Security Risk Analysis - Threat Modeling

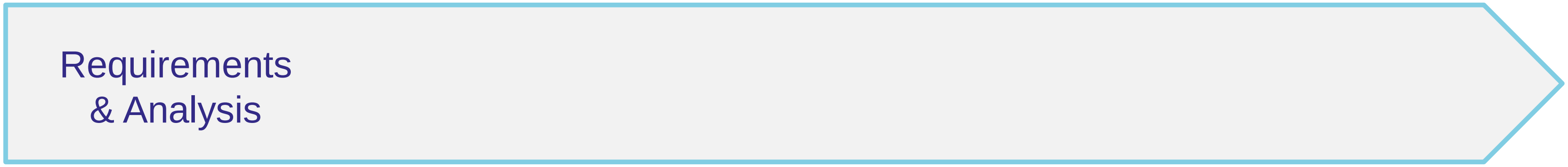
Special Topic: Practical Computer Security

CSCI-GA.3033-019

Challenges in Security

- Problems of security arise most often with the people using the system, rather than the technology.
- Security needs to deal with changing environments and the emergence of new threats.
- Security is often not considered in the original design, or not built in as the system is developed, but is retrofitted in the project.
- Security often costs. Large amounts of time and other resources may be spent in developing secure systems.

Security Activities in SDLC



Requirements
& Analysis

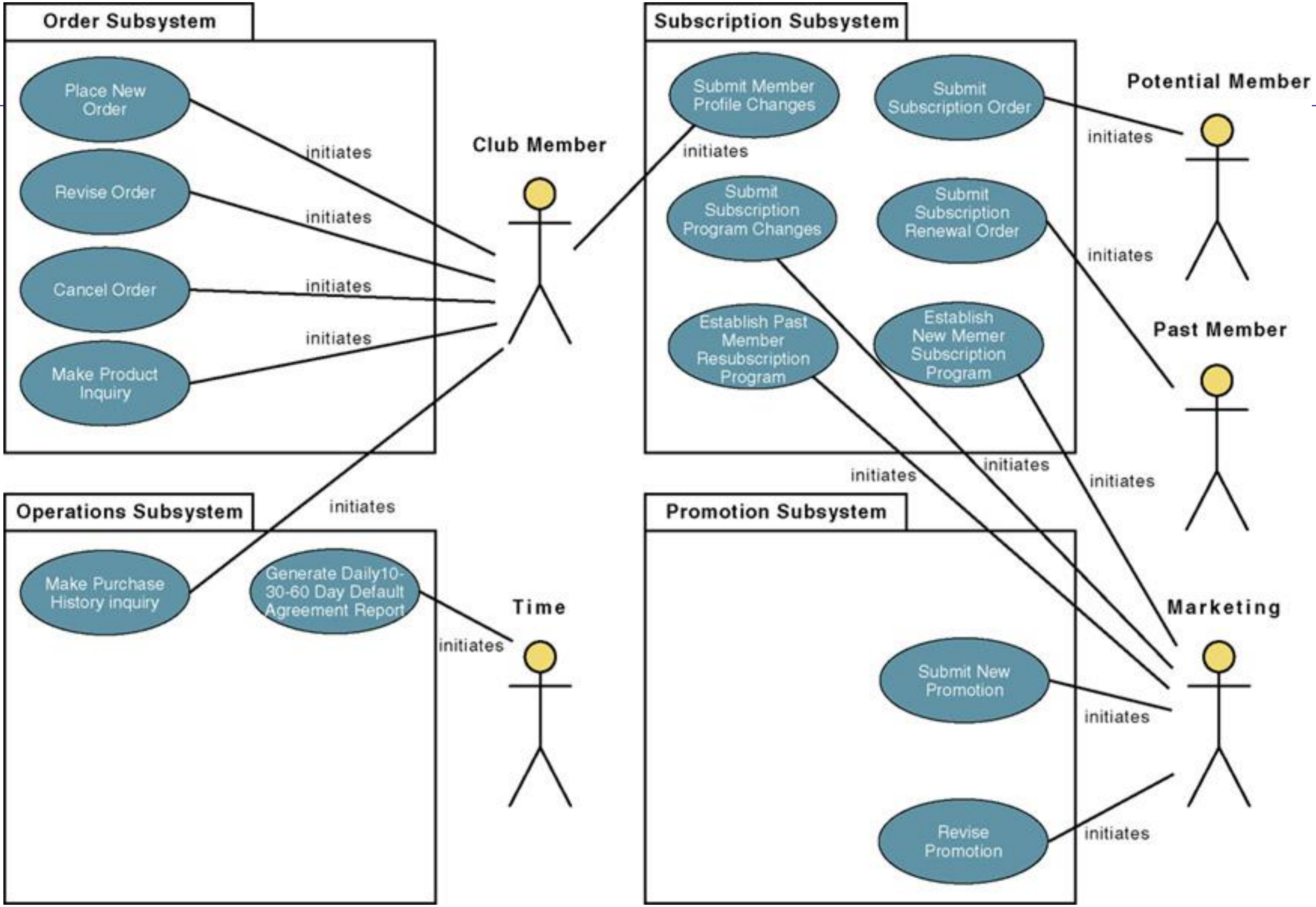
Functional vs. Nonfunctional Requirements

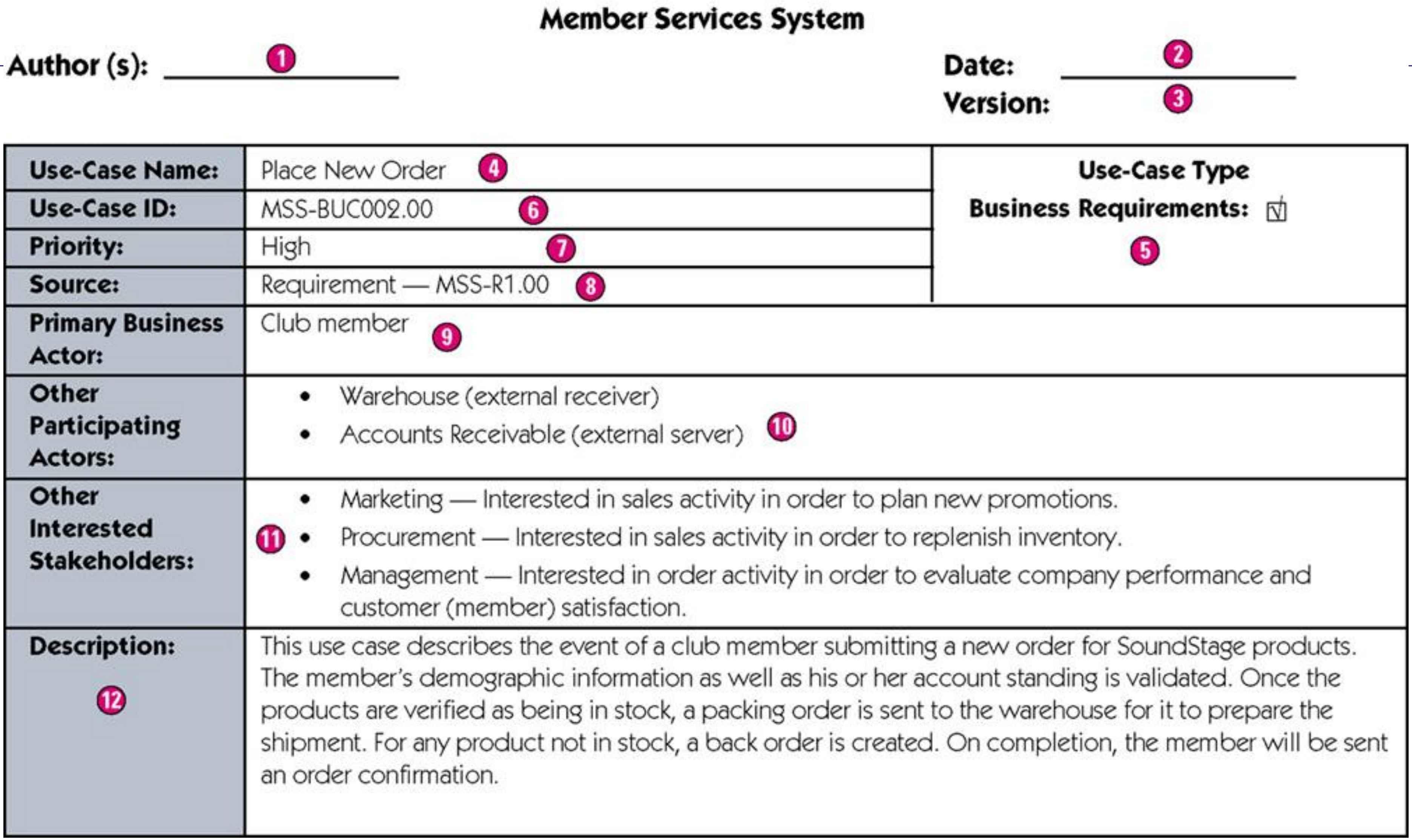
Examples of Functional requirements

- The system shall allow users to view book’s information by categories defined by users.
- The system shall allow users to add new book’s information.

Examples of Nonfunctional requirements

- The user interface for the system shall be suited for screens with resolution 1024x768.
- The system shall be available 99.99% of the time for any 24-hour period.
- The system shall not disclose any personal information about customers apart from their name and reference.


Order Subsystem
Place New Order
Revise Order
Cancel Order
Make Product Inquiry
initiates
initiates
initiates
initiates
Club Member
Subscription Subsystem
Submit Member Profile Changes
Submit Subscription Order
initiates
Submit Subscription Program Changes
Submit Subscription Renewal Order
Establish Past Member Resubscription Program
Establish New Memer Subscription Program
Potential Member
initiates
initiates
Past Member
initiates
initiates
initiates
Operations Subsystem
Make Purchase History inquiry
Generate Daily10-30-60 Day Default Agreement Report
initiates
Time
initiates
Promotion Subsystem
Marketing
Submit New Promotion
initiates
Revise Promotion
initiates

