Lecture 4

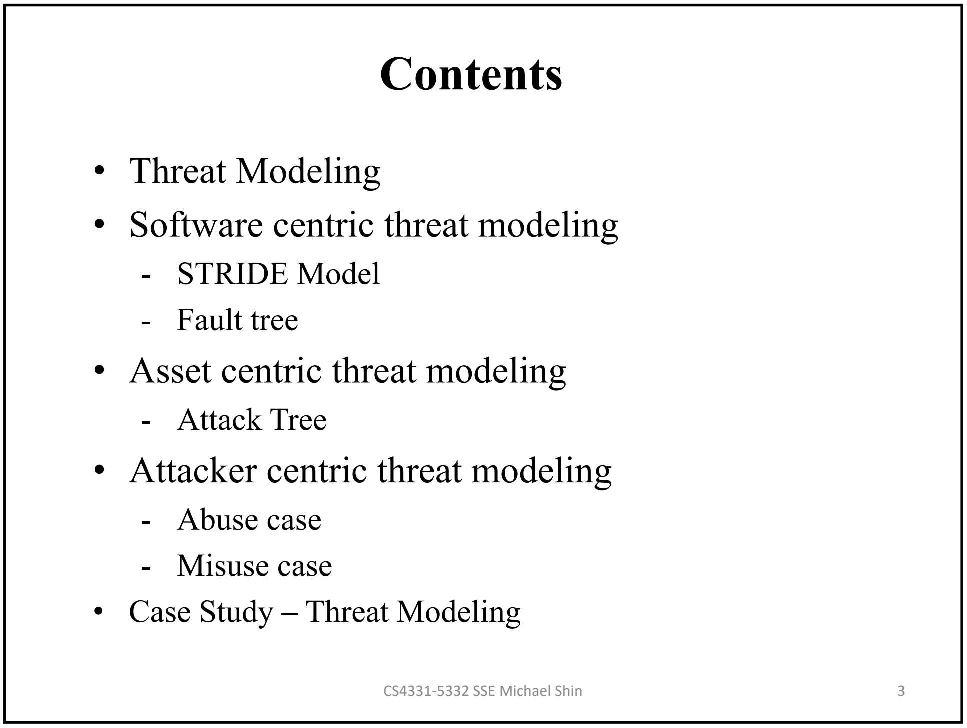
Threat Modeling

1

References

- Wikipedia, https://en.wikipedia.org/wiki/Threat_modelThreat Modeling
- Kohnfelder and Praerit Garg, “The threats to our products” April 1, 1999.
- Adam Shostack, *“Threat Modeling: Designing for Security”,* Wiley, 2014.
- Wikipedia, https://en.wikipedia.org/wiki/Fault_tree_analysis
- Bruce Schneier, “Attack Trees,” Dr. Dobb’s Journal, December 1999.
- John McDermott and Chris Fox, “Using Abuse Case Models for Security Requirements Analysis,” 1999.
- Guttorm Sindre and Andreas Opdahl, “Eliciting Security Requirements with Misuse Cases,” 2005.

2

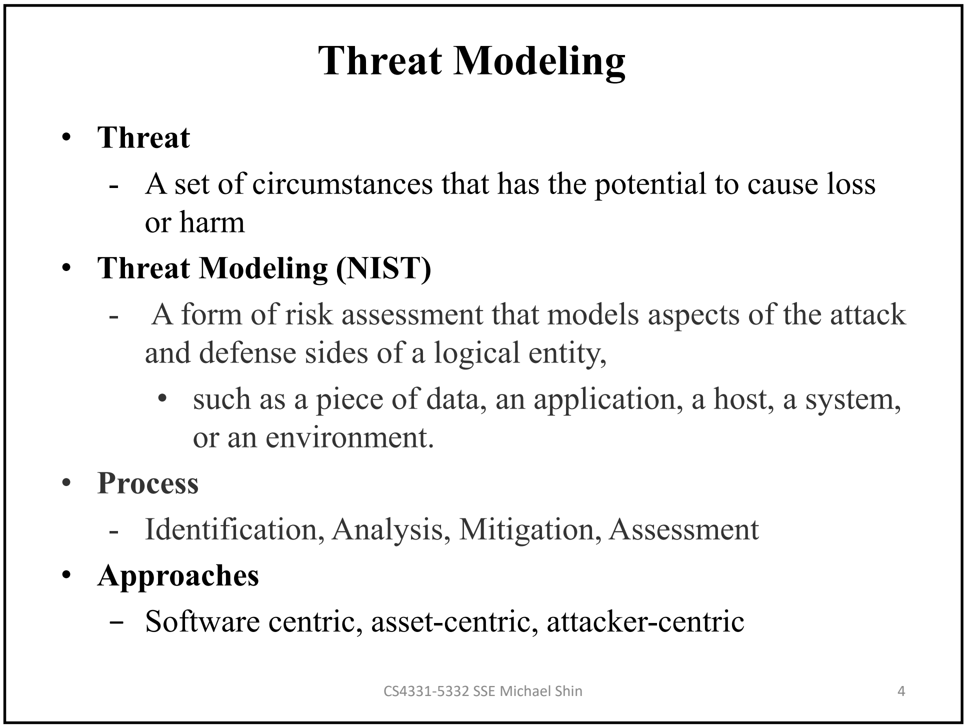


Contents

- Threat Modeling
- Software centric threat modeling
  - STRIDE Model
  - Fault tree
- Asset centric threat modeling
  - Attack Tree
- Attacker centric threat modeling
  - Abuse case
  - Misuse case
- Case Study – Threat Modeling

CS4331-5332 SSE Michael Shin 3

3

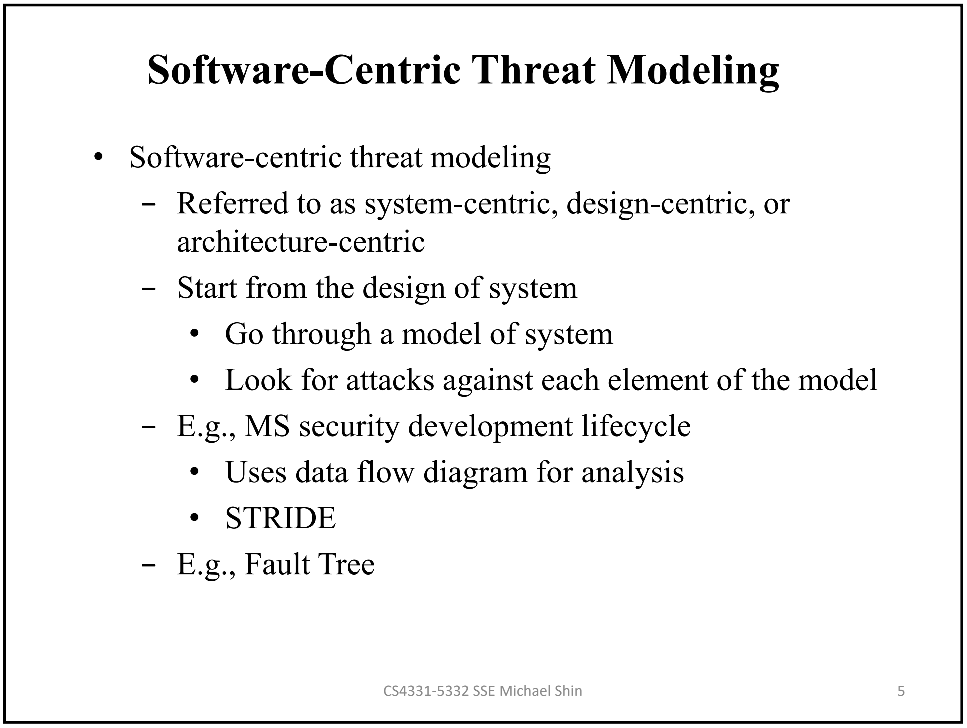


Threat Modeling

- **Threat**
  - A set of circumstances that has the potential to cause loss or harm
- **Threat Modeling (NIST)**
  - A form of risk assessment that models aspects of the attack and defense sides of a logical entity,
    - such as a piece of data, an application, a host, a system, or an environment.
- **Process**
  - Identification, Analysis, Mitigation, Assessment
- **Approaches**
  - Software centric, asset-centric, attacker-centric

CS4331-5332 SSE Michael Shin 4

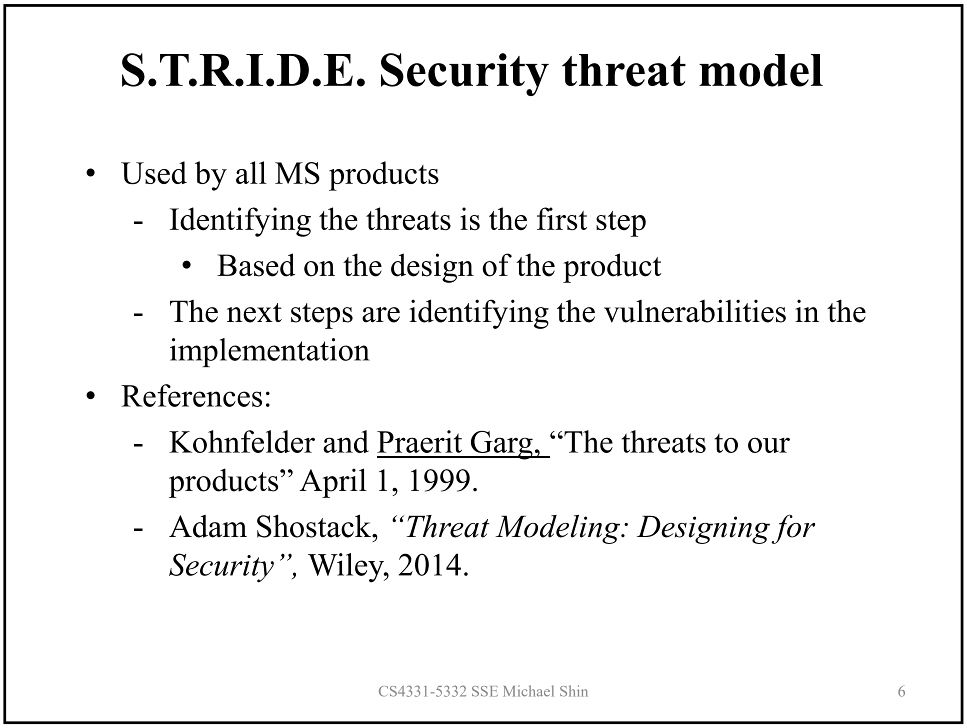
4



Software-Centric Threat Modeling

- Software-centric threat modeling
  - Referred to as system-centric, design-centric, or architecture-centric
  - Start from the design of system
    - Go through a model of system
    - Look for attacks against each element of the model
  - E.g., MS security development lifecycle
    - Uses data flow diagram for analysis
    - STRIDE
  - E.g., Fault Tree

