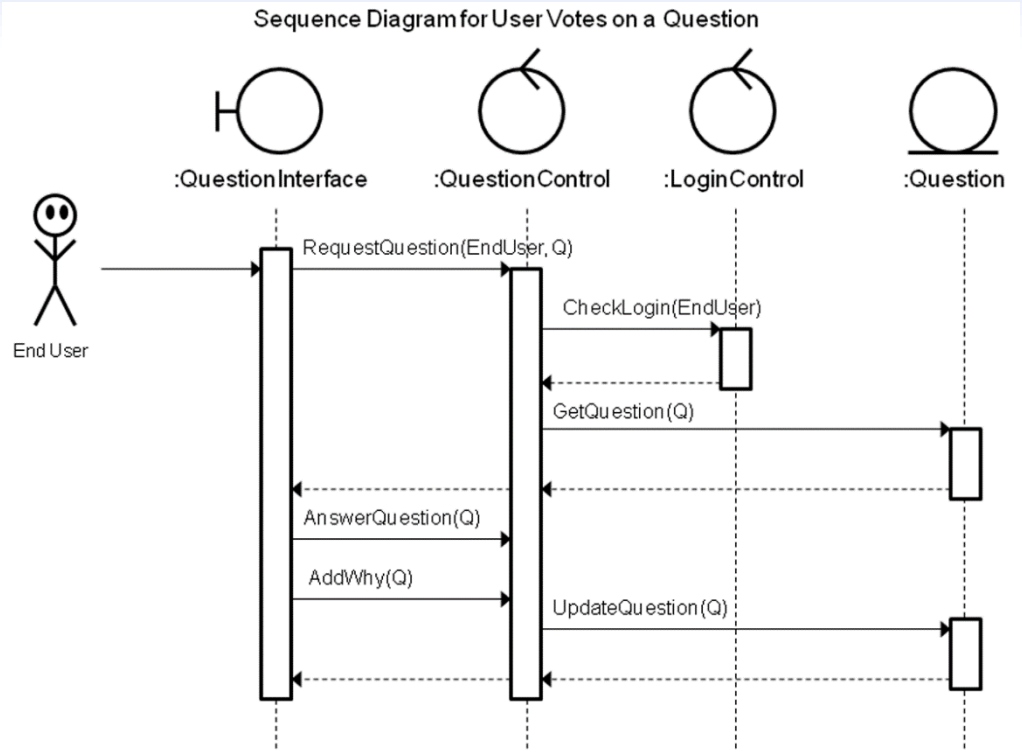
There are several types of classes that will be determined in a sequence diagram:

- An **entity class** is, broadly, a storage class. The objects it generates are the housing for most of the data in a system.
- A **boundary class** is primarily responsible for handling interactions between the actors and the system.
- A **control class** is a coordinator for the system. These classes insulate entity classes from changing business rules and policies, effectively modularizing the system to a degree.

Sequence Diagrams Process Steps

1. Choose a single use case.
2. Write out each process of the use case in detail.
3. Specify a boundary class between the system and each actor.
4. Specify a single control class for the use case.
5. Specify an entity class for each object referenced in the use case.
6. Specify a control class for any generalized use case referenced.
7. Align the use case steps next to the row of actors and classes.
8. Determine lifetimes for the objects based on a complete sequence of messages to and from the boundary class indicating a single overall transaction in the system.
9. Refine your classes for any functionality that is too complex for a single class.
10. Refine your classes for any functionality that is too complex for a single class. (Sample next slide)

Sample Sequence Diagram



Data Planning

- The goal of data planning is to establish the database construct that will be needed to support the system being developed.
- An **entity-relationship model** (**E-R model**) is a relational diagram used to establish tables, table attributes, and relationships within a system.
- The cardinality is the number of relationships that can occur from one table to another.

Knowing Your Boundaries

There are several good restrictions for boundary classes:

- A boundary class is not allowed to directly execute any input.
- A boundary class on the client machine is not allowed to divulge or contain privacy data that is not entered by the client or sent by the internal control class.
- A boundary class must authenticate the communicating control class to which it is connected.