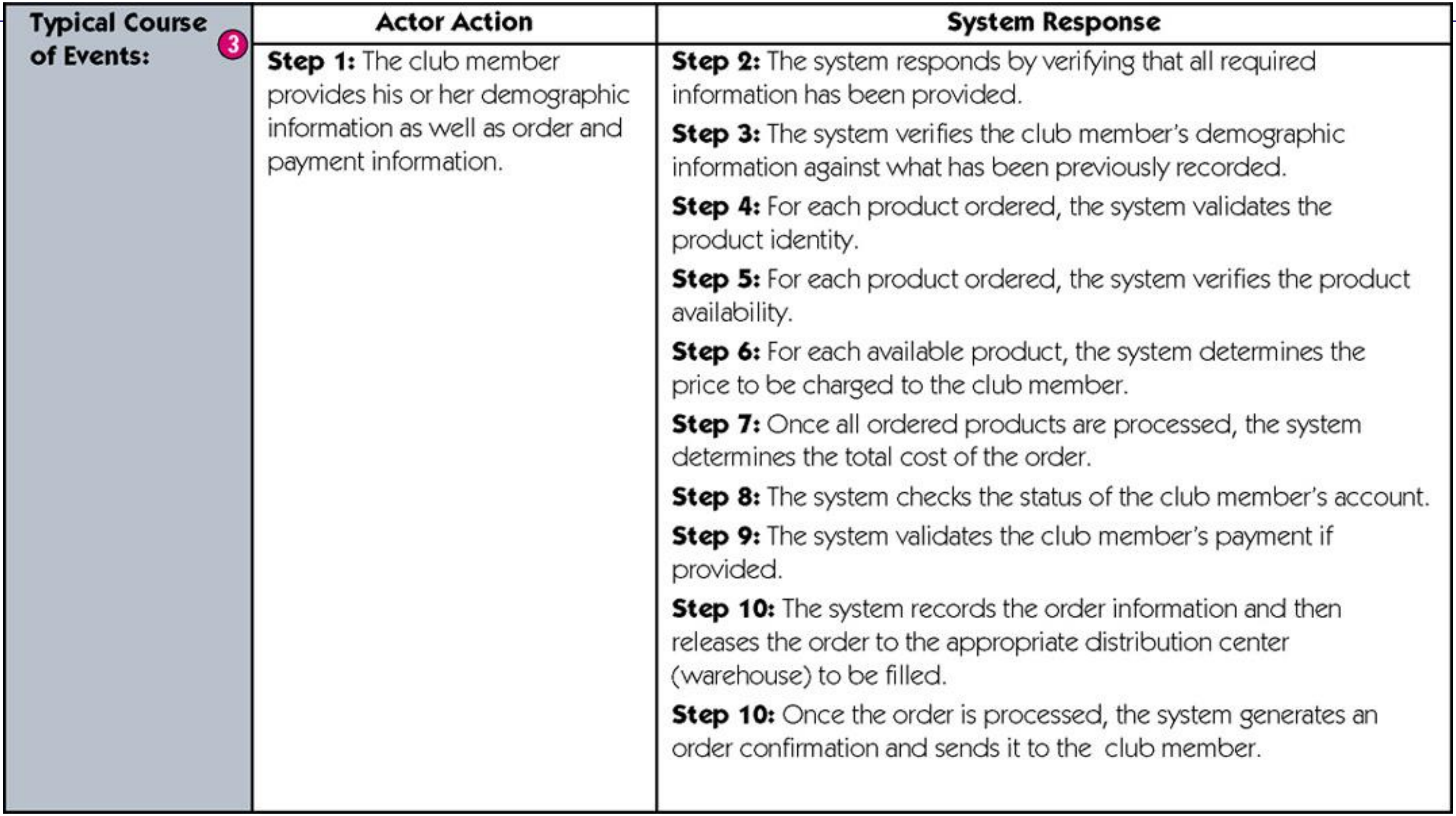


Member Services System

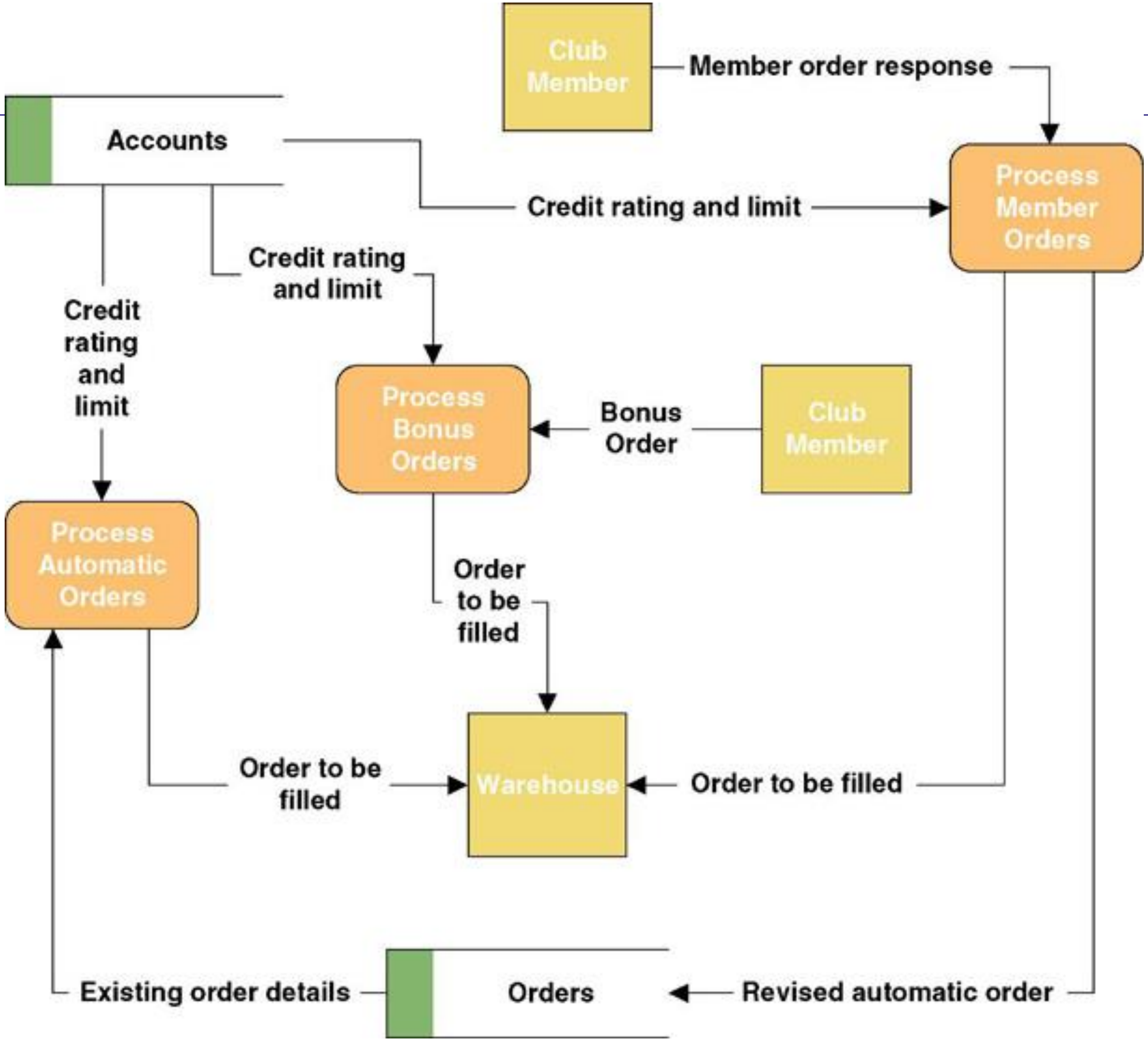
Author (s): ____❶____ Date: ____❷____

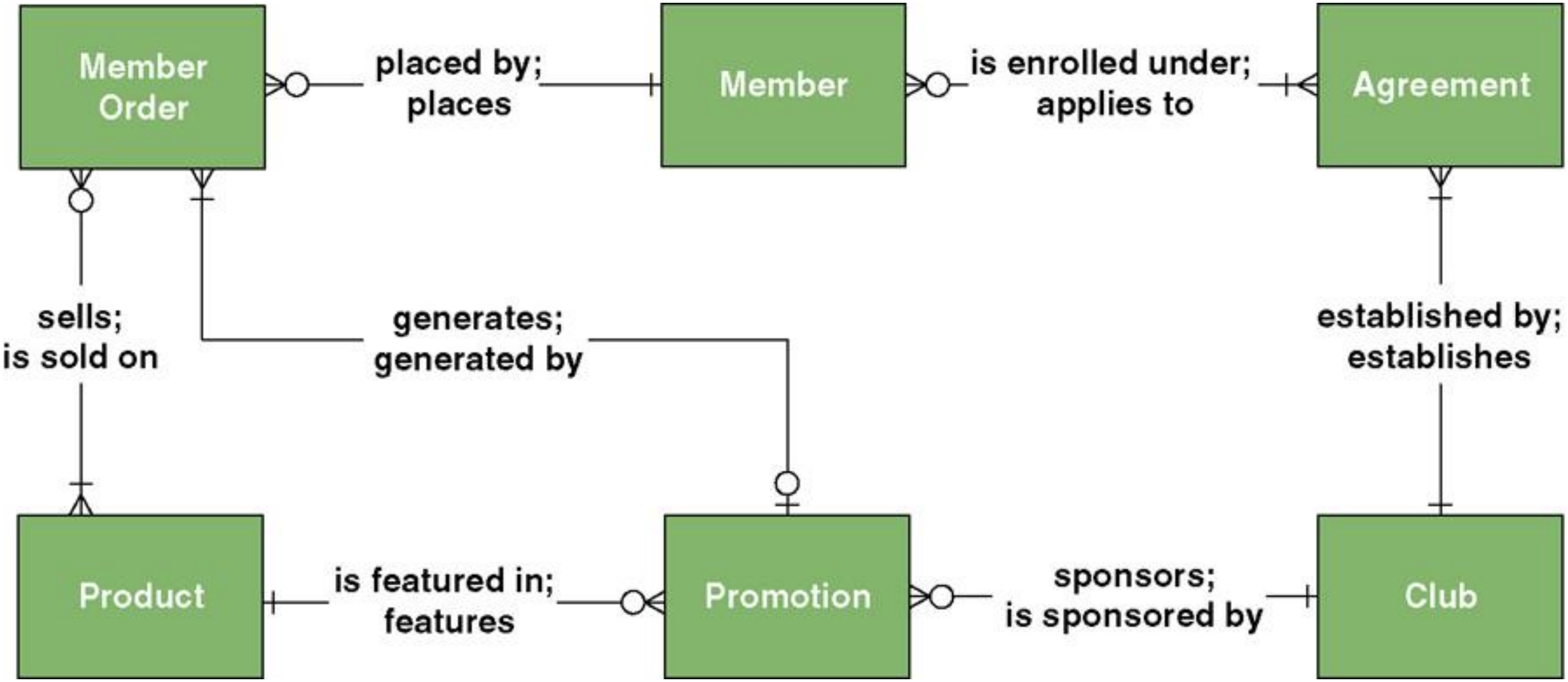
Version: ❸

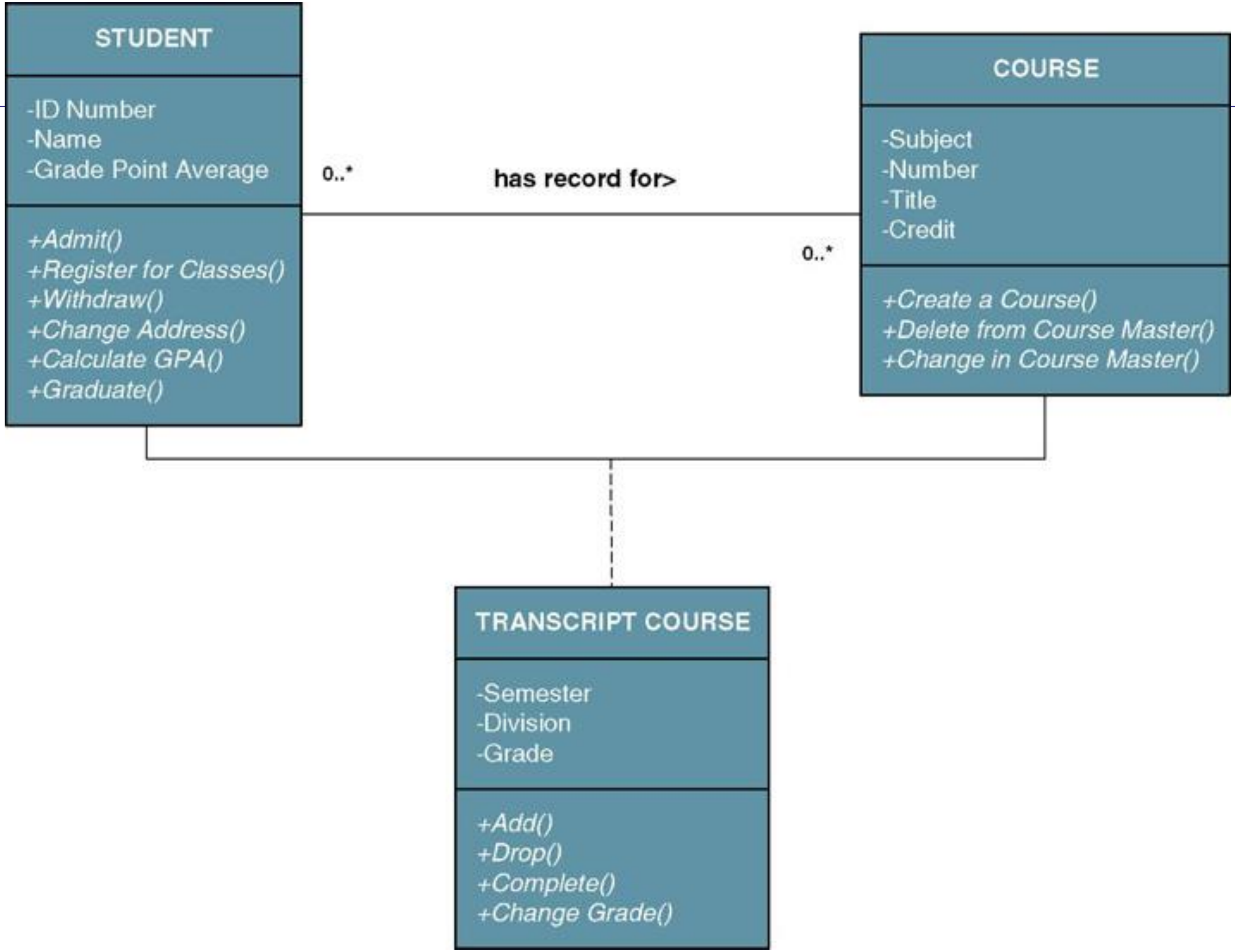
Use-Case Name:	Place New Order ❹	**Use-Case Type** **Business Requirements:** ☑ ❺
Use-Case ID:	MSS-BUC002.00 ❻	
Priority:	High ❼	
Source:	Requirement — MSS-R1.00 ❽	
Primary Business Actor:	Club member ❾	
Other Participating Actors:	• Warehouse (external receiver) • Accounts Receivable (external server) ❿	
Other Interested Stakeholders:	⓫ • Marketing — Interested in sales activity in order to plan new promotions. • Procurement — Interested in sales activity in order to replenish inventory. • Management — Interested in order activity in order to evaluate company performance and customer (member) satisfaction.	
Description: ⓬	This use case describes the event of a club member submitting a new order for SoundStage products. The member's demographic information as well as his or her account standing is validated. Once the products are verified as being in stock, a packing order is sent to the warehouse for it to prepare the shipment. For any product not in stock, a back order is created. On completion, the member will be sent an order confirmation.	



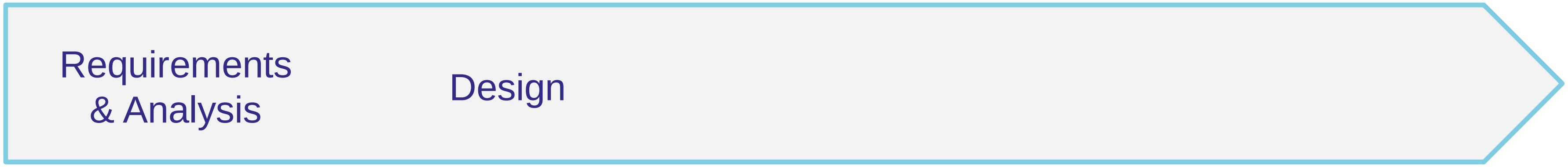
Typical Course of Events: 3	**Actor Action**	**System Response**
	Step 1: The club member provides his or her demographic information as well as order and payment information.	**Step 2:** The system responds by verifying that all required information has been provided. **Step 3:** The system verifies the club member's demographic information against what has been previously recorded. **Step 4:** For each product ordered, the system validates the product identity. **Step 5:** For each product ordered, the system verifies the product availability. **Step 6:** For each available product, the system determines the price to be charged to the club member. **Step 7:** Once all ordered products are processed, the system determines the total cost of the order. **Step 8:** The system checks the status of the club member's account. **Step 9:** The system validates the club member's payment if provided. **Step 10:** The system records the order information and then releases the order to the appropriate distribution center (warehouse) to be filled. **Step 10:** Once the order is processed, the system generates an order confirmation and sends it to the club member.