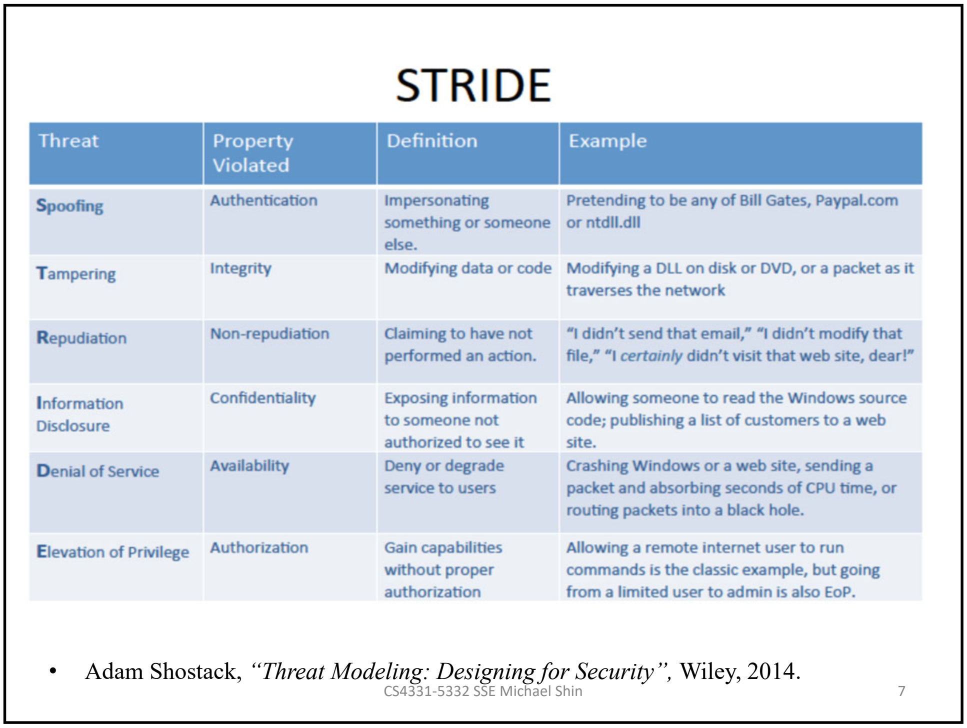
5



S.T.R.I.D.E. Security threat model

- Used by all MS products
  - Identifying the threats is the first step
    - Based on the design of the product
  - The next steps are identifying the vulnerabilities in the implementation
- References:
  - Kohnfelder and Praerit Garg, “The threats to our products” April 1, 1999.
  - Adam Shostack, *“Threat Modeling: Designing for Security”,* Wiley, 2014.

6

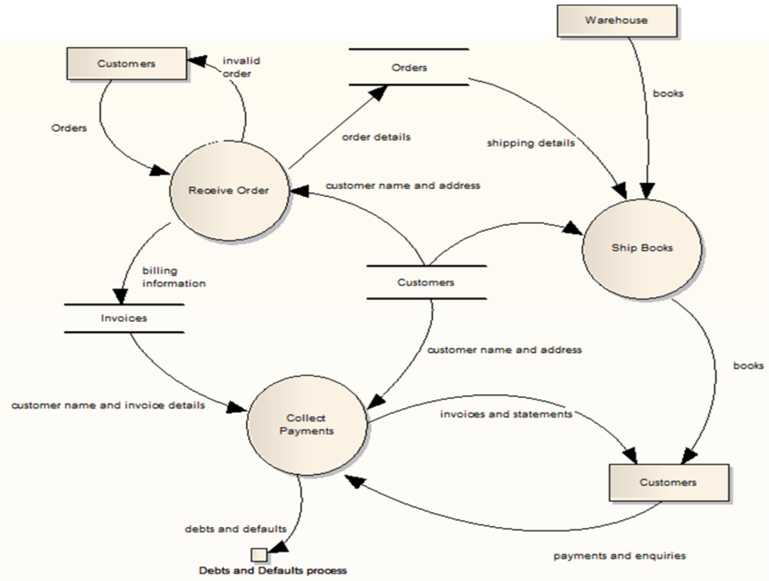


STRIDE

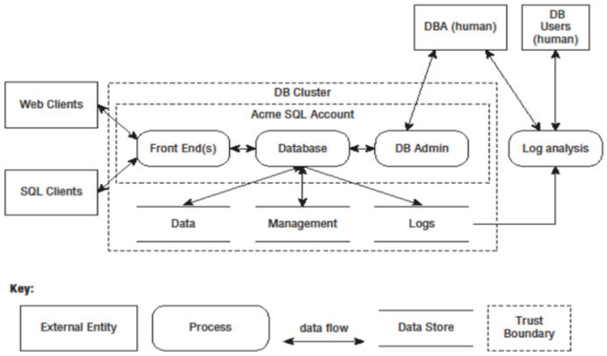
Threat	Property Violated	Definition	Example
Spoofing	Authentication	Impersonating something or someone else.	Pretending to be any of Bill Gates, Paypal.com or ntdll.dll
Tampering	Integrity	Modifying data or code	Modifying a DLL on disk or DVD, or a packet as it traverses the network
Repudiation	Non-repudiation	Claiming to have not performed an action.	"I didn't send that email," "I didn't modify that file," "I *certainly* didn't visit that web site, dear!"
Information Disclosure	Confidentiality	Exposing information to someone not authorized to see it	Allowing someone to read the Windows source code; publishing a list of customers to a web site.
Denial of Service	Availability	Deny or degrade service to users	Crashing Windows or a web site, sending a packet and absorbing seconds of CPU time, or routing packets into a black hole.
Elevation of Privilege	Authorization	Gain capabilities without proper authorization	Allowing a remote internet user to run commands is the classic example, but going from a limited user to admin is also EoP.

- Adam Shostack, *"Threat Modeling: Designing for Security"*, Wiley, 2014.

Data Flow Diagram for Order book



Data Flow Diagram (Example)



- **Trust Boundary**
  - A boundary where program data or execution changes its level of trust
- Adam Shostack, Threat Modeling: Designing for Security, Wiley, 2014.

Example Threat Tracking Tables

Diagram Element	Threat Type	Threat	Bug ID
Data flow #4, web server to business logic	Tampering	Add orders without payment checks	4553 “Need integrity controls on channel”
	Info disclosure	Payment instruments sent in clear	4554 “need crypto” #PCI

Threat Type	Diagram Element(s)	Threat	Bug ID
Tampering	Web browser	Attacker modifies our JavaScript order checking	4556 “Add order-checking logic to server”
Elevation of Privilege	Data flow #2 from browser to server	Failure to authenticate	4557 “~~Add~~ enforce HTTPS everywhere”

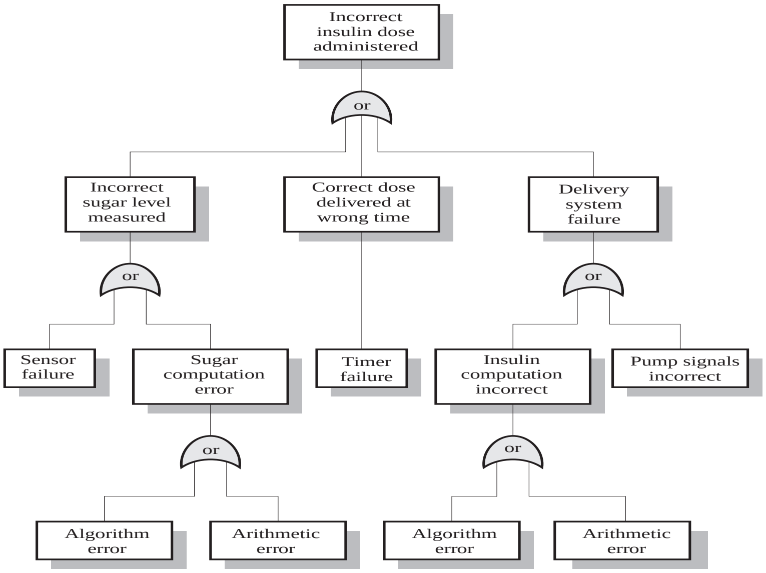
Both are fine, help you iterate over diagrams in different ways

- *Threat Modeling: Designing for Security* (Wiley, 2014) by Adam Shostack

Fault Tree

- Top-down, deductive failure analysis
  - An undesired state of a system is analyzed
    - Using Boolean logic to combine a series of lower-level events
    - AND, OR, Exclusive OR, …
    - Basic event, intermediate event, undeveloped event, …
- Safety cannot be measured when a system is tested
  - Safety assurance concentrates on faults with hazard potential
- Proof by contradiction
  - Pre-conditions for a hazard state cannot hold