Knowing Your Boundaries

- A control class on the outside of the trust boundary should never be allowed to interface directly with a control class on the inside of the trust boundary.
- A control class must authenticate the boundary classes from which it receives a message.
- A control class must provide authentication to the boundary class to which it is communicating.
- A control class must evaluate or process input from a boundary class before directly executing any information coming from a boundary class.
- A control class must provide authentication to an entity class from which it is requesting privacy data or mission-critical data.
- A control class that is not trusted cannot directly communicate to an entity class that is trusted.
- All data members of a control class must be private or protected.

Entity Classes

- An entity class can divulge information only to a control class.
- An entity class on a client machine may not directly access information inside the system trust boundary.
- An entity class may not communicate with a boundary class.
- An entity class housing private, confidential, or mission-critical data must authenticate the control class with which it is communicating.
- An entity class has the right to refuse to divulge information.
- An entity class is not allowed to have public data members.

Communication, Activity, and State Diagrams

- A **communication diagram** is a collaboration diagram, and is an alternate view of the sequence diagram in which all of the interactions between the classes are mapped as function calls for the class.
- An **activity diagram** is a type of support diagram in which the workflow of a use case is mapped out in flowchart format with a well-defined starting point and clear end points.
- A **state diagram** is a model of the lifespan of an object within the system.

Vulnerability Mapping

- The overall goal of performing vulnerability mapping is to determine the most likely locations within the system in development where an attacker will strike.
- To start vulnerability mapping, identify the input locations of the system, the internal communications, and the interprocess communications.

Vulnerability Mapping

The following basic classification system will work:

- **V3:** This is the highest level of vulnerability.
- **V2:** This is the moderate level of vulnerability.
- **V1:** This is the lowest priority level of vulnerability.

Complete Business System Specifications

The specification for the system is now at its most detailed level for the business side of the system development.

- An executive summary of the system overview, similar to what is presented as the project overview for the case project
- The collection of nonfunctional requirements
- The collection of functional requirements and associated use cases
- The sequence diagrams constructed from the use cases
- The use case overview diagram
- The class analysis diagram
- The entity-relationship model
- Optional activity diagrams
- Optional state diagrams
- Optional communication diagrams
- The vulnerability map/attack surface document

Summary

- This phase of development is concerned with translating the requirements initially gathered into a model of the system that is to be developed.
- UML is the common language for system modeling.
- The vulnerability map is the first step of the attack surface model for the system.
- The final attack surface will be used to make a decision on whether or not the system upholds the security criteria.

In class analysis

Break into groups and analyze the following requirements:

x. Users can register for a free account

17. Free accounts must still register contact information.

- **Create sequence diagram**
- **Note vulnerability analysis**

In class analysis

Break into groups and analyze the following requirements:

18. Registered users can search for a question they are allowed to answer.

19. Registered users can vote A or B on a question they are allowed to answer.

- **Create sequence diagram**
- **Note vulnerability analysis**

In class analysis

Break into groups and analyze the following requirements:

24. Users can answer a question when they are allowed to access it by the user who posted the question.

26. Search results will already be filtered by the user's permissions when they are returned.

- **Create sequence diagram**
- **Note vulnerability analysis**

In class exercises revisited

Bob's Pizza Shack

In groups consider this scenario

Bob is a small business owner of Bob's Pizza Shack and wants to create a website to allow online credit card delivery orders

1) Identify 4 functional requirements
    1) Create use cases and note internal, external, confidential
    2) Apply MMM
    3) Create sequences and note vulnerability analysis

Alice's Online Bank

In groups consider this scenario

- Alice opens Alice's Online Bank (AOB)
  - Internal use bank employees
  - Interoperates with another bank
  - Interacts with customers
    - Web
    - Mobile app

1) Identify 4 functional requirements
   1) Create use cases and note internal, external, confidential
   2) Apply MMM
   3) Create sequences and note vulnerability analysis

Location based social media app

In groups consider this scenario Open source group wants to create a mobile app to allow groups to communicate/find each other in public demonstrations/protests

- Communication internet, as well as, P2P (WiFi/Bluetooth) in case internet cut off – so if person you want to contact is on other side of crowd and no internet as long as P2P network can be established with app can reach somebody outside your immediate vicinity via the P2P network
- Should be able to communicate securely messages and images to people you identify within your group (group as whole or direct)
- Should be able to share GPS location with people in group (group as whole or direct)