Club Member
Member order response
Accounts
Credit rating and limit
Process Member Orders
Credit rating and limit
Credit rating and limit
Process Bonus Orders
Bonus Order
Club Member
Process Automatic Orders
Order to be filled
Order to be filled
Warehouse
Order to be filled
Existing order details
Orders
Revised automatic order


Member Order
placed by; places
Member
is enrolled under; applies to
Agreement
sells; is sold on
generates; generated by
established by; establishes
Product
is featured in; features
Promotion
sponsors; is sponsored by
Club

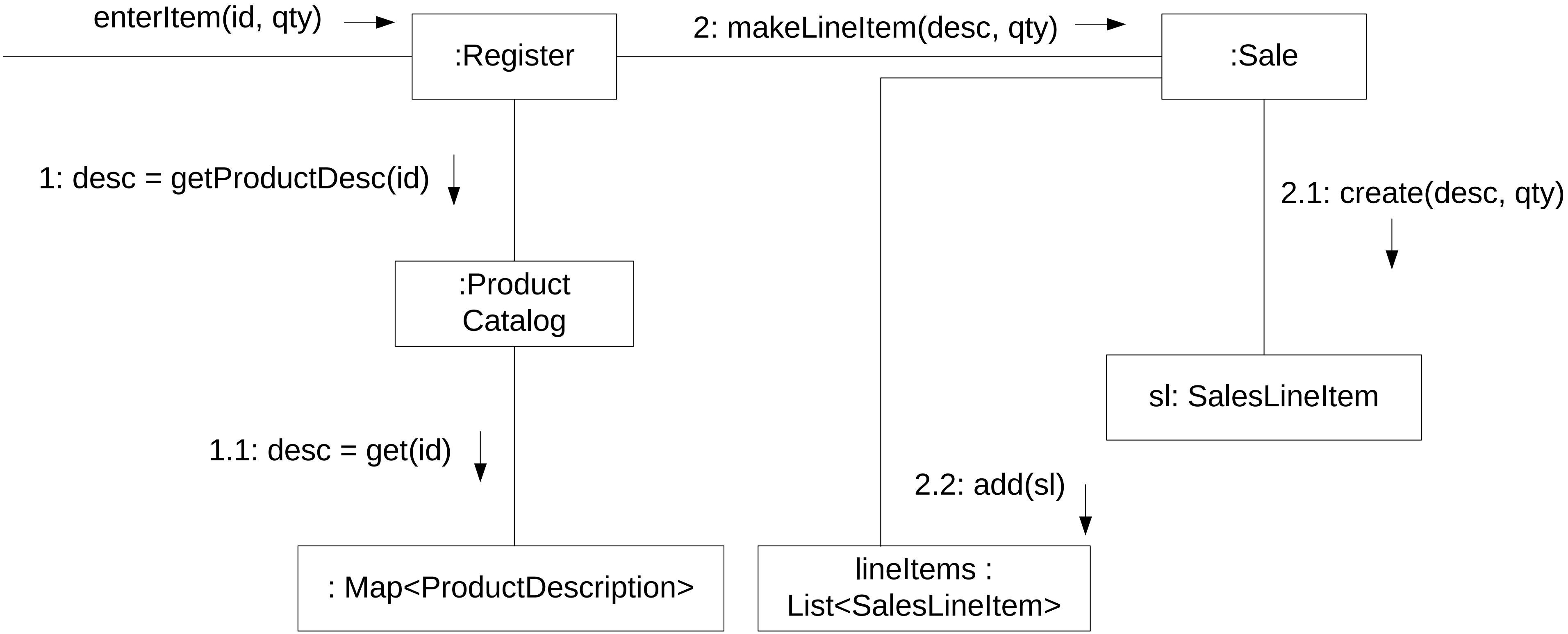

STUDENT
-ID Number
-Name
-Grade Point Average
+Admit()
+Register for Classes()
+Withdraw()
+Change Address()
+Calculate GPA()
+Graduate()
0..*
has record for>
0..*
COURSE
-Subject
-Number
-Title
-Credit
+Create a Course()
+Delete from Course Master()
+Change in Course Master()
TRANSCRIPT COURSE
-Semester
-Division
-Grade
+Add()
+Drop()
+Complete()
+Change Grade()

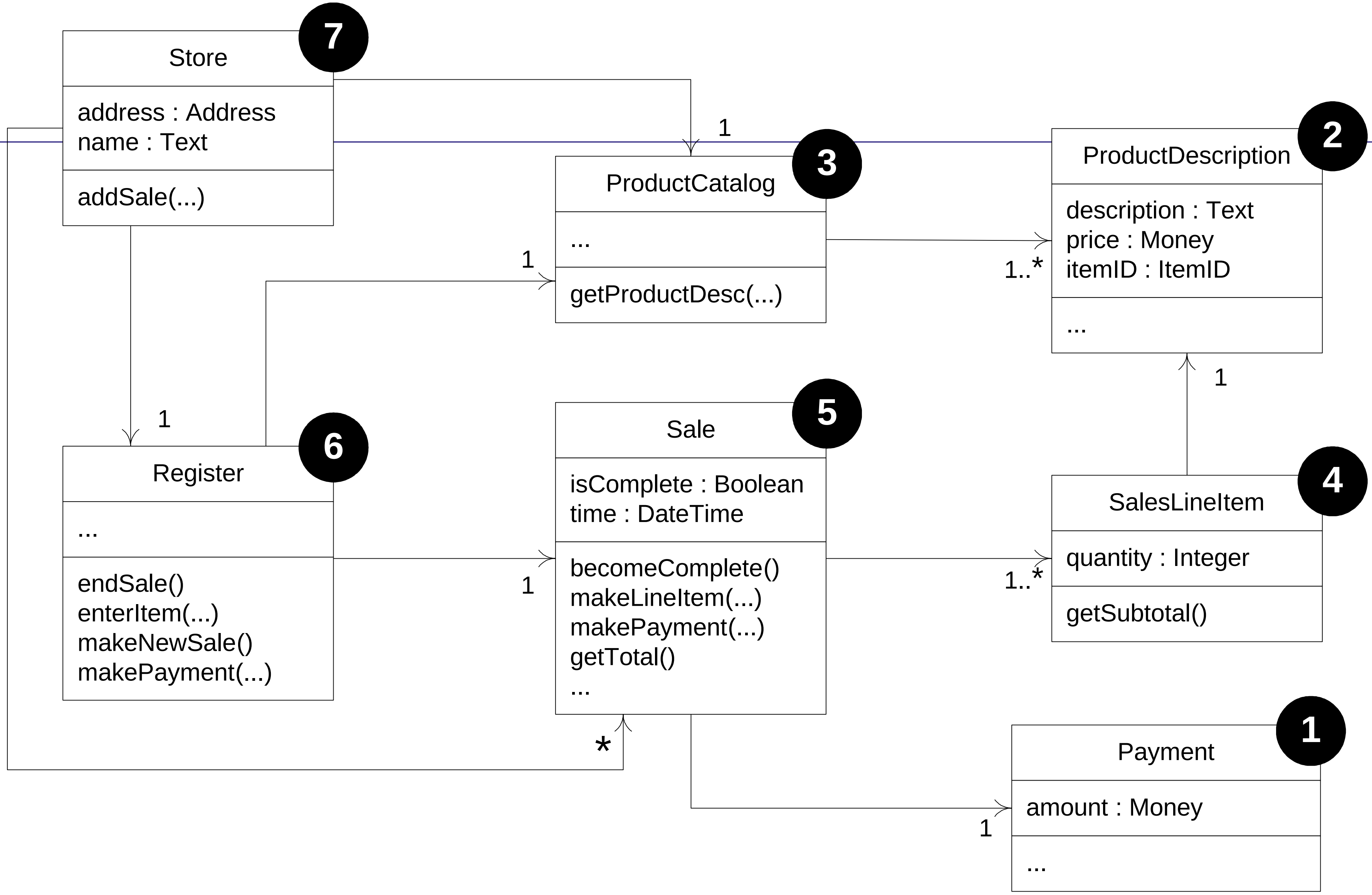
Security Activities in SDLC



Requirements & Analysis

Design


enterItem(id, qty)
:Register
2: makeLineItem(desc, qty)
:Sale
1: desc = getProductDesc(id)
2.1: create(desc, qty)
:Product Catalog
sl: SalesLineItem
1.1: desc = get(id)
2.2: add(sl)
: Map<ProductDescription>
lineItems : List<SalesLineItem>

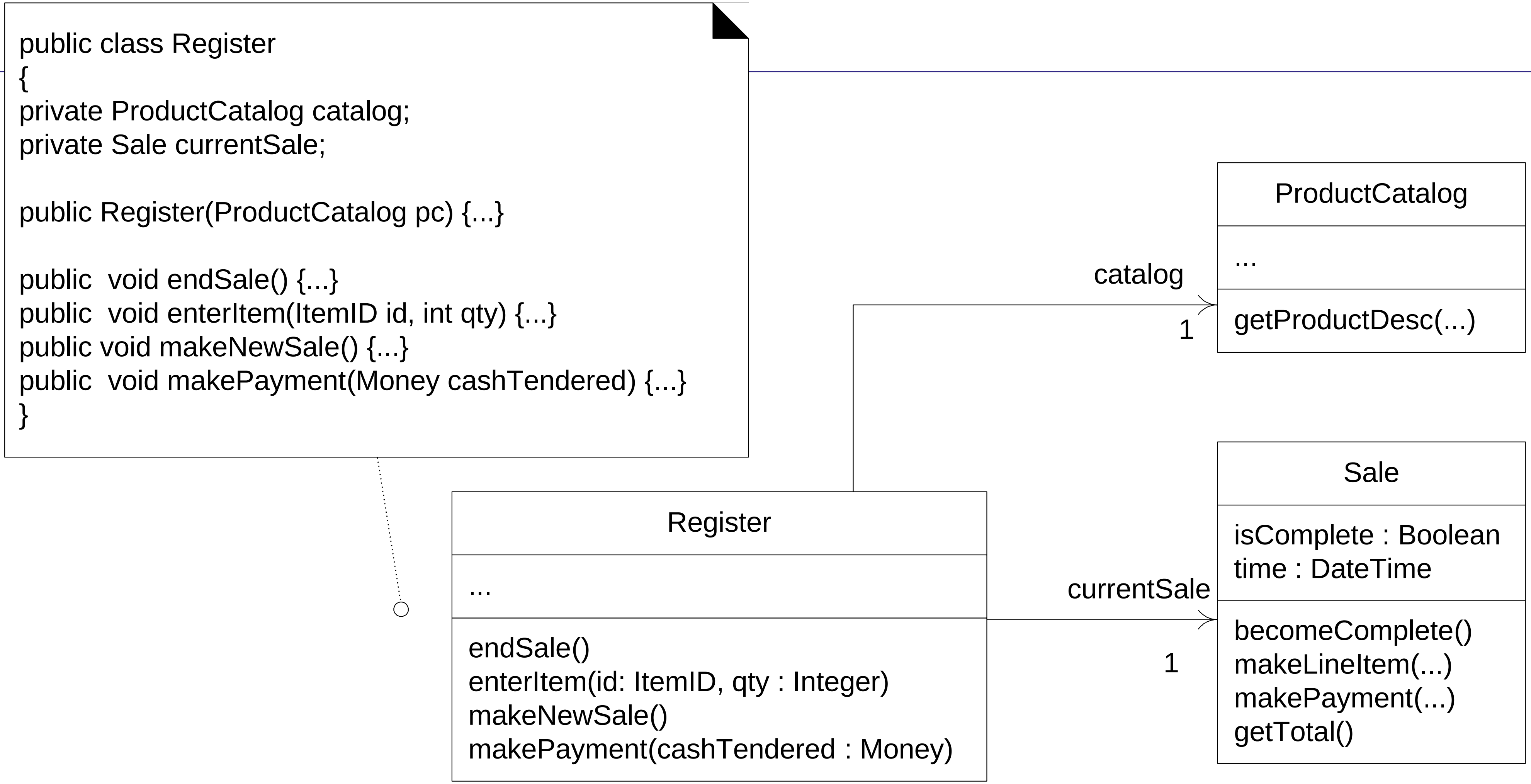

7
Store
address : Address
name : Text
addSale(...)
1
3
ProductCatalog
...
getProductDesc(...)
1
2
ProductDescription
description : Text
price : Money
itemID : ItemID
...
1..*
1
1
6
Register
...
endSale()
enterItem(...)
makeNewSale()
makePayment(...)
1
5
Sale
isComplete : Boolean
time : DateTime
becomeComplete()
makeLineItem(...)
makePayment(...)
getTotal()
...
*
1..*
4
SalesLineItem
quantity : Integer
getSubtotal()
1
1
Payment
amount : Money
...