Example of Fault tree for safety



Incorrect insulin dose administered

or

Incorrect sugar level measured

Correct dose delivered at wrong time

Delivery system failure

or

or

Sensor failure

Sugar computation error

Timer failure

Insulin computation incorrect

Pump signals incorrect

or

or

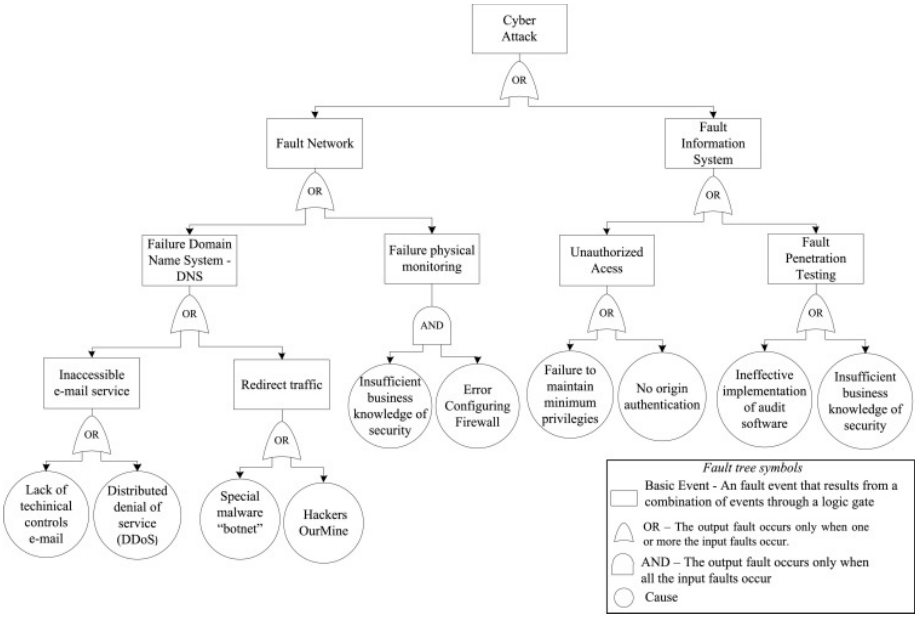
Algorithm error

Arithmetic error

Algorithm error

Arithmetic error

Example of fault tree analysis for security



Cyber Attack

OR

Fault Network

Fault Information System

OR

Failure Domain Name System - DNS

Failure physical monitoring

OR

Unauthorized Acess

Fault Penetration Testing

OR

AND

OR

OR

Inaccessible e-mail service

Redirect traffic

Insufficient business knowledge of security

Error Configuring Firewall

Failure to maintain minimum privilegies

No origin authentication

Ineffective implementation of audit software

Insufficient business knowledge of security

OR

OR

Lack of techinical controls e-mail

Distributed denial of service (DDoS)

Special malware "botnet"

Hackers OurMine

Fault tree symbols

□ Basic Event - An fault event that results from a combination of events through a logic gate

△ OR – The output fault occurs only when one or more the input faults occur.

AND – The output fault occurs only when all the input faults occur

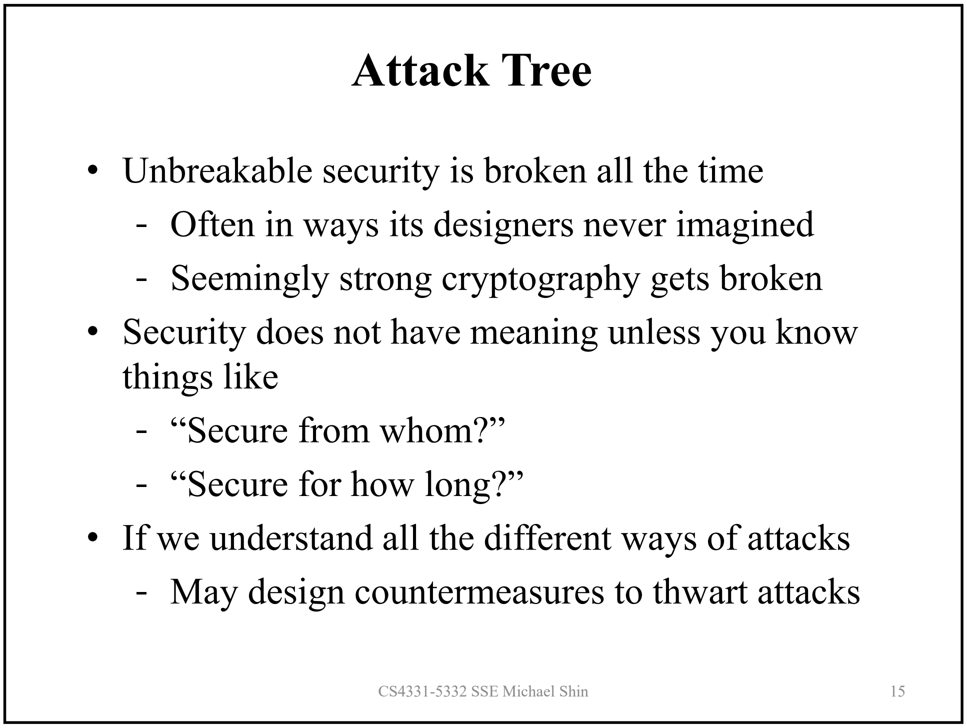
○ Cause

13

Asset-Centric Threat Modeling

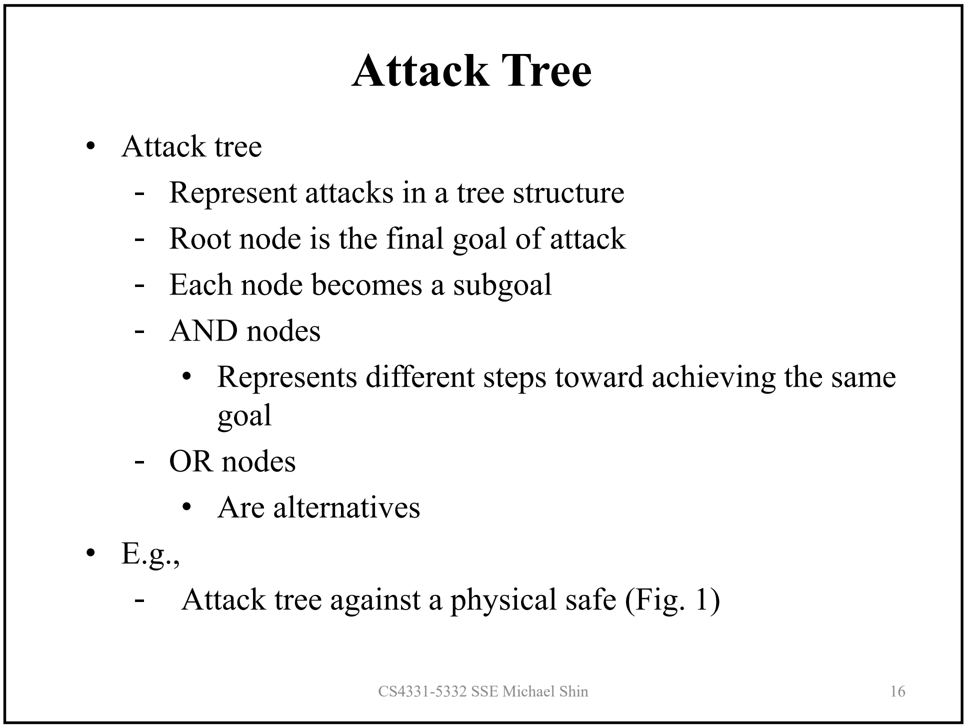
- Asset-centric threat modeling
  - Start from assets entrusted to a system
  - Assets
    - Any data, device, or other component supporting information-related activities
    - Assets include hardware, software, and secret information
  - E.g., Attack Tree
    - Bruce Schneier, "Attack Trees," Dr. Dobb's Journal, December 1999

14



Attack Tree

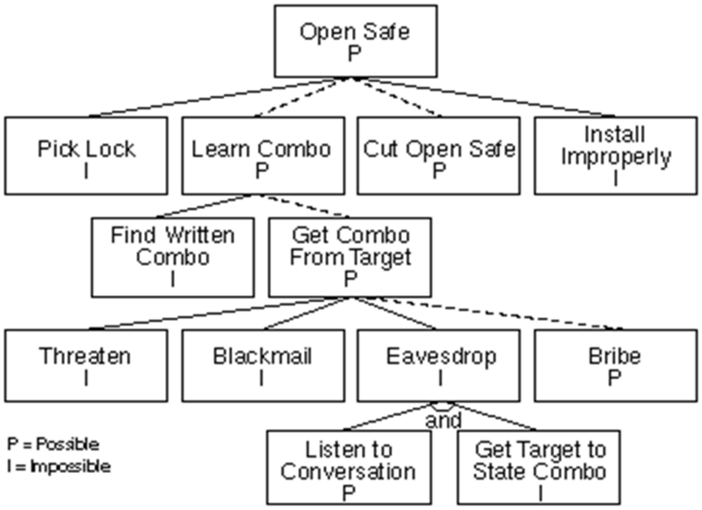
- Unbreakable security is broken all the time
  - Often in ways its designers never imagined
  - Seemingly strong cryptography gets broken
- Security does not have meaning unless you know things like
  - “Secure from whom?”
  - “Secure for how long?”
- If we understand all the different ways of attacks
  - May design countermeasures to thwart attacks



Attack Tree

- Attack tree
  - Represent attacks in a tree structure
  - Root node is the final goal of attack
  - Each node becomes a subgoal
  - AND nodes
    - Represents different steps toward achieving the same goal
  - OR nodes
    - Are alternatives
- E.g.,
  - Attack tree against a physical safe (Fig. 1)

Fig. 1 and Fig.2 Attack Tree and Possible Attacks



Attack Tree

- Once completed, it can be analyzed
  - I (impossible) and P (possible) : Fig. 2
  - No Special equipment and Special equipment: Fig. 3
  - Cost of attacks – Fig. 4
    - Tree with different costs assigned to the leaf nodes
    - These can propagate up the tree
    - OR nodes have the value of their cheapest child
    - AND nodes have the value of the sum of their children
  - Tree – can be used to determine where a system is vulnerable