Security Activities in SDLC

Requirements & Analysis

Design

Implementation

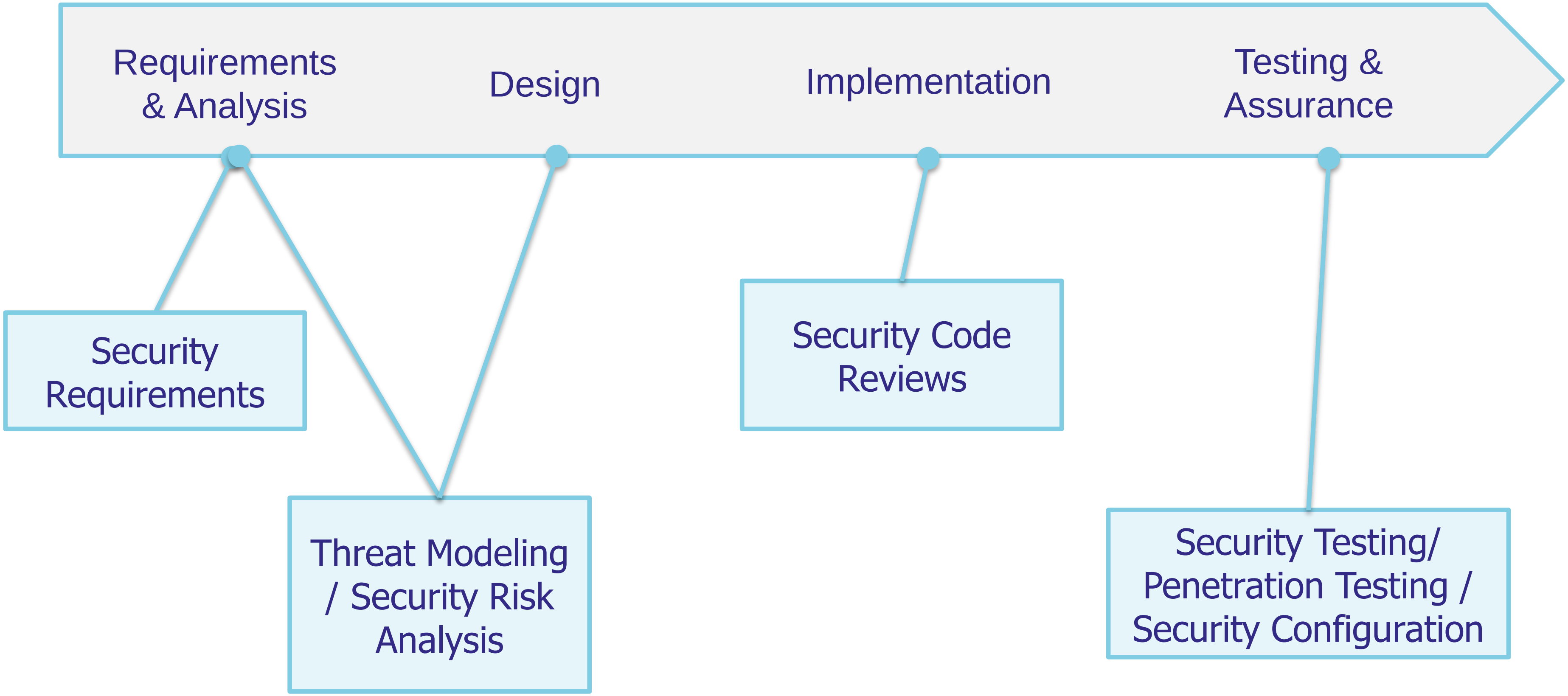


```
public class Register
{
private ProductCatalog catalog;
private Sale currentSale;

public Register(ProductCatalog pc) {...}

public  void endSale() {...}
public  void enterItem(ItemID id, int qty) {...}
public void makeNewSale() {...}
public  void makePayment(Money cashTendered) {...}
}
```

Security Activities in SDLC



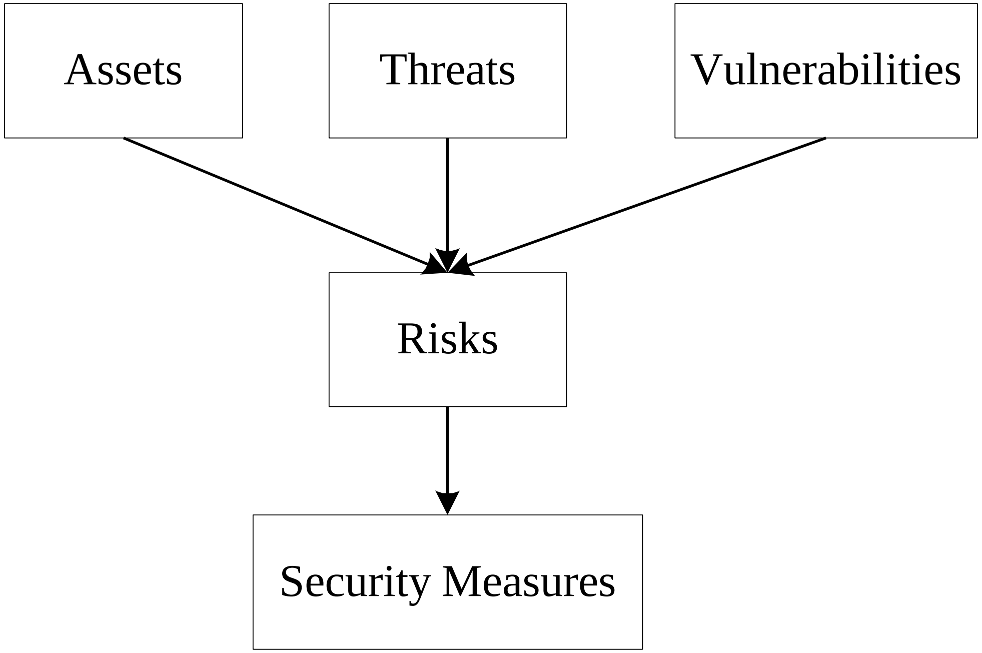
Security Risk Analysis & Management

- Risk Management
  - Is a method of identifying vulnerabilities, threats and assessing the possible impacts
- Risk Analysis
  - Ensure that security is cost-effective and relevant to threats

Risk = Threat + Vulnerability + Impact

Threat ……… *others* control
Vulnerability ……*we* control

Risk Analysis Process Framework



- **Assets** - cost, importance and impacts
- **Threats**- likelihood, severity and impacts
- **Vulnerabilities -** likelihood

Security risk is the evaluation of the combination of likelihood of threat, likelihood of vulnerability and impact for a given state of a system. Risks need to be assessed to determine the level of security required for the assets that need to be secured.

Value of assets

- If a server costs $4,000, should this be input as the value of the asset in the risk assessment?
    - What about cost of replacing or repairing it, the loss of productivity, and the value of any data that may be corrupted?

(Harris, 2016)

Cost that make up the value

- Cost to acquire/develop
- Cost to maintain / to replace
- Value of asset to owners and users / adversaries
- Price others are willing to pay for the asset
- Operational/production activities affected
- Liability issues if the asset is compromised
- Usefulness and role of the asset in the organization

(Harris, 2016)

Risk Analysis Approaches

- Quantitative Risk Analysis
  - Monetary/numeric values are assigned to asset value, threat frequency, severity of vulnerability, impact damage, safeguard costs, uncertainty
  - The total and residual risks are determined with equations.
- Qualitative Risk Analysis
  - Opinion and scenario-based and uses a rating system to relay the risk criticality levels

Quantitative Risk Analysis

Single Loss Expectancy (SLE)

SLE = Asset Value x Exposure Factor (EF)

where EF represents the percentage of loss a realized threat could have on an asset (e.g. if a fire where to occur, 25 percent of asset would be damaged)

Annual Loss Expectancy (ALE)

ALE = SLE x Annualized Rate of Occurrence (ARO)

where ARO represents the estimated frequency of a specific threat taking place within a 12-month timeframe.

(Harris, 2016)

Quantitative Risk Analysis : Example

Asset	Threat	SLE	ARO	ALE
Facility	Fire	$230,000	0.1	$23,000
Trade Secret	Stolen	$40,000	0.01	$400
File server	Failed	$11,500	0.1	$1,150
Data	Virus	$6,500	1.0	$6,500
Customer credit card info	Stolen	$300,000	3.0	$900,000

(Harris, 2016)

Qualitative Risk Analysis : Example

Threat = Unauthorized access to confidential information

	Severity of Threat	Probability of Threat	Potential Loss	Effectiveness of Firewall	Effectiveness of IDS	Effectiveness of Honeypot
IT manager	4	2	4	4	3	2
DB admin	4	4	4	3	4	1
Application programmer	2	3	3	4	2	1
System operator	3	4	3	4	2	1
Operational operator	5	4	4	4	4	2
Results	3.6	3.4	3.6	3.8	3	1.4

(Harris, 2016)

Residual Risk

- Countermeasures are introduced to reduce its overall risk to an acceptable level.
- No system is 100 percent secure -> residual risk – risk left over for us to deal with.

Total risk = threats x vulnerability x asset value

(when no controls is implemented)

Residual risk = (threats x vulnerability x asset value) x controls gap

Residual risk = total risk - countermeasures

*These formulas are used to illustrate the relation of the different items that make up risk in a conceptual manner.

Handling Risk

- **Risk reduction/mitigation**—Implement a countermeasure
- **Risk transference**—Purchase insurance to transfer a portion or all of the potential cost of a loss to a third party.
- **Risk acceptance**—Do nothing. Deal with risk by accepting the potential cost and loss if the risk occurs.
- **Risk avoidance**—Discontinue activity.

What is Threat Modeling?

- “Have you threat modeled?”
  - Analysis process to figure out the significant threats (what might go wrong?) to the system

- “What is your threat model?”
  - Examples: Our threat model is someone stealing our sensitive information.

What is Threat Modeling?

- “The threat model was completely wrong.” - Why Cryptosystems Fail? - Ross Anderson
- Threat modeling
  - is about using models to find security problems.
  - use of abstractions to help in thinking about risks.
  - enables you to find issues in things you haven’t built yet
  - is a process to understand security threat to a system, determine risks from those threats, and establish appropriate mitigations.
- Threat modeling is the key to a *focused* defense.

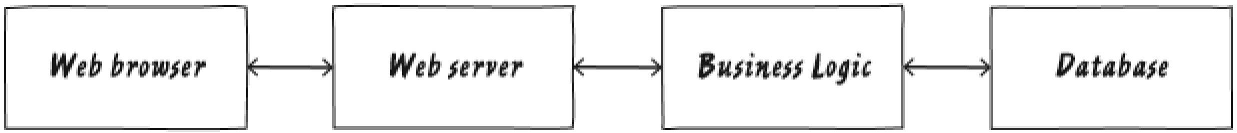
Threat Modeling Process

4-step framework

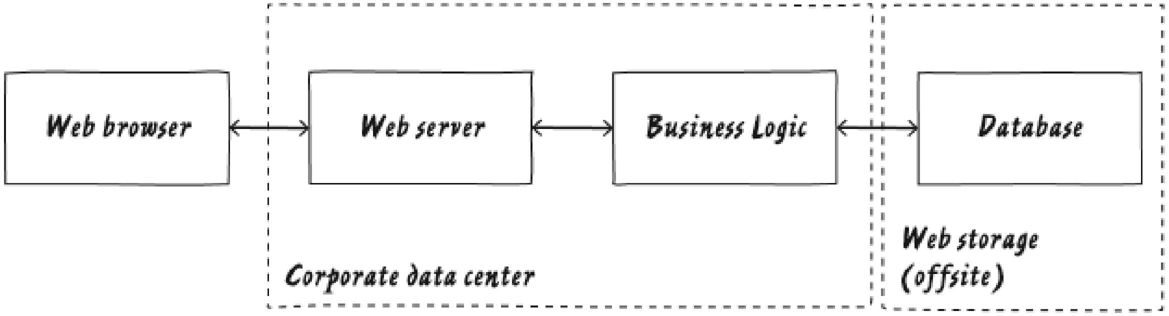
- What are you building?
  - Characterizing your system
- What can go wrong?
  - Finding threats
- What should you do about those things that can go wrong?
  - Addressing each threat
- Did you do a decent job of analysis?
  - Check your work.

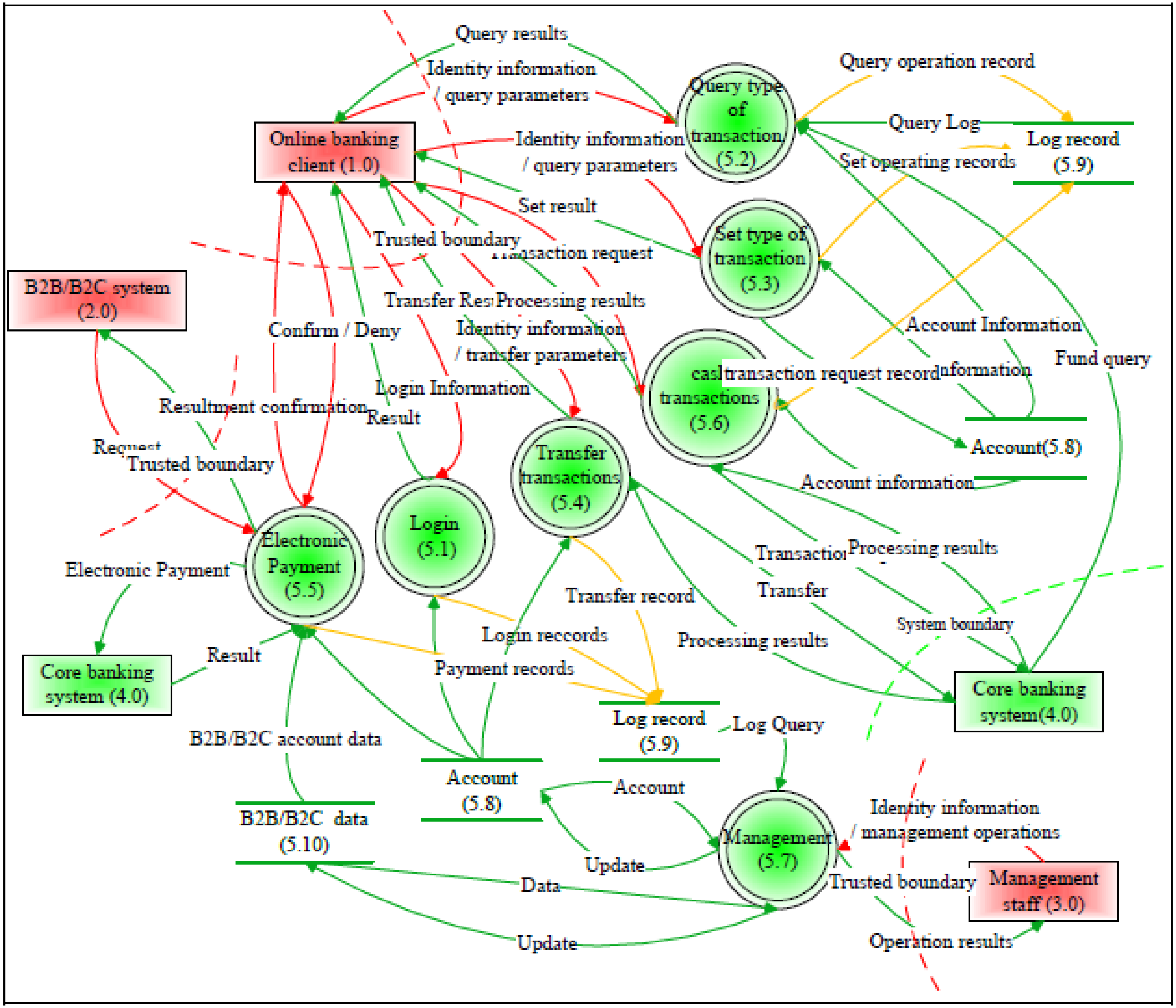
What are you building?

- Diagrams are a good way to communicate what you are building.
  - Refer to http://www.sersc.org/journals/IJSIA/vol8_no2_2014/28.pdf for examples.

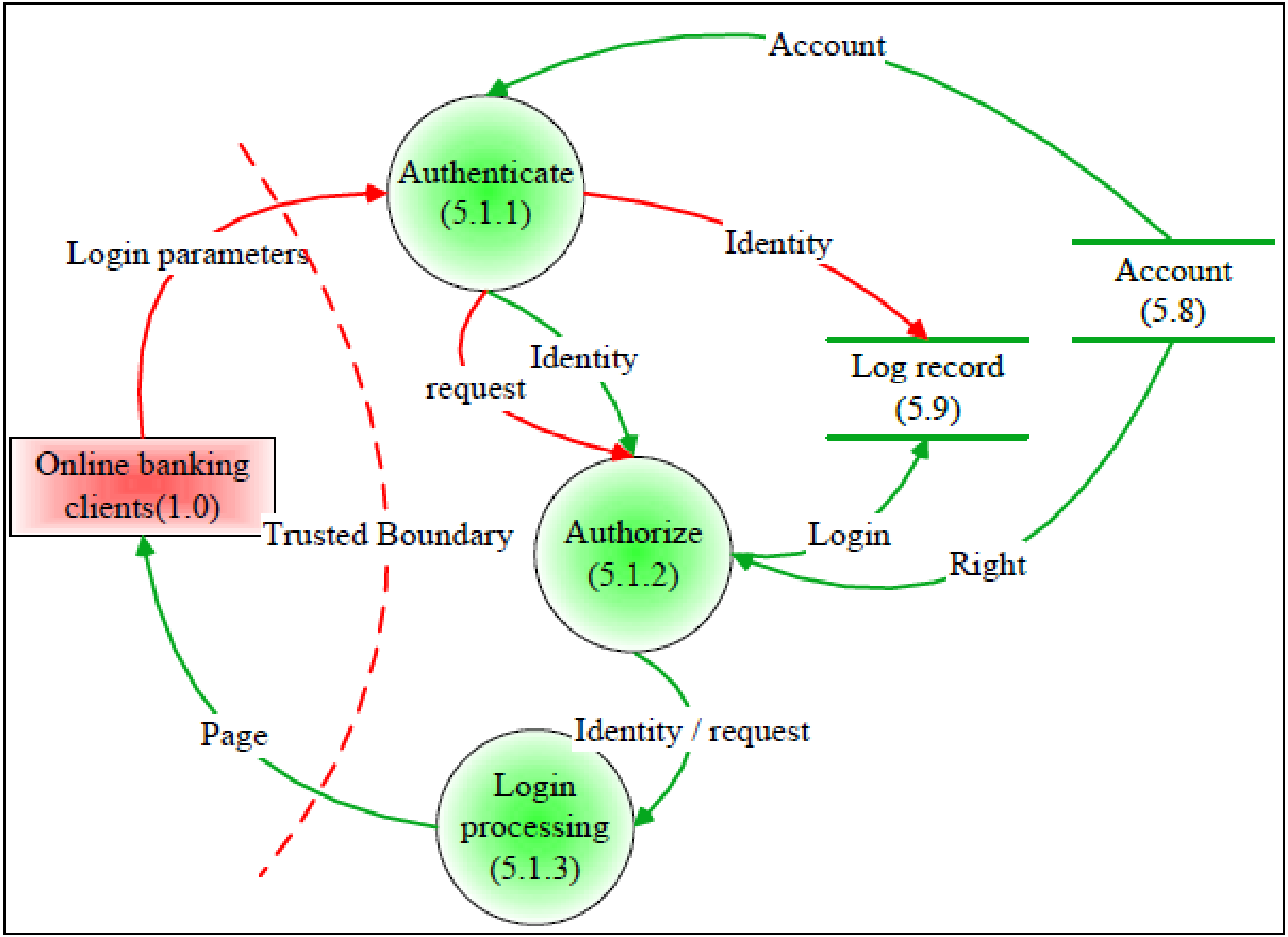


- Trust boundaries. - threats that cross boundaries are likely important ones (“who controls what”)
  - Draw trust boundaries when different people control different things.

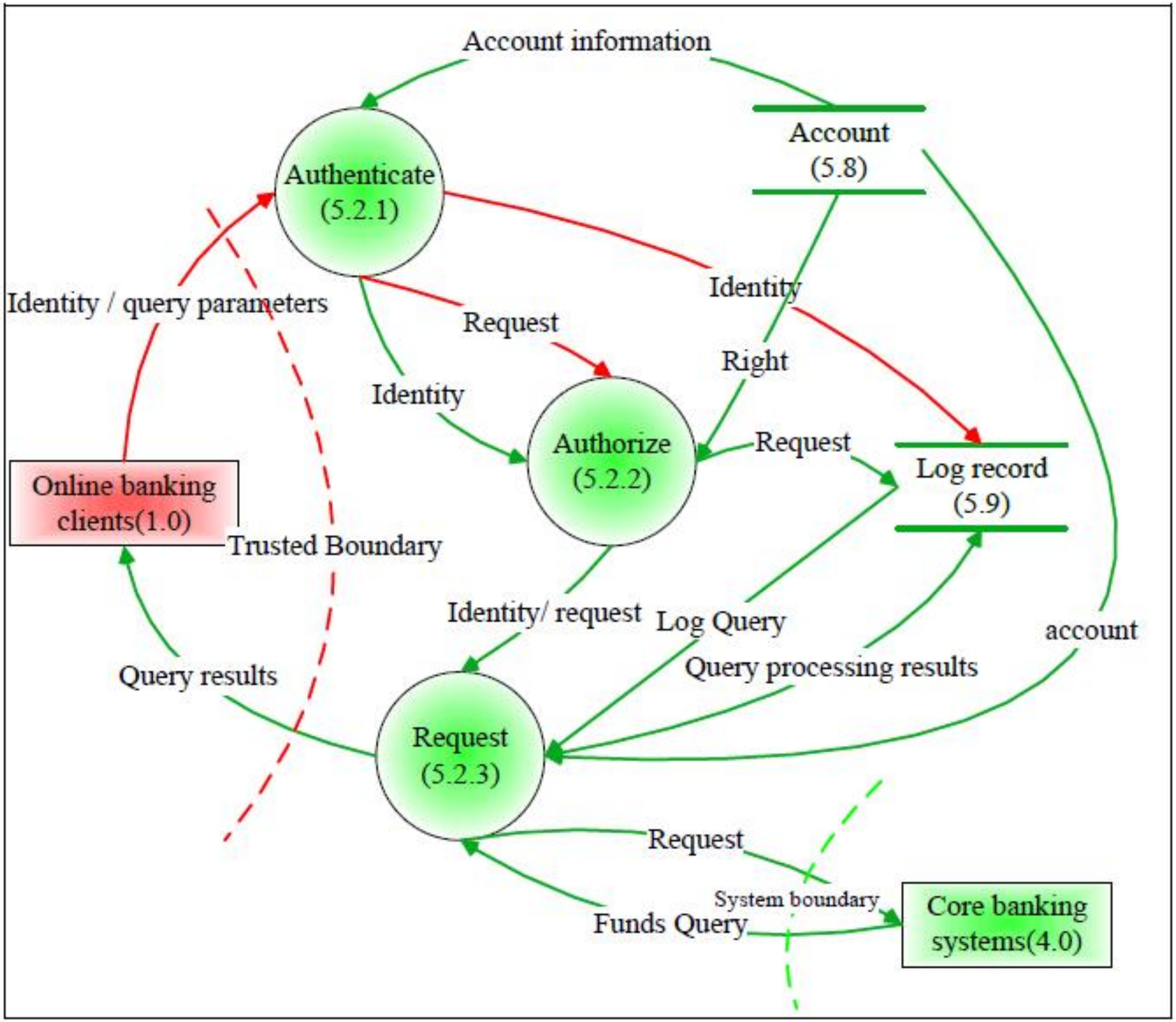




Ref: http://www.sersc.org/journals/IJSIA/vol8_no2_2014/28.pdf



Ref: http://www.sersc.org/journals/IJSIA/vol8_no2_2014/28.pdf



Ref: http://www.sersc.org/journals/IJSIA/vol8_no2_2014/28.pdf

What can go wrong? - Identifying Threats

- Once you have a diagram of your system, you can start looking for what can go wrong with its security.
- “A threat is the adversary’s goal, or what an adversary might try to do to a system” – Swiderski & Snyder, 2004
- ‘Think like an adversary!’
  - How to identify possible threats?
  - Any problem?
    - Not systematic and unstructured.
    - Likely to leave possible attacks uninvestigated.

Trap: ‘Think like an attacker’ - ‘Think like a professional chef’

- Adam Shostack

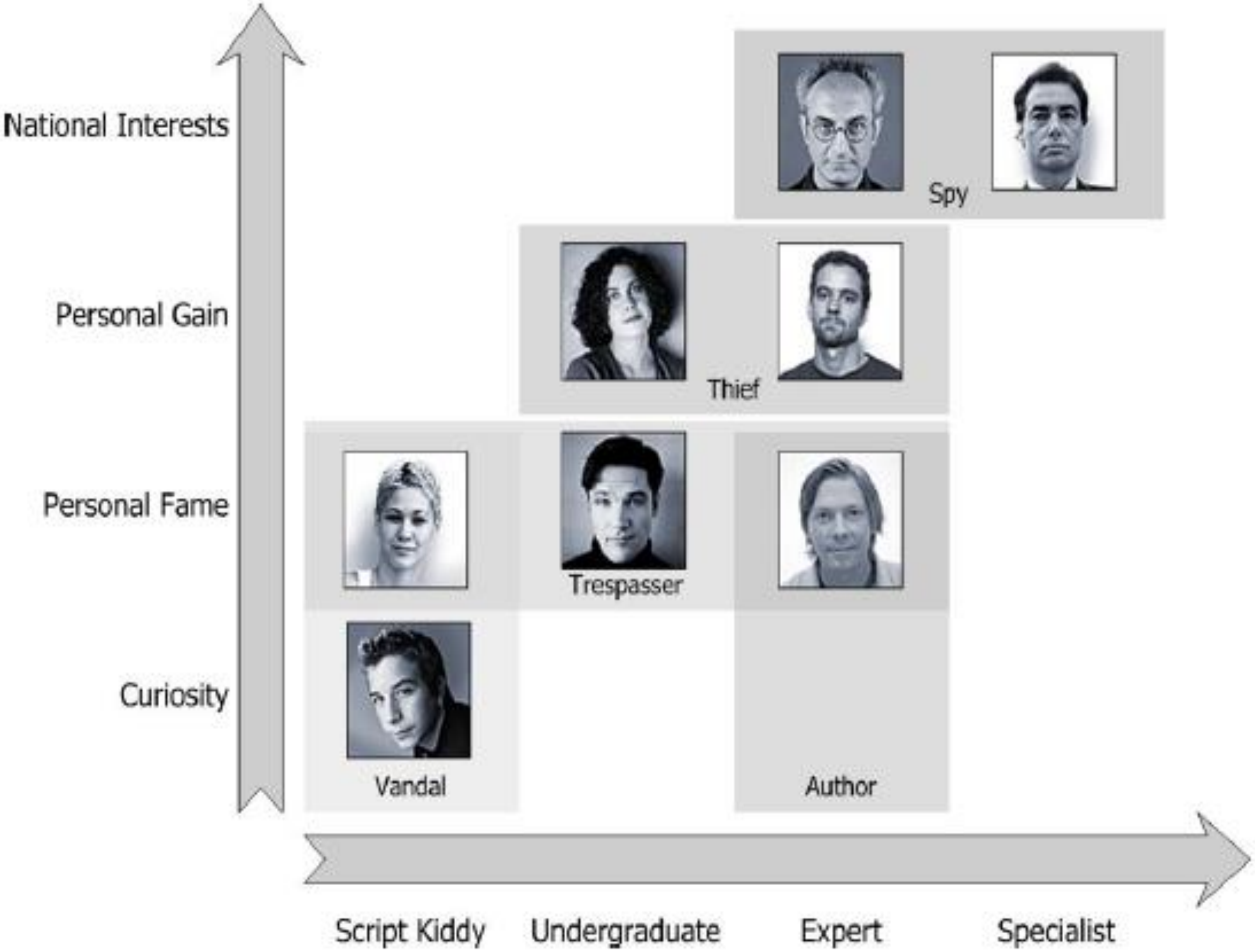
Brainstorming your threats

- Quality depends on
  - Experience of the brainstormers
  - Time spent

- Perspective on brainstorming
  - Unstructured discussion
  - When to stop (exit criteria)