Figure 3: Special Equipment Required

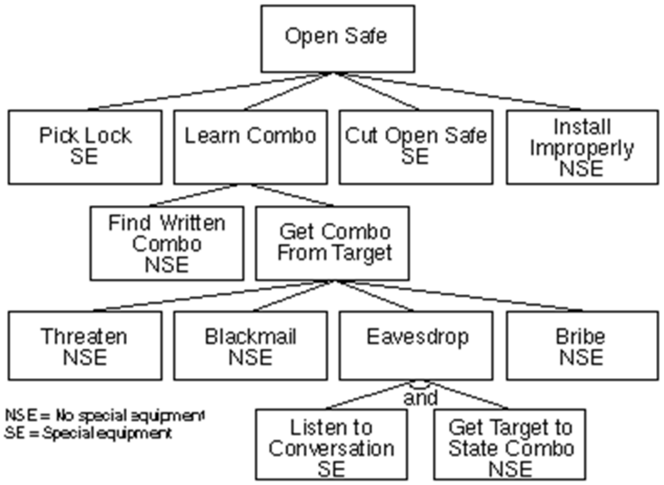
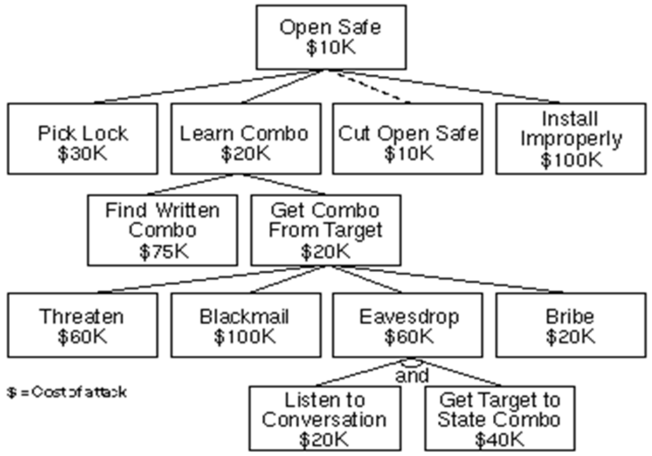
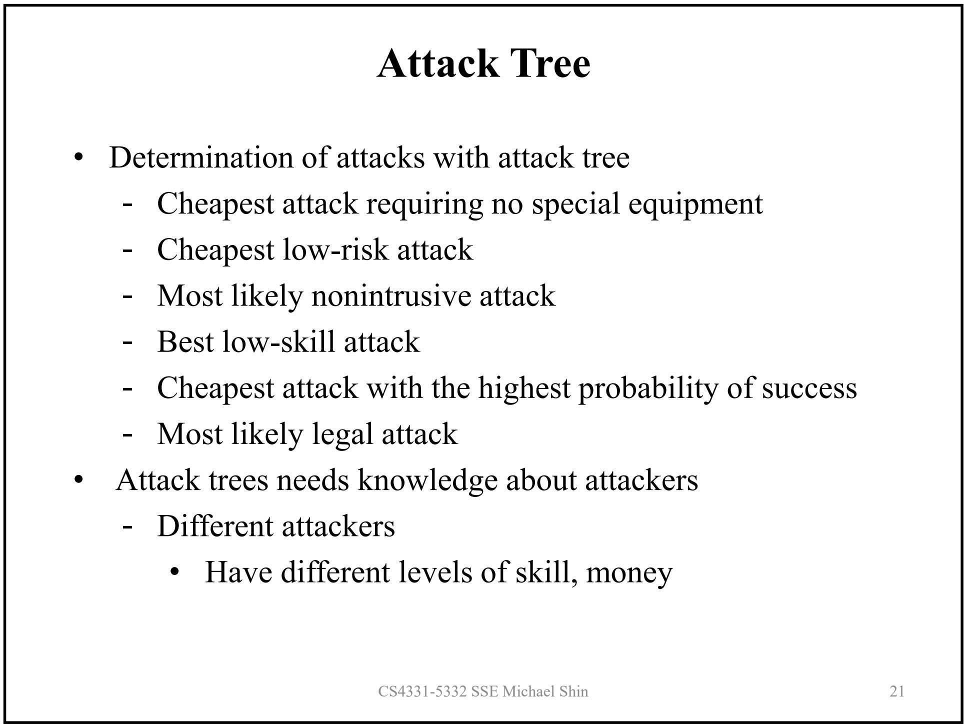


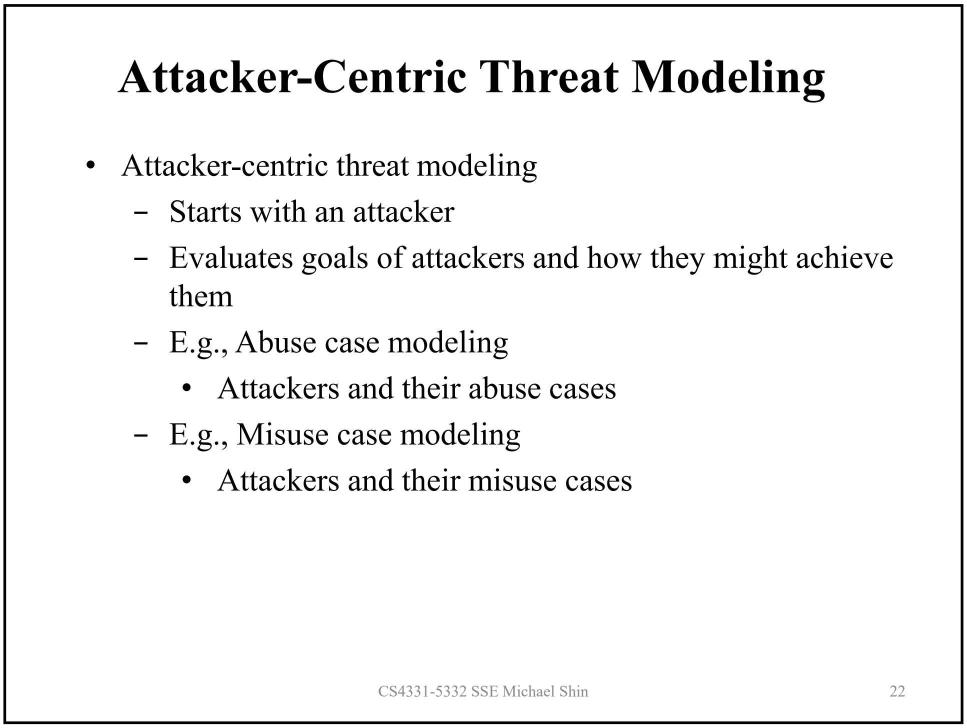
Figure 4: Cost of Attack





Attack Tree

- Determination of attacks with attack tree
  - Cheapest attack requiring no special equipment
  - Cheapest low-risk attack
  - Most likely nonintrusive attack
  - Best low-skill attack
  - Cheapest attack with the highest probability of success
  - Most likely legal attack
- Attack trees needs knowledge about attackers
  - Different attackers
    - Have different levels of skill, money



Attacker-Centric Threat Modeling

- Attacker-centric threat modeling
  - Starts with an attacker
  - Evaluates goals of attackers and how they might achieve them
  - E.g., Abuse case modeling
    - Attackers and their abuse cases
  - E.g., Misuse case modeling
    - Attackers and their misuse cases

Abuse Case

- John McDermott and Chris Fox, “Using Abuse Case Models for Security Requirements Analysis,” 1999.
- Notation - Represented with the UML use case diagram
  - Not shown on a use case diagram along with use cases
- Abuse Case
  - A specification of complete interaction between a system and actors
  - The results of the interaction
    - Harmful to the system, actors, or stakeholders in the system

Actor in Abuse Case Model

- Actor
  - External agents participating in use cases
    - Should not be the same actors in the use case
    - Actors act maliciously with different roles
      - E.g., Malicious Student
  - Give detailed descriptions in terms of actor’s resources, skills, objectives
    - E.g., Malicious Student
      - limited resources, technical skills, get score maliciously

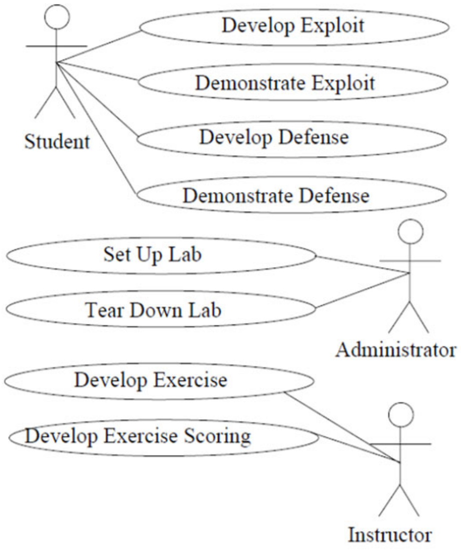
Constructing Abuse Case Model

- Identify the actors
  - Harmful users, any intruders
- Identify the abuse cases
  - Name abuse cases for each actor

Abuse Case – Internet-based Information Security lab

- Use case diagram (Fig.2)
  - Give students experiences
    - Security vulnerabilities, security testing, and defense
  - Remote lab
    - Servers, Resources, Targets, and Exercises
  - Actors and use cases
    - Student
      - Develop Exploit, Demonstrate Exploit, Develop Defense, Demonstrate Defense use cases
    - Administrator
      - Set Up Lab, and Tear Down Lab use cases
    - Instructor
      - Develop Exercise, Develop Exercise Scoring use cases