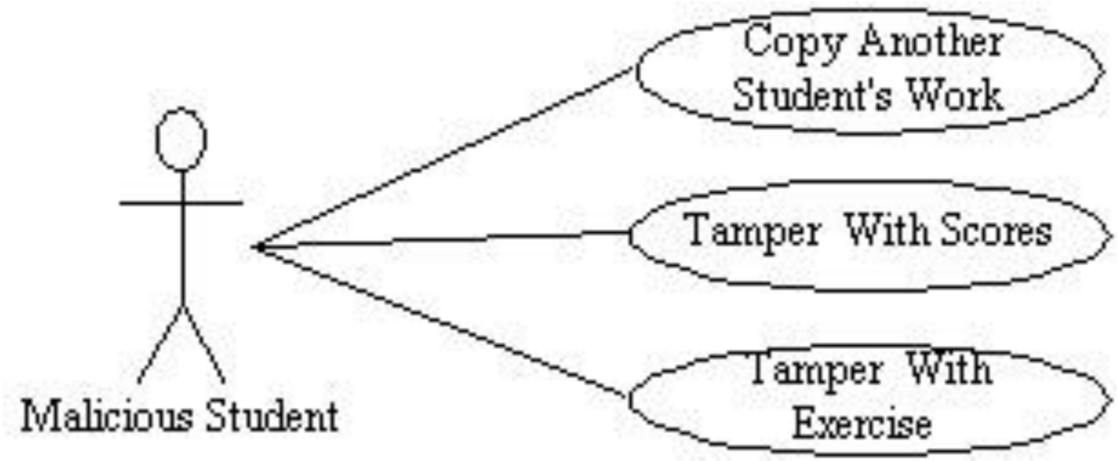
Threat Modeling Approaches

- Focusing on assets
  - Evaluates from asset identification
- Focusing on attacks
  - Evaluates from the point of view of an attacker
- Focusing on software
  - Evaluates based on the software being built or a system being deployed


National Interests
Personal Gain
Personal Fame
Curiosity
Spy
Thief
Trespasser
Vandal
Author
Script Kiddy
Undergraduate
Expert
Specialist

Abuse Case [McDermott 1999]

- Aims
  - Means to capture and analyse security requirements



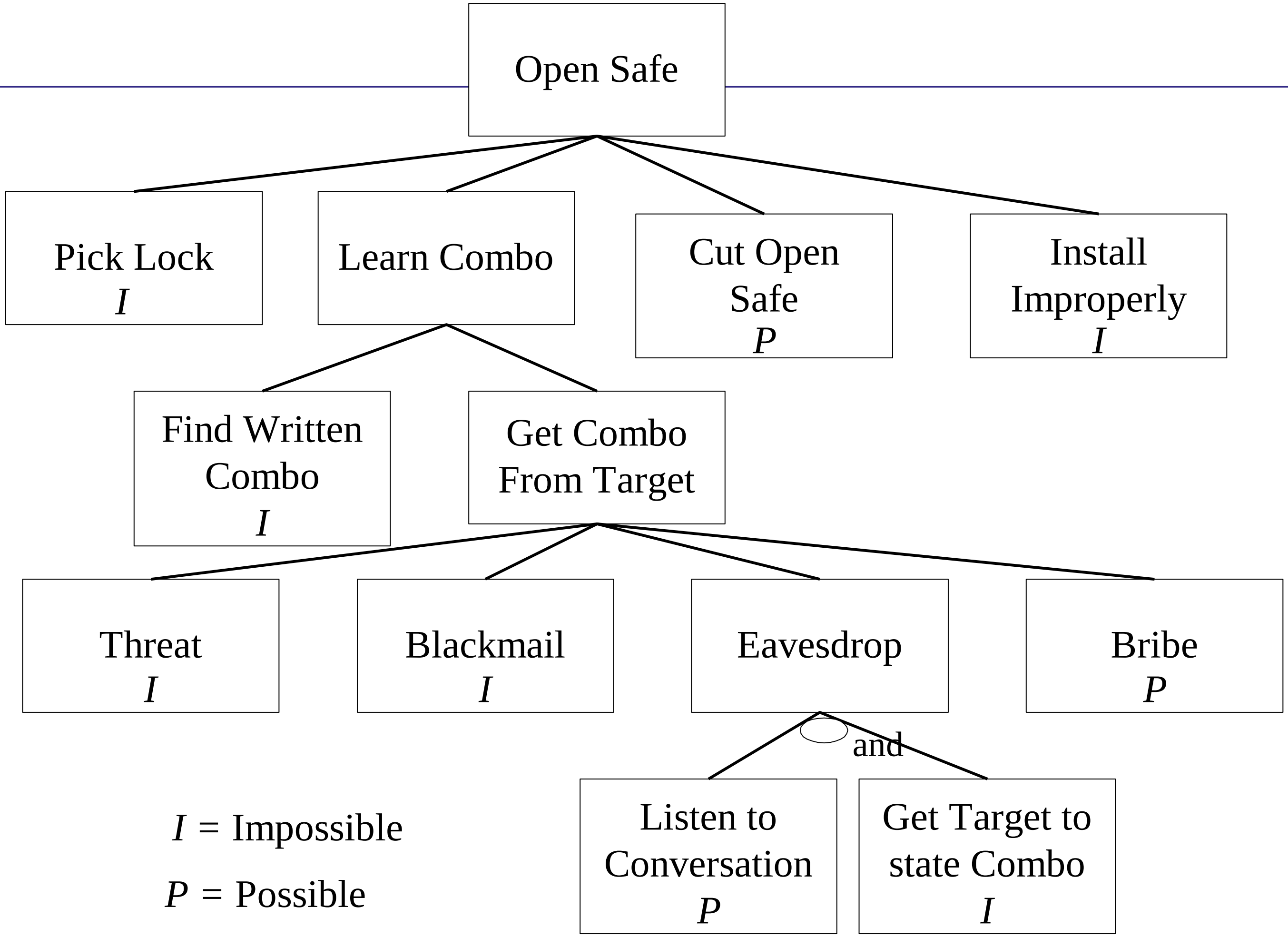
- Student -> *Malicious* Student
- Drawback - No systematic way of generating threats

Attack Tree Analysis [Schneier, 1999]

Represent attacks against a system in a tree structure, with the goal as the root node and different ways of achieving that goal as leaf nodes.

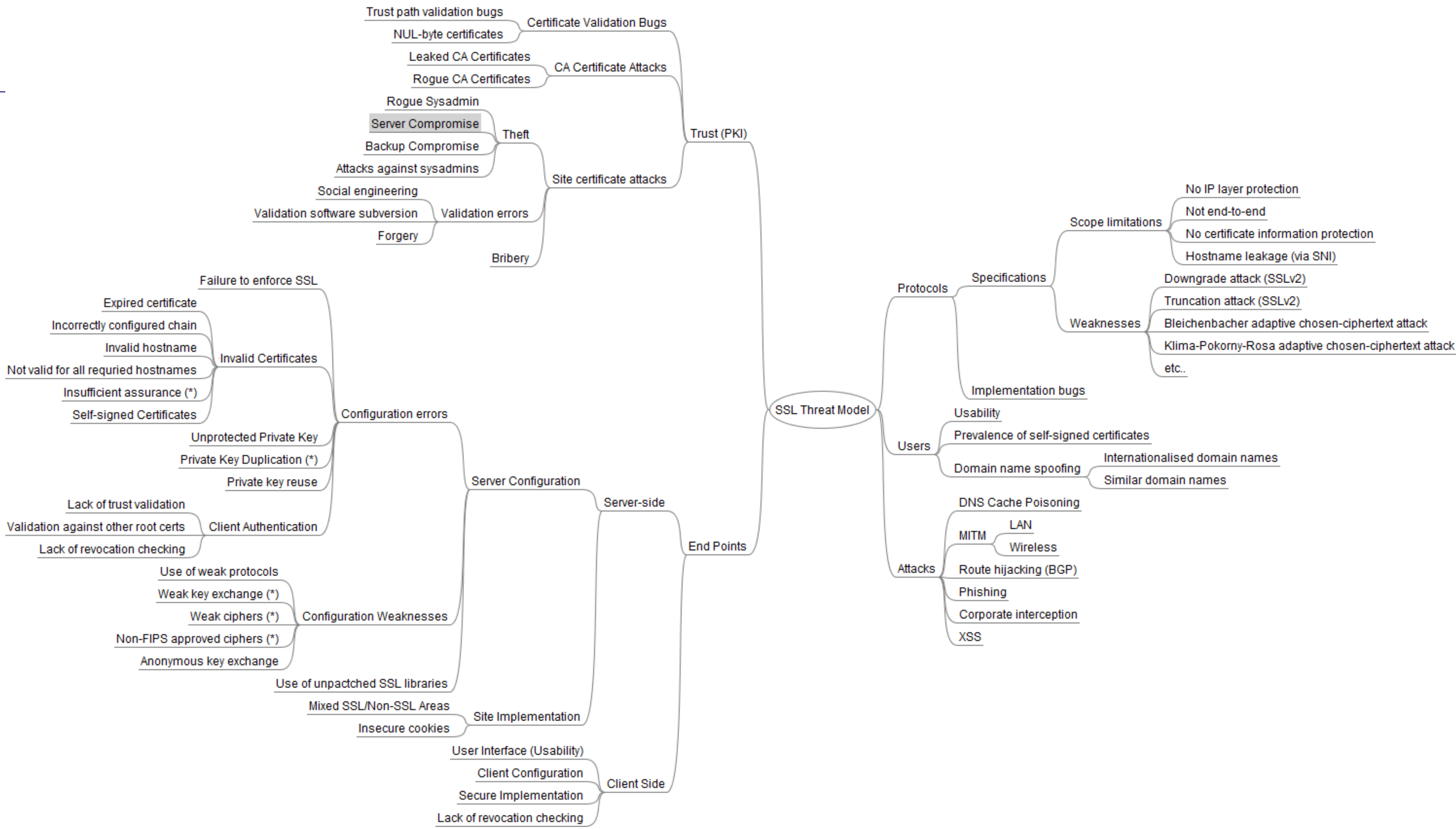
Steps in constructing Schneier’s attack trees are:

1. Identify the possible attack goals; that represent weaknesses in the system security.
2. Construct an attack tree for each attack goal.
3. Consider all possible attacks (sub goals) against the goal in AND-decomposition or OR-decomposition.
4. Repeat the process down the tree for each level of sub goals.



I = Impossible

P = Possible



- SSL Threat Model
  - Trust (PKI)
    - Certificate Validation Bugs
      - Trust path validation bugs
      - NUL-byte certificates
    - CA Certificate Attacks
      - Leaked CA Certificates
      - Rogue CA Certificates
    - Site certificate attacks
      - Theft
        - Rogue Sysadmin
        - Server Compromise
        - Backup Compromise
        - Attacks against sysadmins
      - Validation errors
        - Social engineering
        - Validation software subversion
        - Forgery
      - Bribery
  - End Points
    - Server-side
      - Server Configuration
        - Configuration errors
          - Failure to enforce SSL
          - Invalid Certificates
            - Expired certificate
            - Incorrectly configured chain
            - Invalid hostname
            - Not valid for all requried hostnames
            - Insufficient assurance (*)
            - Self-signed Certificates
          - Unprotected Private Key
          - Private Key Duplication (*)
          - Private key reuse
          - Client Authentication
            - Lack of trust validation
            - Validation against other root certs
            - Lack of revocation checking
        - Configuration Weaknesses
          - Use of weak protocols
          - Weak key exchange (*)
          - Weak ciphers (*)
          - Non-FIPS approved ciphers (*)
          - Anonymous key exchange
        - Use of unpactched SSL libraries
      - Site Implementation
        - Mixed SSL/Non-SSL Areas
        - Insecure cookies
    - Client Side
      - User Interface (Usability)
      - Client Configuration
      - Secure Implementation
      - Lack of revocation checking
  - Protocols
    - Specifications
      - Scope limitations
        - No IP layer protection
        - Not end-to-end
        - No certificate information protection
        - Hostname leakage (via SNI)
      - Weaknesses
        - Downgrade attack (SSLv2)
        - Truncation attack (SSLv2)
        - Bleichenbacher adaptive chosen-ciphertext attack
        - Klima-Pokorny-Rosa adaptive chosen-ciphertext attack
        - etc..
    - Implementation bugs
  - Users
    - Usability
    - Prevalence of self-signed certificates
    - Domain name spoofing
      - Internationalised domain names
      - Similar domain names
  - Attacks
    - DNS Cache Poisoning
    - MITM
      - LAN
      - Wireless
    - Route hijacking (BGP)
    - Phishing
    - Corporate interception
    - XSS

Source: https://blog.ivanristic.com/downloads/SSL_Threat_Model.png

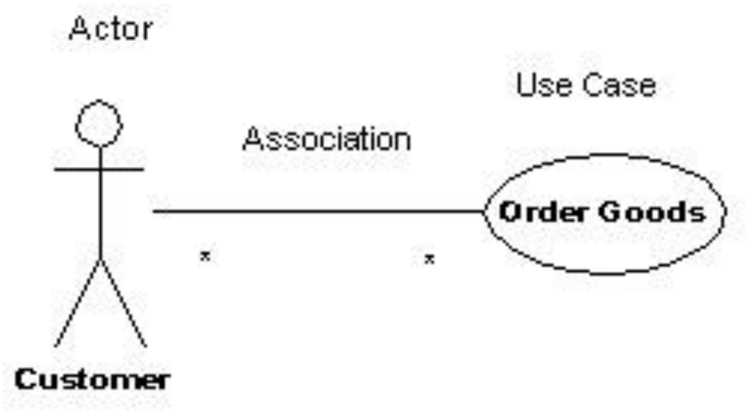
Deviational Techniques on Use Case Model [Srivatanakul et. al., 2004]

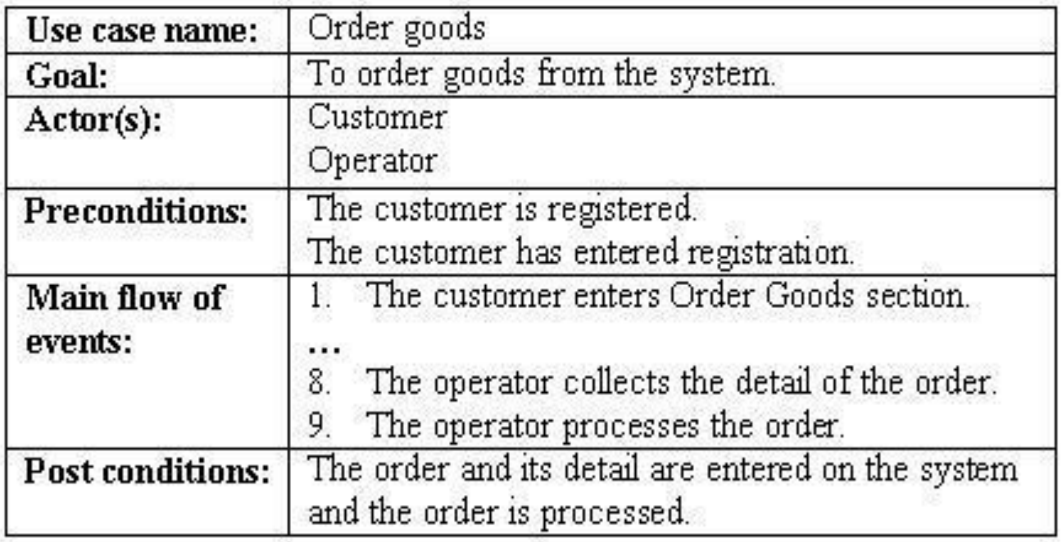
- Aim
  - Systematic analysis of security issues/ requirements from evidence produced from the system development

- Apply HAZOP (Hazard and Operability Analysis) to Use Case Model
  - Use of guidewords to prompt deviations
  - ‘NO’ – no action takes place
  - ‘NO’ – no fluid flowed

HAZOP – A safety technique that base analysis on the deviations (unintended or unexpected behaviours) of a system.

HAZOP and Use Case: example application





Use case name:	Order goods
Goal:	To order goods from the system.
Actor(s):	Customer Operator
Preconditions:	The customer is registered. The customer has entered registration.
Main flow of events:	1. The customer enters Order Goods section. … 8. The operator collects the detail of the order. 9. The operator processes the order.
Post conditions:	The order and its detail are entered on the system and the order is processed.

- Customer’s intent

MORE – the customer excessively orders goods

- Association

MORE – multiple sessions from one customer

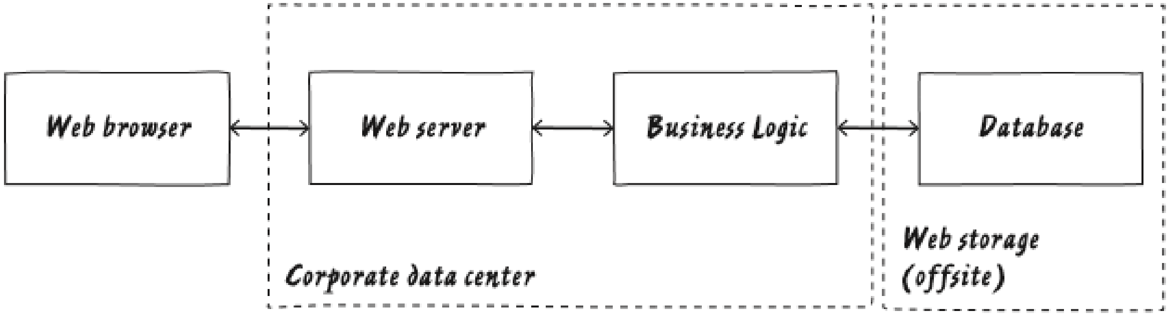
- Use case action

OTHER THAN – incorrect payment is sent out

STRIDE (Microsoft)

- ***S***_poofing_ – pretending to be something or someone you are not.
- ***T***_ampering_ **-** modifying something you’re not supposed to.
- ***R***_epudiation_ – claiming you didn’t do something.
- ***I***_nformation disclosure_- exposing information to unauthorized person.
- ***D***_enial of service_ - reducing the ability of valid users to access resources.
- ***E***_levation of privilege_ - when an unprivileged user gains privileged status.

Applying STRIDE : an example



- Spoofing
- Tampering
- Repudiation
- Information Disclosure
- Denial of Service
- Elevation of privilege

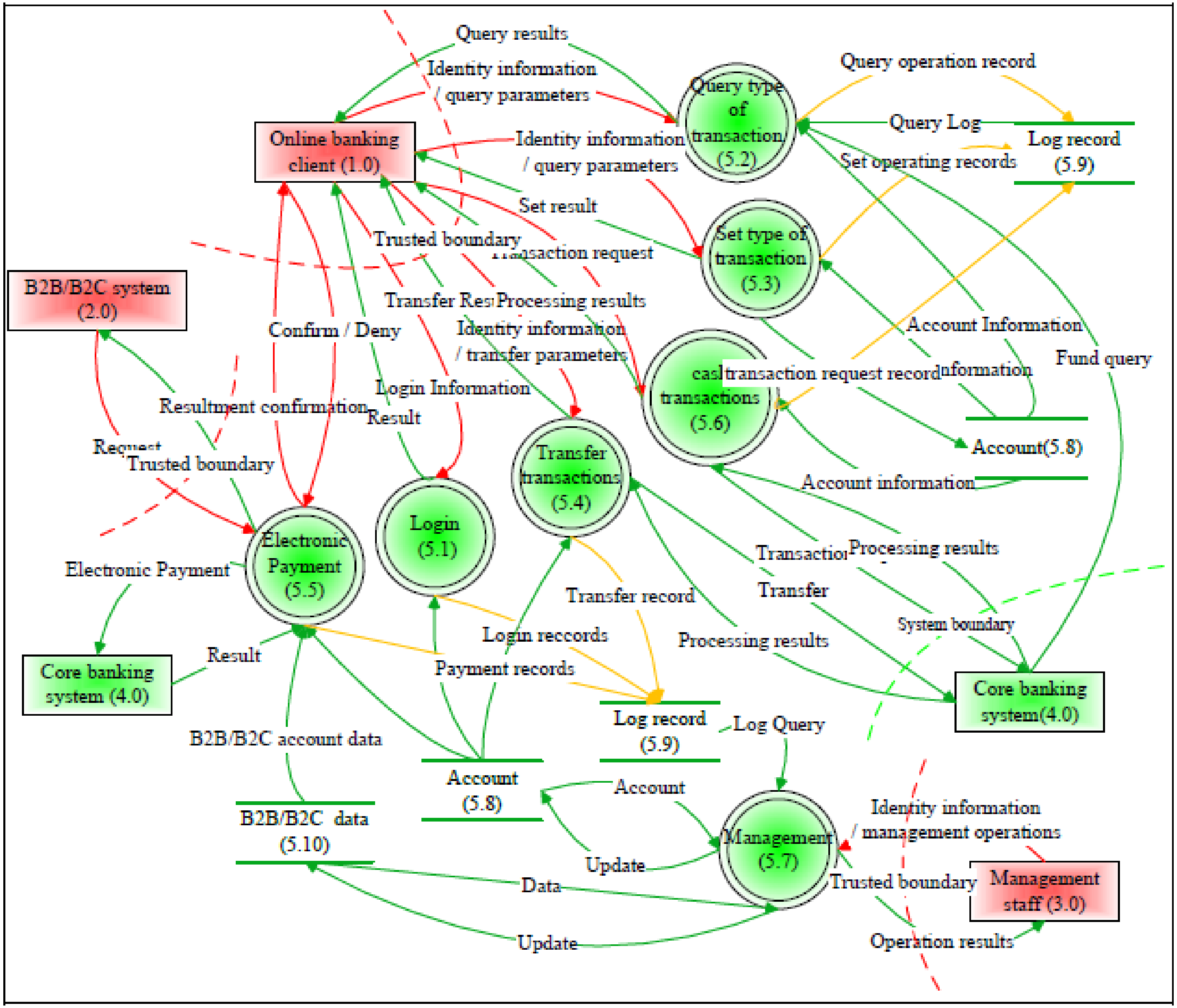
STRIDE is a tool to guide you to identify threats,
not to ask you to categorize what you've found!

The STRIDE per Element Approach

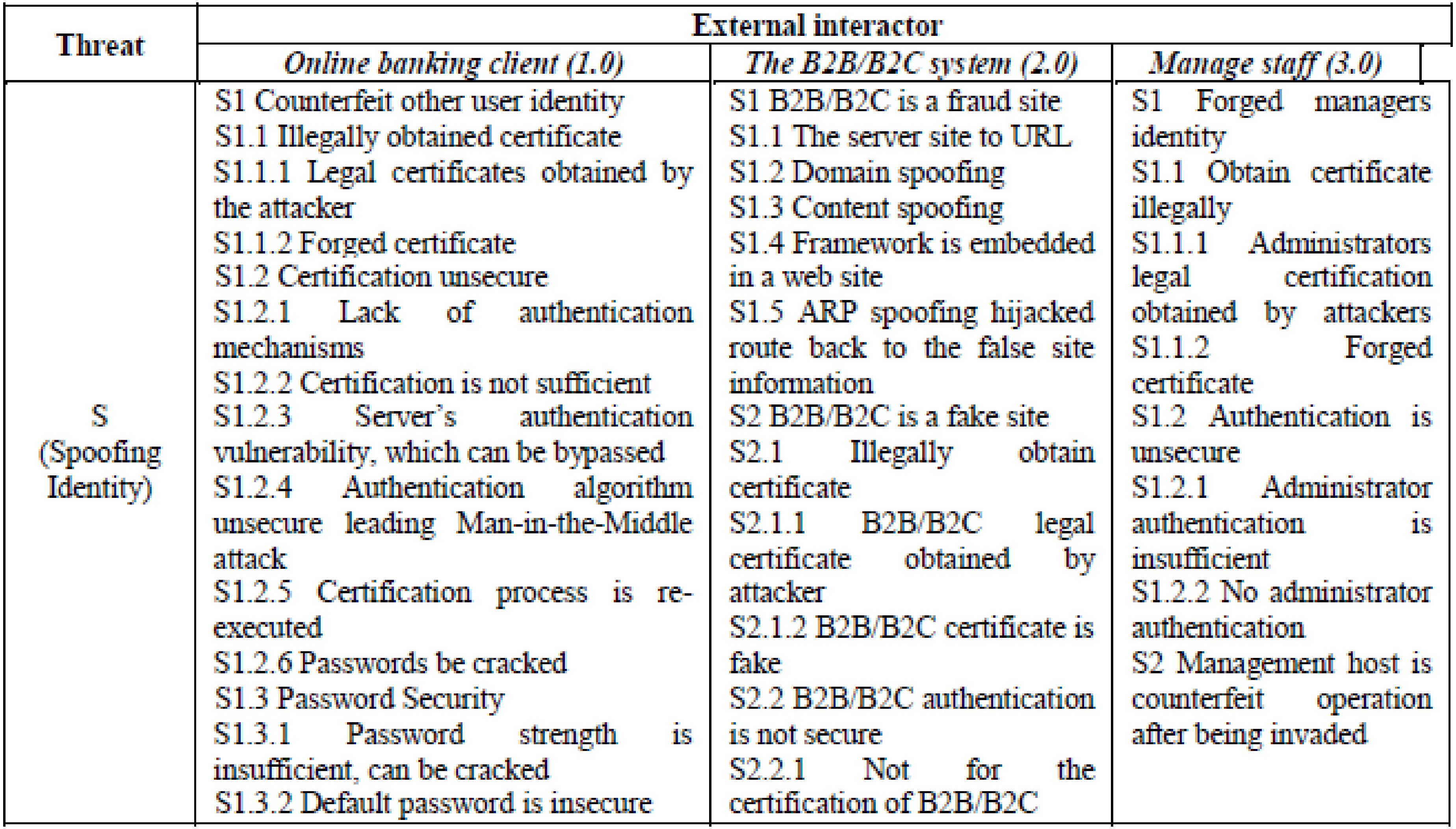
- For each element on the diagram (DFD)