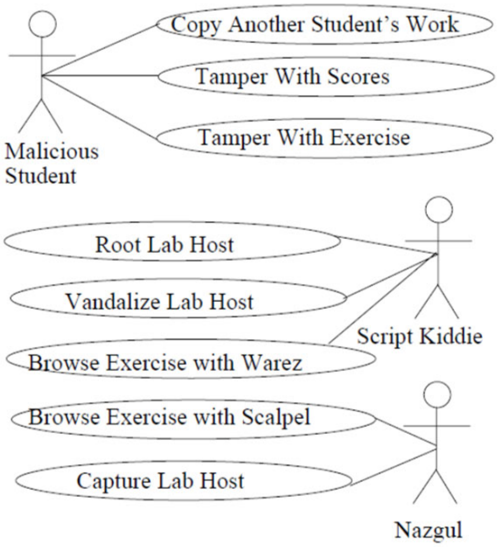
Figure 2. Use Case Diagram for an Internet-Based Information Security Laboratory



Abuse Case – Internet-based Information Security lab

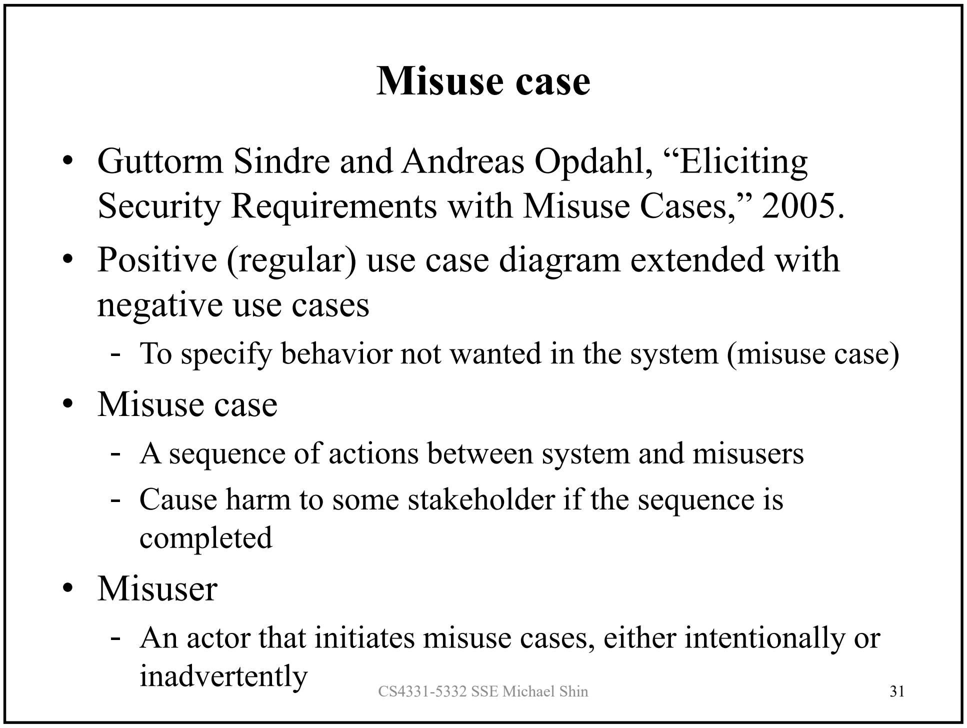
- Actors (Fig.3)
  - Malicious Student
    – limited resources, technical skills, get score maliciously
  - Script Kiddie
    - Resources (HW, SW, limited internet access, fellow script kiddie, personal funds or theft from an employer)
    - Skills: limited resources (using tools and techniques devised by other people)
    - Objectives: Criminal objectives (vandalism and theft), demonstrating technical skill
  - Nazgul
    - Resources: Groups with budgets, technical assistance from an organization
    - Skills: Superior technical skills
    - Objectives: Accomplishing objectives of organization (espionage, warfare, terrorism or similar harmful activities)

Figure 3. Abuse Case Diagram for an Internet-Based Information Security Laboratory



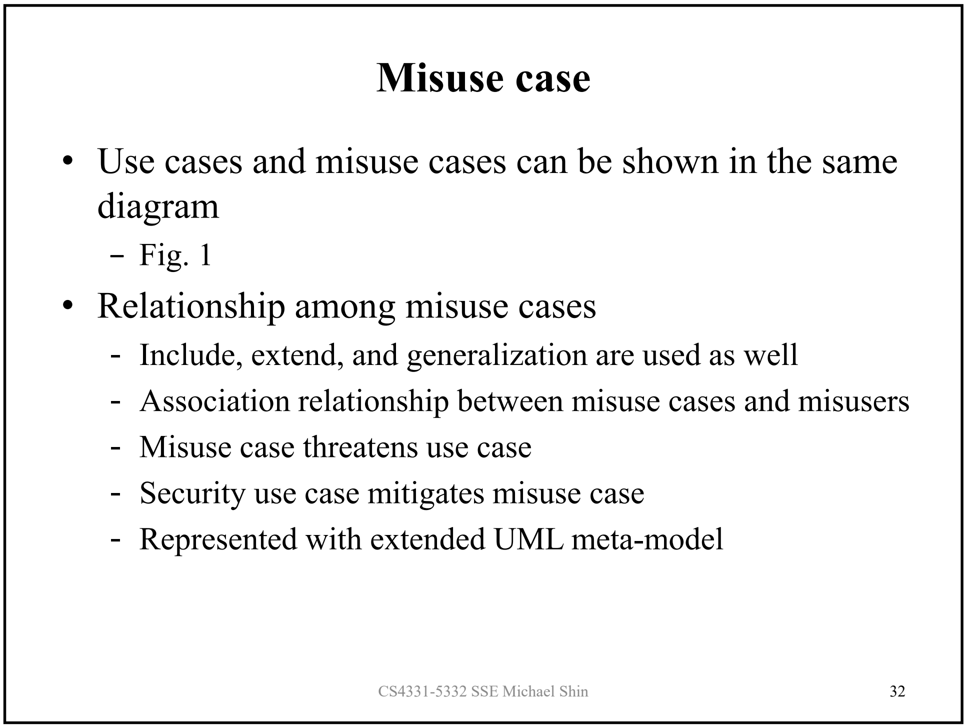
Applications of Abuse Case Modeling

- Requirements Phase
  - Used to show customers what will be prevented and what will not
    - Useful for security requirements elicitation
- Design
  - Can apply abuse cases through verifying process
    - Show each use case with the appropriate level of assurance
- Testing
  - Form a test team that has the same skills and resources as the actors associated with the abuse case
  - Let them exercise system features



Misuse case

- Guttorm Sindre and Andreas Opdahl, “Eliciting Security Requirements with Misuse Cases,” 2005.
- Positive (regular) use case diagram extended with negative use cases
  - To specify behavior not wanted in the system (misuse case)
- Misuse case
  - A sequence of actions between system and misusers
  - Cause harm to some stakeholder if the sequence is completed
- Misuser
  - An actor that initiates misuse cases, either intentionally or inadvertently



Misuse case

- Use cases and misuse cases can be shown in the same diagram
  - Fig. 1
- Relationship among misuse cases
  - Include, extend, and generalization are used as well
  - Association relationship between misuse cases and misusers
  - Misuse case threatens use case
  - Security use case mitigates misuse case
  - Represented with extended UML meta-model

Fig. 1 Example use and misuse cases for an e-store

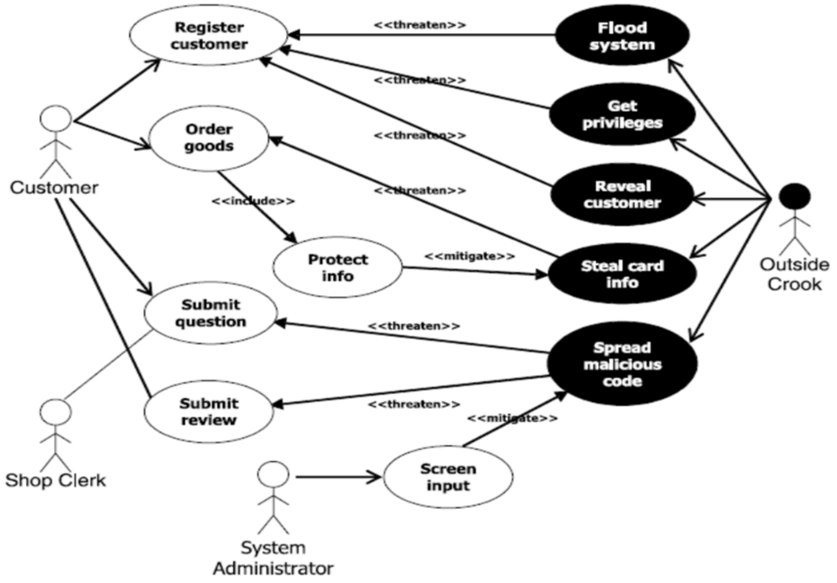
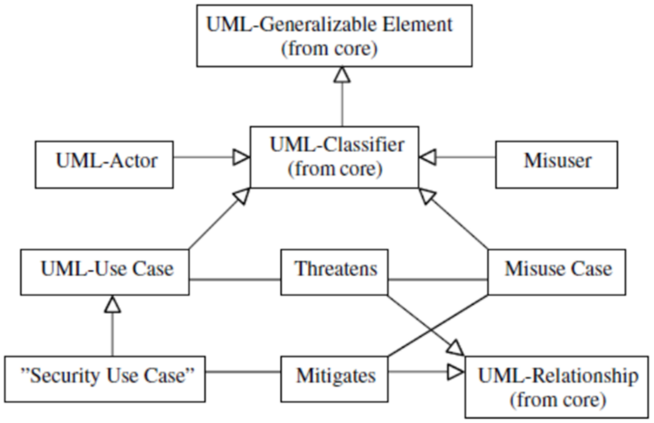


Fig. 2 Metamodel of the basic concepts presented in this section and their relation to the UML metamodel


UML-Generalizable Element (from core)
UML-Actor
UML-Classifier (from core)
Misuser
UML-Use Case
Threatens
Misuse Case
"Security Use Case"
Mitigates
UML-Relationship (from core)

Misuse case description

- Use case diagram give an overview of system functionality
  - Described in textual description using template
- Lightweight use case description
  - Embed description of misuse case within a use case template
  - Use case description template is extended with either threats field or a column
  - Table1 – T1 in a separate Threats item
  - Table2 – use case with a Threats column
  - Better to apply early in development when brainstorming threats
  - More appropriate for misuse cases less critical for overall security