  -> Apply STRIDE

Elements	S	T	R	I	D	E
External	✓		✓			
Process	✓	✓	✓	✓	✓	✓
Data Store		✓	✓ (logs)	✓	✓	
Data Flow		✓		✓	✓	

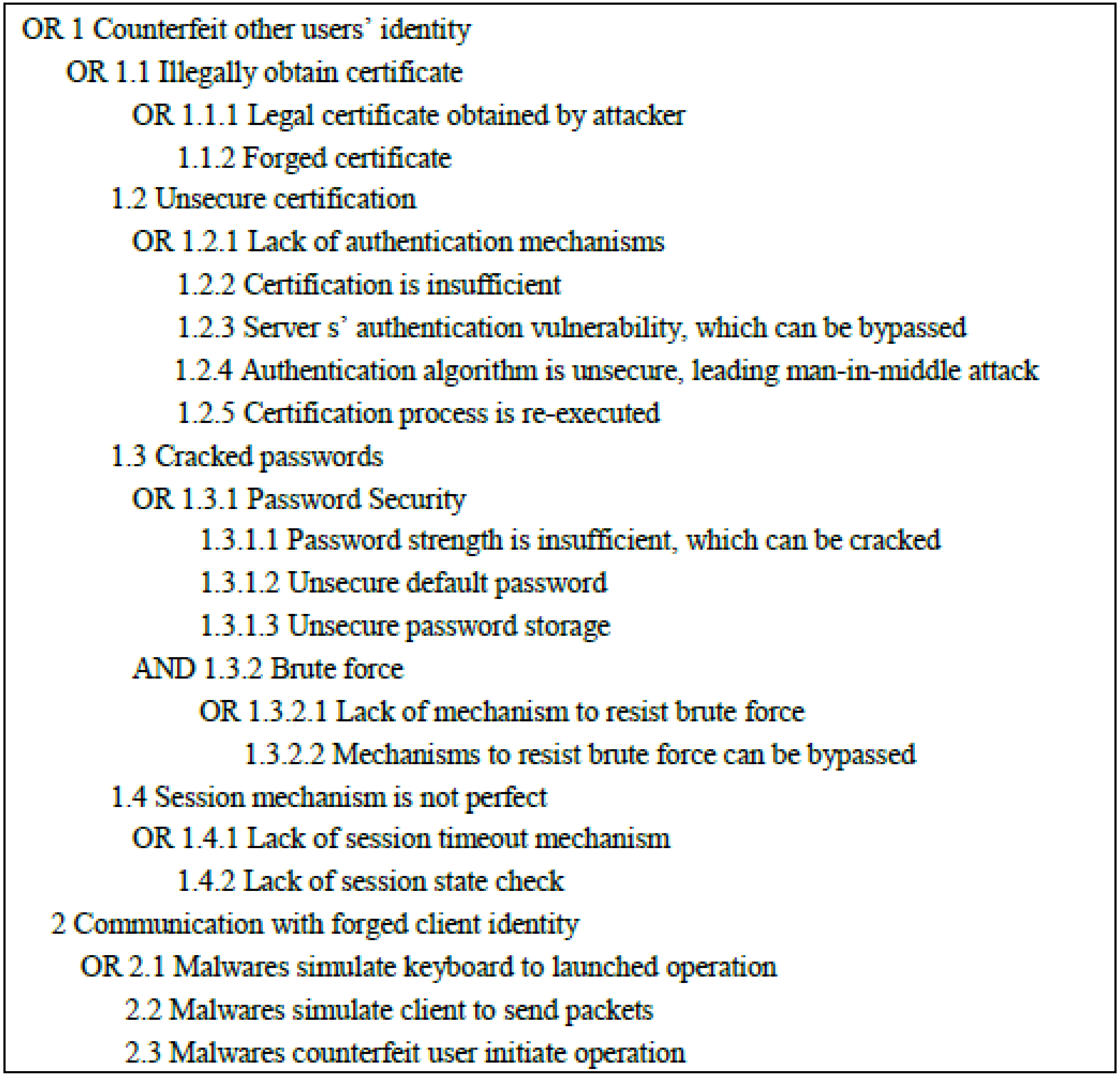


Xin, T., & Xiaofang, B. (2014)



Threat	External interactor		
	Online banking client (1.0)	*The B2B/B2C system (2.0)*	*Manage staff (3.0)*
S (Spoofing Identity)	S1 Counterfeit other user identity S1.1 Illegally obtained certificate S1.1.1 Legal certificates obtained by the attacker S1.1.2 Forged certificate S1.2 Certification unsecure S1.2.1 Lack of authentication mechanisms S1.2.2 Certification is not sufficient S1.2.3 Server's authentication vulnerability, which can be bypassed S1.2.4 Authentication algorithm unsecure leading Man-in-the-Middle attack S1.2.5 Certification process is re-executed S1.2.6 Passwords be cracked S1.3 Password Security S1.3.1 Password strength is insufficient, can be cracked S1.3.2 Default password is insecure	S1 B2B/B2C is a fraud site S1.1 The server site to URL S1.2 Domain spoofing S1.3 Content spoofing S1.4 Framework is embedded in a web site S1.5 ARP spoofing hijacked route back to the false site information S2 B2B/B2C is a fake site S2.1 Illegally obtain certificate S2.1.1 B2B/B2C legal certificate obtained by attacker S2.1.2 B2B/B2C certificate is fake S2.2 B2B/B2C authentication is not secure S2.2.1 Not for the certification of B2B/B2C	S1 Forged managers identity S1.1 Obtain certificate illegally S1.1.1 Administrators legal certification obtained by attackers S1.1.2 Forged certificate S1.2 Authentication is unsecure S1.2.1 Administrator authentication is insufficient S1.2.2 No administrator authentication S2 Management host is counterfeit operation after being invaded

Xin, T., & Xiaofang, B. (2014)



- OR 1 Counterfeit other users' identity
  - OR 1.1 Illegally obtain certificate
    - OR 1.1.1 Legal certificate obtained by attacker
    - 1.1.2 Forged certificate
  - 1.2 Unsecure certification
    - OR 1.2.1 Lack of authentication mechanisms
    - 1.2.2 Certification is insufficient
    - 1.2.3 Server s' authentication vulnerability, which can be bypassed
    - 1.2.4 Authentication algorithm is unsecure, leading man-in-middle attack
    - 1.2.5 Certification process is re-executed
  - 1.3 Cracked passwords
    - OR 1.3.1 Password Security
      - 1.3.1.1 Password strength is insufficient, which can be cracked
      - 1.3.1.2 Unsecure default password
      - 1.3.1.3 Unsecure password storage
    - AND 1.3.2 Brute force
      - OR 1.3.2.1 Lack of mechanism to resist brute force
      - 1.3.2.2 Mechanisms to resist brute force can be bypassed
  - 1.4 Session mechanism is not perfect
    - OR 1.4.1 Lack of session timeout mechanism
    - 1.4.2 Lack of session state check
- 2 Communication with forged client identity
  - OR 2.1 Malwares simulate keyboard to launched operation
  - 2.2 Malwares simulate client to send packets
  - 2.3 Malwares counterfeit user initiate operation

Xin, T., & Xiaofang, B. (2014)

What should you do about those things that can go wrong? - Address Threats

- **Mitigating Threats** – reducing the risk by making it harder for an attacker to take advantage of a threat (with countermeasure).
- **Eliminating threats** – removing the function/feature associated with the risk.
- **Transferring threats** – letting someone or something else handle the risk
- **Accepting the risk** – accepting the risk that is not worth the expense or cost.

The Interplay of Attacks, Mitigations, & Requirements

- There are threats that cannot be effectively mitigated
- What do you do when you find threats that violate your requirements and cannot be mitigated?
  - You’ll discover that some threats are hard or impossible to address, and you’ll adjust requirements to match.

Mitigation Strategies/Techniques: Examples

- Spoofing threats

Threat Target	Mitigation Strategy	Mitigation Technique
Spoofing a Person	Identification and authentication	Usernames, real names, or other identifiers: passwords, tokens, biometrics
Spoofing a network address	Cryptographic	HTTPS/SSL, IPsec

- Tampering threats

Threat Target	Mitigation Strategy	Mitigation Technique
Tampering with a file	Operating System	ACLs
Tampering with a network packet	Cryptographic	HTTPS/SSL, IPsec

Mitigation Strategies/Techniques: Examples (2)

- Repudiation threats

Threat Target	Mitigation Strategy	Mitigation Technique
No logs	Log	Log all the security-relevant information
Logs come under attack	Protect the logs	ACL

- Information Disclosure threats

Threat Target	Mitigation Strategy	Mitigation Technique
Directory or filename	Leverage the OS	ACLs
Network monitoring	Encryption	HTTPS/SSL, IPsec

Did you do a decent job of analysis? - Check your work.

- Checking the model
  - Ensure that the final model matched with what you built.

“Sometimes we connect to this web service via SSL, and sometimes we fall back to HTTP,”

- Checking each threat
  - Did you do the right thing with each threat you found?
  - Have you found all the threats you should find?

- Checking your tests
  - Ensure that you have built a good test to detect the problem.

How to make sure that you have a realistic threat model?

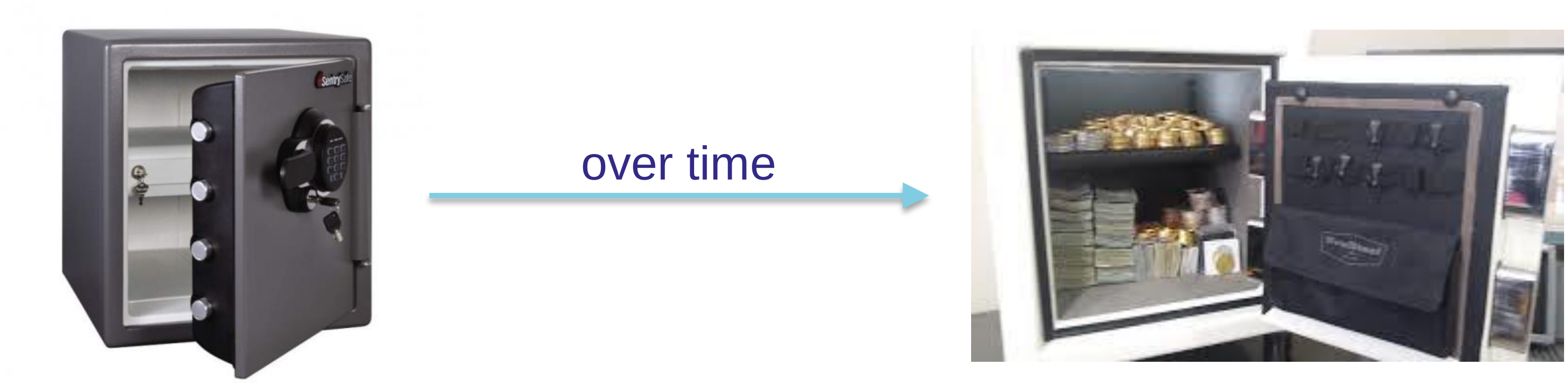
- Use other threat models as starting point (for a similar system)
- Challenge assumptions you are making in the threat model
- Keep updated on the new possible attacks and exploits
- Consider the consequences of failure: cost vs. probability

Why threat model?

- Understanding Security Requirements
    - helps ask “Is this really a requirement?”
- Find problems when there is time to fix them
    - helps find design issues early in the process
    - finding them early lets you avoid re-engineering
- Build mitigations into the design
- Addresses other issues
    - Threat modeling will lead you to categories of issues that other tools will not find e.g. errors of omission

Final Notes

- Changing Threat Model – keep track of when your threat model changes
  - Design decision changes
  - New features or functionality



over time

References

- Myagmar, S., Lee, A. J., & Yurcik, W. (2005). Threat modeling as a basis for security requirements. In *Symposium on requirements engineering for information security (SREIS)* (Vol. 2005, pp. 1-8).
- Shostack, A. (2014). *Threat modeling: Designing for security*. John Wiley & Sons.
- McDermott, J. and Fox, 1999. Using Abuse Case Models for Security Requirement Analysis. *In*: *Proceedings of 15th Annual Computer Security Applications Conference*, Phoenix, Arizona.
- Schneier, B. (1999). Attack trees. *Dr. Dobb's journal*, *24*(12), 21-29.
- Srivatanakul, T., Clark, J. A., & Polack, F. (2004). Effective security requirements analysis: Hazop and use cases. In *International Conference on Information Security* (pp. 416-427). Springer Berlin Heidelberg.
- **Xin, T., & Xiaofang, B. (2014). Online Banking Security Analysis based on STRIDE Threat Model. *International Journal of Security and Its Applications*, *8*(2), 271-282.**
- S. Harris, *CISSP all-in-one exam guide*. McGraw-Hill, Inc., 2016.