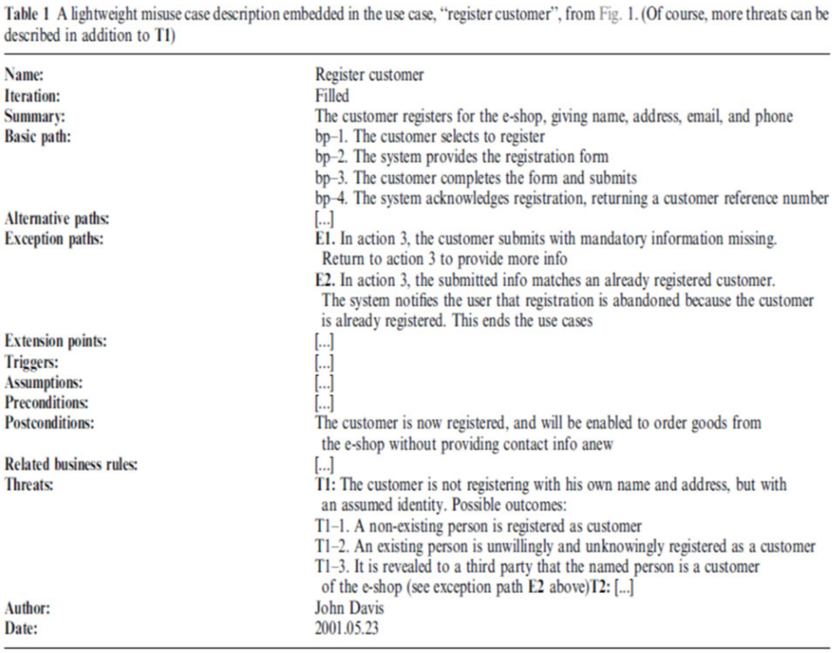


Table 1 A lightweight misuse case description embedded in the use case, "register customer", from Fig. 1. (Of course, more threats can be described in addition to T1)

Name:	Register customer
Iteration:	Filled
Summary:	The customer registers for the e-shop, giving name, address, email, and phone
Basic path:	bp–1. The customer selects to register bp–2. The system provides the registration form bp–3. The customer completes the form and submits bp–4. The system acknowledges registration, returning a customer reference number
Alternative paths:	[...]
Exception paths:	**E1.** In action 3, the customer submits with mandatory information missing. Return to action 3 to provide more info **E2.** In action 3, the submitted info matches an already registered customer. The system notifies the user that registration is abandoned because the customer is already registered. This ends the use cases
Extension points:	[...]
Triggers:	[...]
Assumptions:	[...]
Preconditions:	[...]
Postconditions:	The customer is now registered, and will be enabled to order goods from the e-shop without providing contact info anew
Related business rules:	[...]
Threats:	**T1:** The customer is not registering with his own name and address, but with an assumed identity. Possible outcomes: T1–1. A non-existing person is registered as customer T1–2. An existing person is unwillingly and unknowingly registered as a customer T1–3. It is revealed to a third party that the named person is a customer of the e-shop (see exception path **E2** above)**T2:** [...]
Author:	John Davis
Date:	2001.05.23

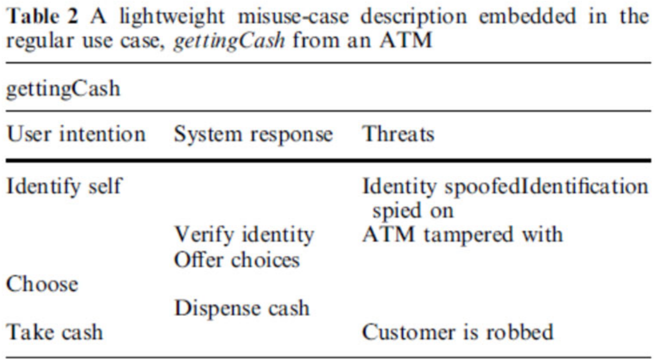


Table 2 A lightweight misuse-case description embedded in the regular use case, *gettingCash* from an ATM

gettingCash

User intention	System response	Threats
Identify self		Identity spoofedIdentification spied on
	Verify identity	ATM tampered with
	Offer choices	
Choose		
	Dispense cash	
Take cash		Customer is robbed

Misuse case description

- Extensive misuse case description
  - For each misuse case, describe an extensive misuse case description
  - Basic path, alternative paths, mitigation points, extension points, misuser profile, and so on
  - Applied to complex misuse cases

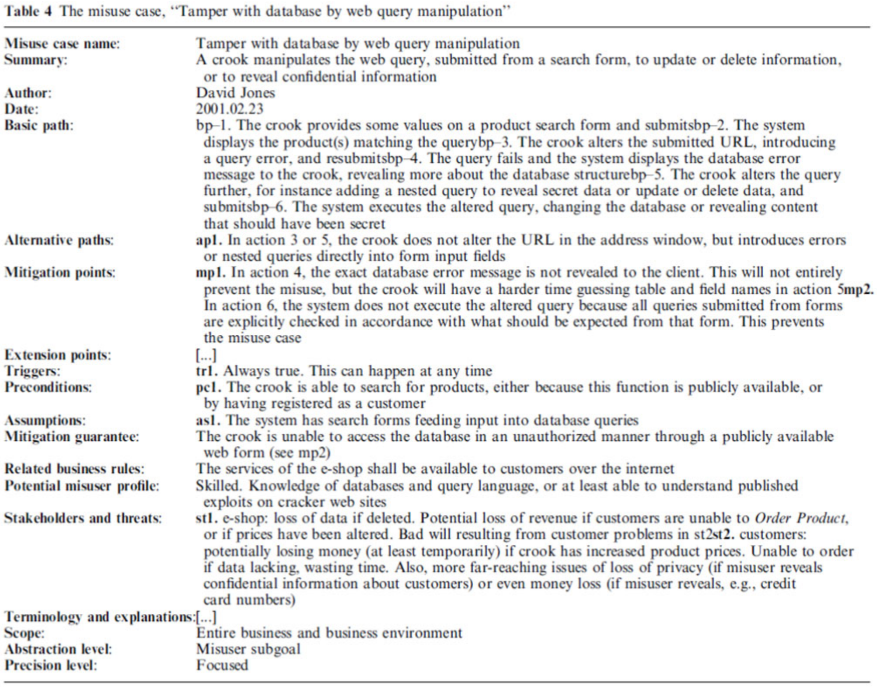
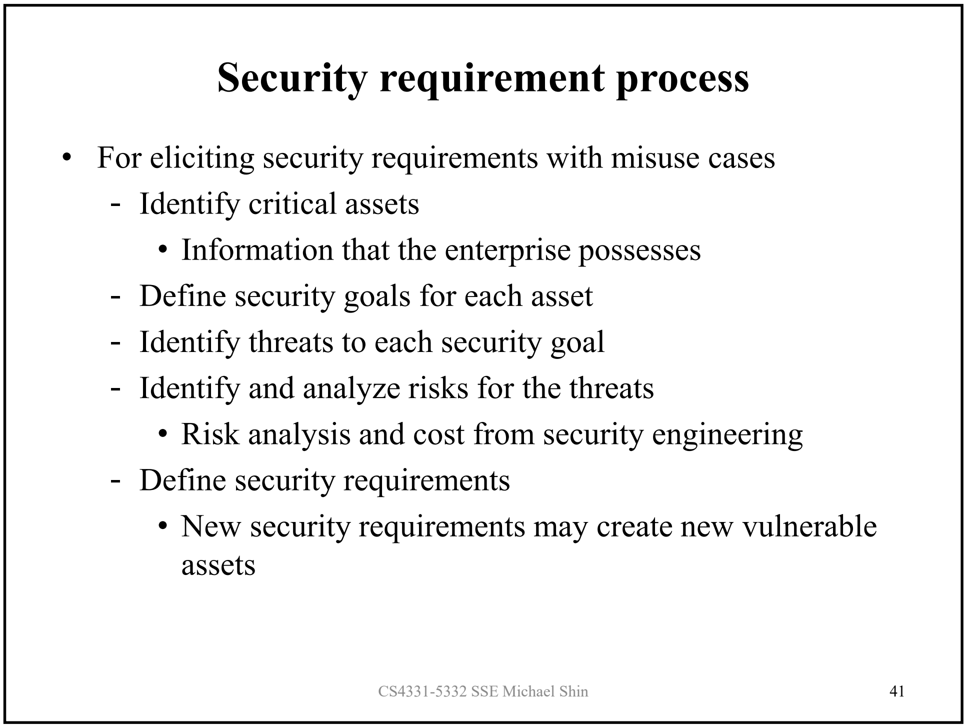


Table 4 The misuse case, "Tamper with database by web query manipulation"

Misuse case name:	Tamper with database by web query manipulation
Summary:	A crook manipulates the web query, submitted from a search form, to update or delete information, or to reveal confidential information
Author:	David Jones
Date:	2001.02.23
Basic path:	bp–1. The crook provides some values on a product search form and submitsbp–2. The system displays the product(s) matching the querybp–3. The crook alters the submitted URL, introducing a query error, and resubmitsbp–4. The query fails and the system displays the database error message to the crook, revealing more about the database structurebp–5. The crook alters the query further, for instance adding a nested query to reveal secret data or update or delete data, and submitsbp–6. The system executes the altered query, changing the database or revealing content that should have been secret
Alternative paths:	**ap1.** In action 3 or 5, the crook does not alter the URL in the address window, but introduces errors or nested queries directly into form input fields
Mitigation points:	**mp1.** In action 4, the exact database error message is not revealed to the client. This will not entirely prevent the misuse, but the crook will have a harder time guessing table and field names in action 5**mp2.** In action 6, the system does not execute the altered query because all queries submitted from forms are explicitly checked in accordance with what should be expected from that form. This prevents the misuse case
Extension points:	[...]
Triggers:	**tr1.** Always true. This can happen at any time
Preconditions:	**pc1.** The crook is able to search for products, either because this function is publicly available, or by having registered as a customer
Assumptions:	**as1.** The system has search forms feeding input into database queries
Mitigation guarantee:	The crook is unable to access the database in an unauthorized manner through a publicly available web form (see mp2)
Related business rules:	The services of the e-shop shall be available to customers over the internet
Potential misuser profile:	Skilled. Knowledge of databases and query language, or at least able to understand published exploits on cracker web sites
Stakeholders and threats:	**st1.** e-shop: loss of data if deleted. Potential loss of revenue if customers are unable to *Order Product*, or if prices have been altered. Bad will resulting from customer problems in st2**st2.** customers: potentially losing money (at least temporarily) if crook has increased product prices. Unable to order if data lacking, wasting time. Also, more far-reaching issues of loss of privacy (if misuser reveals confidential information about customers) or even money loss (if misuser reveals, e.g., credit card numbers)
Terminology and explanations:	[...]
Scope:	Entire business and business environment
Abstraction level:	Misuser subgoal
Precision level:	Focused

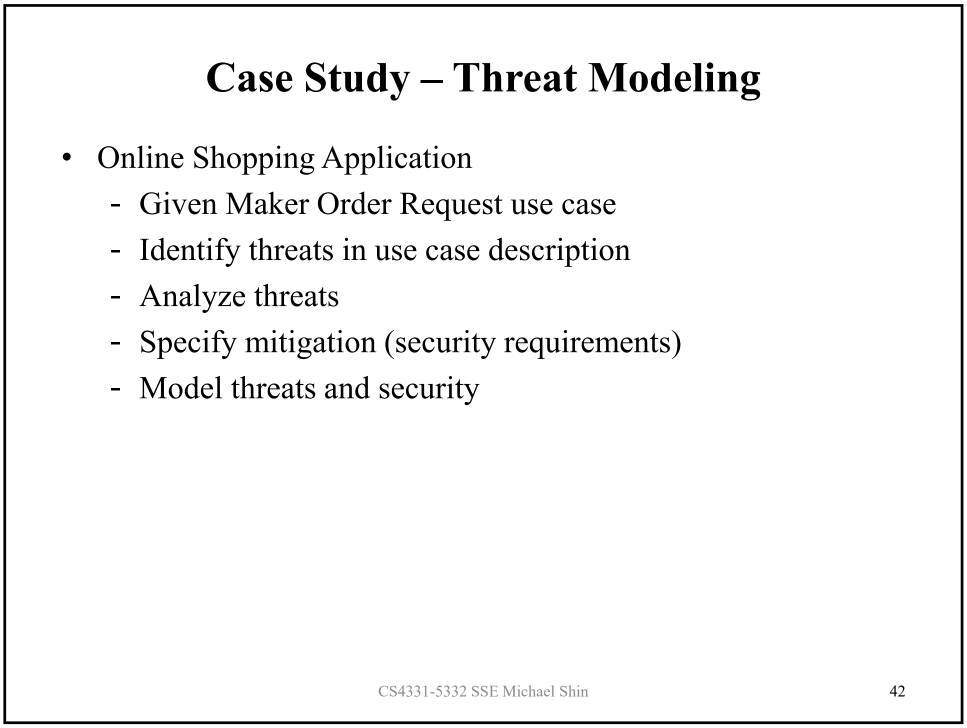
Misuse cases

- Weaknesses
  - Large number of threats may lead to analysis paralysis
    - Reuse of security threats
      - May alleviate the problems
    - But when to stop the security analysis must be developed further
  - There is not always an identifiable misuser
  - Misuse does not always exploit an identifiable sequence of actions



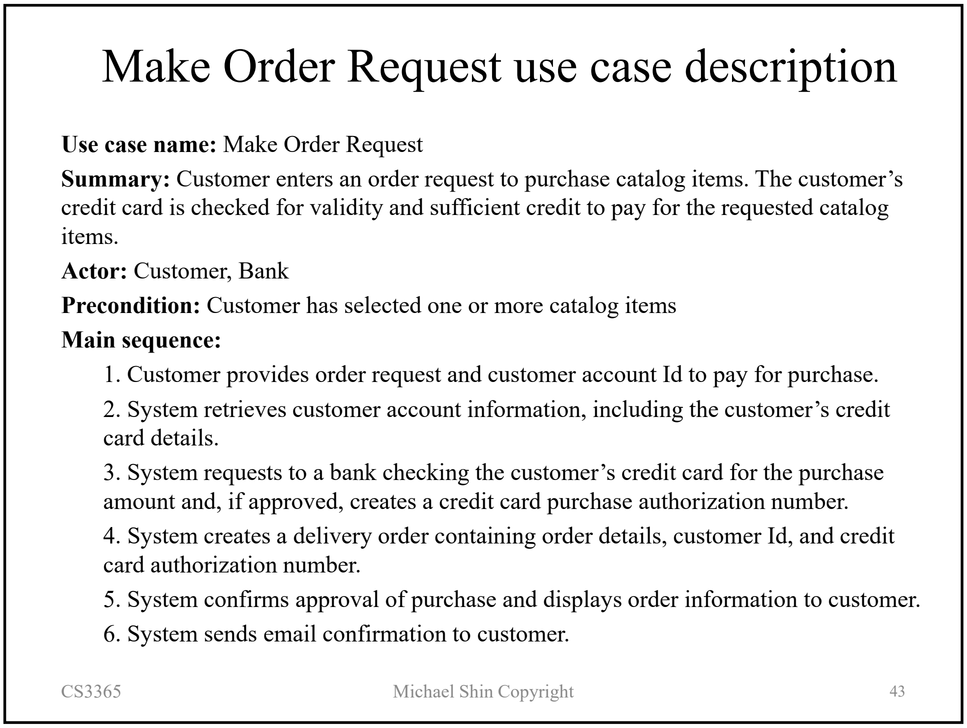
Security requirement process

- For eliciting security requirements with misuse cases
  - Identify critical assets
    - Information that the enterprise possesses
  - Define security goals for each asset
  - Identify threats to each security goal
  - Identify and analyze risks for the threats
    - Risk analysis and cost from security engineering
  - Define security requirements
    - New security requirements may create new vulnerable assets



Case Study – Threat Modeling

- Online Shopping Application
  - Given Maker Order Request use case
  - Identify threats in use case description
  - Analyze threats
  - Specify mitigation (security requirements)
  - Model threats and security



Make Order Request use case description

Use case name: Make Order Request

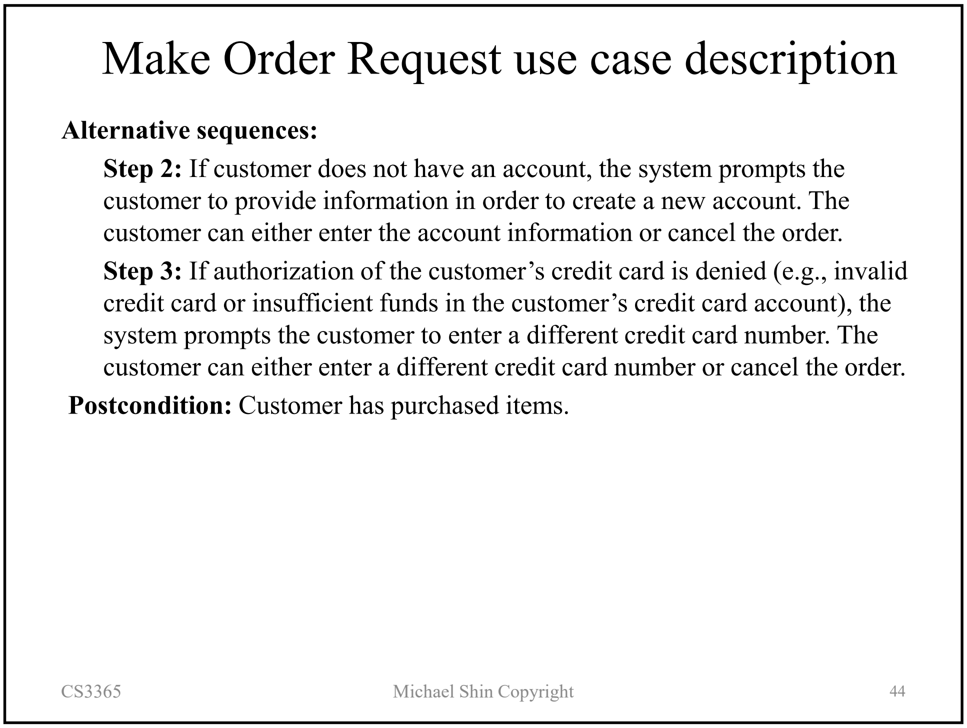
Summary: Customer enters an order request to purchase catalog items. The customer's credit card is checked for validity and sufficient credit to pay for the requested catalog items.

Actor: Customer, Bank

Precondition: Customer has selected one or more catalog items

Main sequence:

1. Customer provides order request and customer account Id to pay for purchase.
2. System retrieves customer account information, including the customer's credit card details.
3. System requests to a bank checking the customer's credit card for the purchase amount and, if approved, creates a credit card purchase authorization number.
4. System creates a delivery order containing order details, customer Id, and credit card authorization number.
5. System confirms approval of purchase and displays order information to customer.
6. System sends email confirmation to customer.



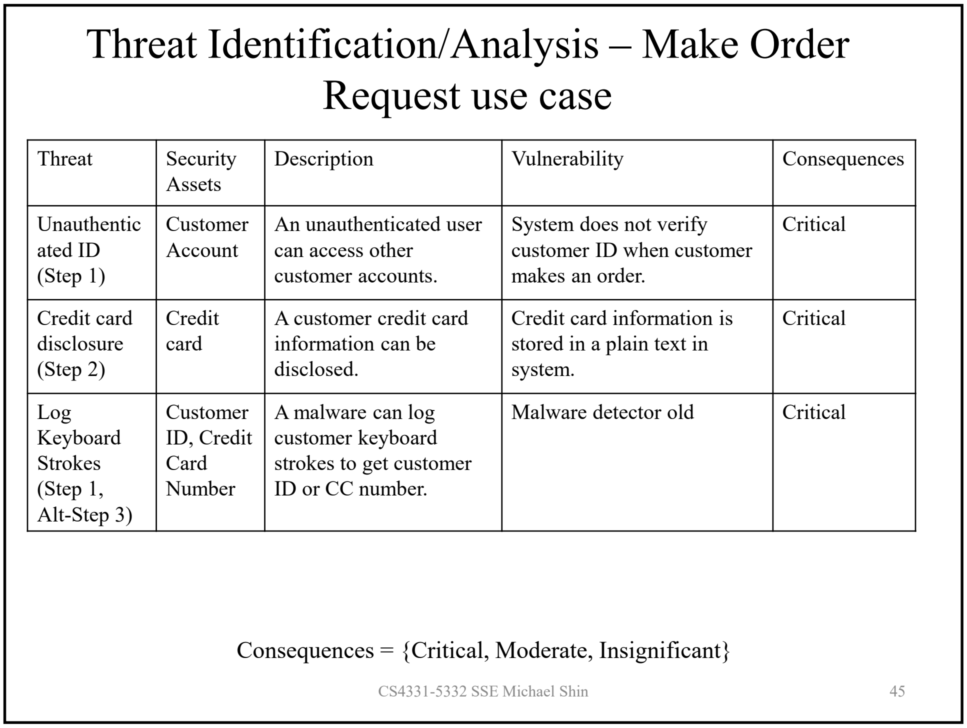
Make Order Request use case description

Alternative sequences:

Step 2: If customer does not have an account, the system prompts the customer to provide information in order to create a new account. The customer can either enter the account information or cancel the order.

Step 3: If authorization of the customer's credit card is denied (e.g., invalid credit card or insufficient funds in the customer's credit card account), the system prompts the customer to enter a different credit card number. The customer can either enter a different credit card number or cancel the order.

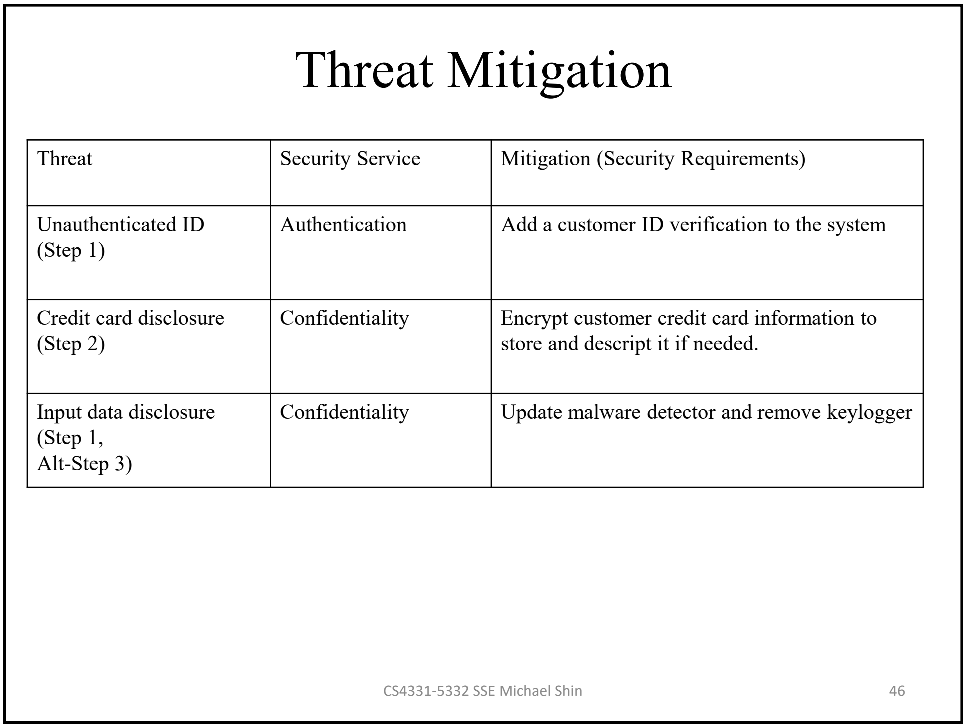
Postcondition: Customer has purchased items.



Threat Identification/Analysis – Make Order Request use case

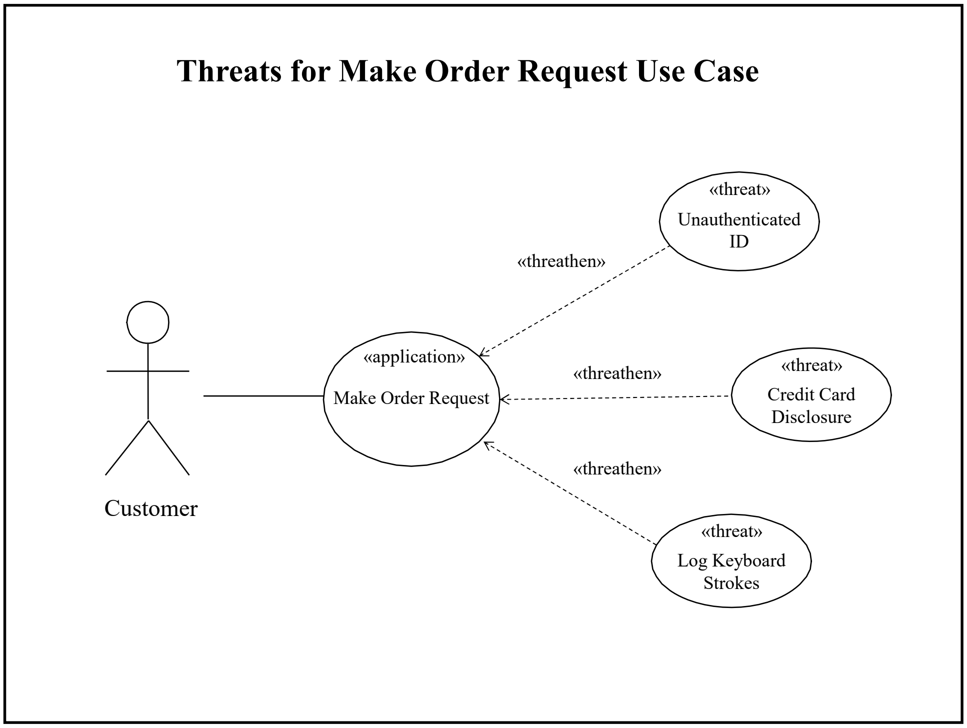
Threat	Security Assets	Description	Vulnerability	Consequences
Unauthenticated ID (Step 1)	Customer Account	An unauthenticated user can access other customer accounts.	System does not verify customer ID when customer makes an order.	Critical
Credit card disclosure (Step 2)	Credit card	A customer credit card information can be disclosed.	Credit card information is stored in a plain text in system.	Critical
Log Keyboard Strokes (Step 1, Alt-Step 3)	Customer ID, Credit Card Number	A malware can log customer keyboard strokes to get customer ID or CC number.	Malware detector old	Critical

Consequences = {Critical, Moderate, Insignificant}



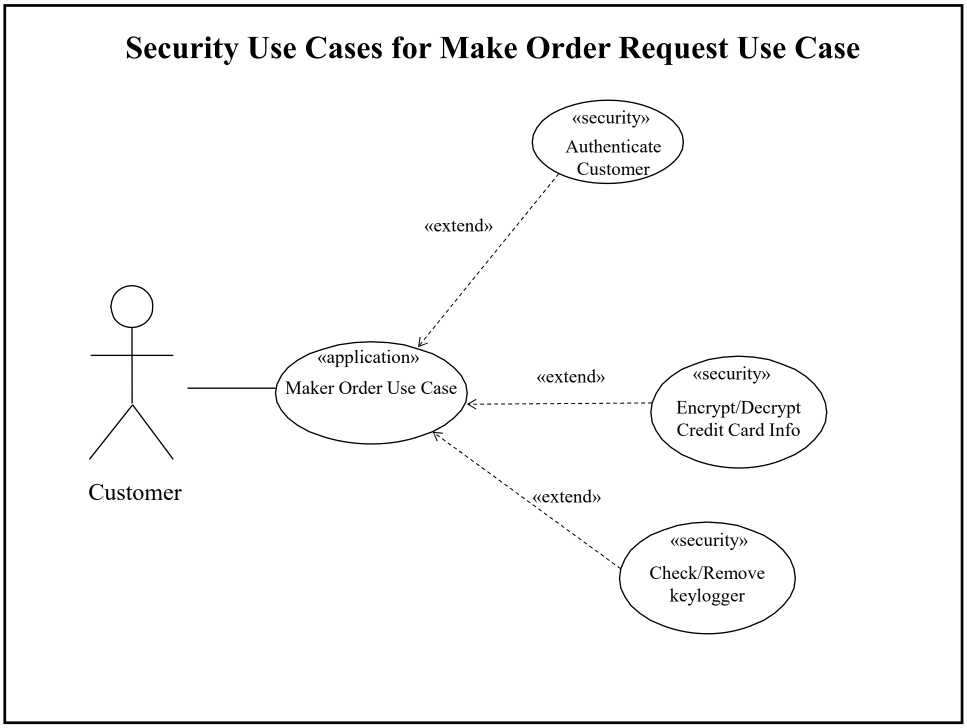
Threat Mitigation

Threat	Security Service	Mitigation (Security Requirements)
Unauthenticated ID (Step 1)	Authentication	Add a customer ID verification to the system
Credit card disclosure (Step 2)	Confidentiality	Encrypt customer credit card information to store and descript it if needed.
Input data disclosure (Step 1, Alt-Step 3)	Confidentiality	Update malware detector and remove keylogger



Threats for Make Order Request Use Case

47



Security Use Cases for Make Order Request Use Case

48