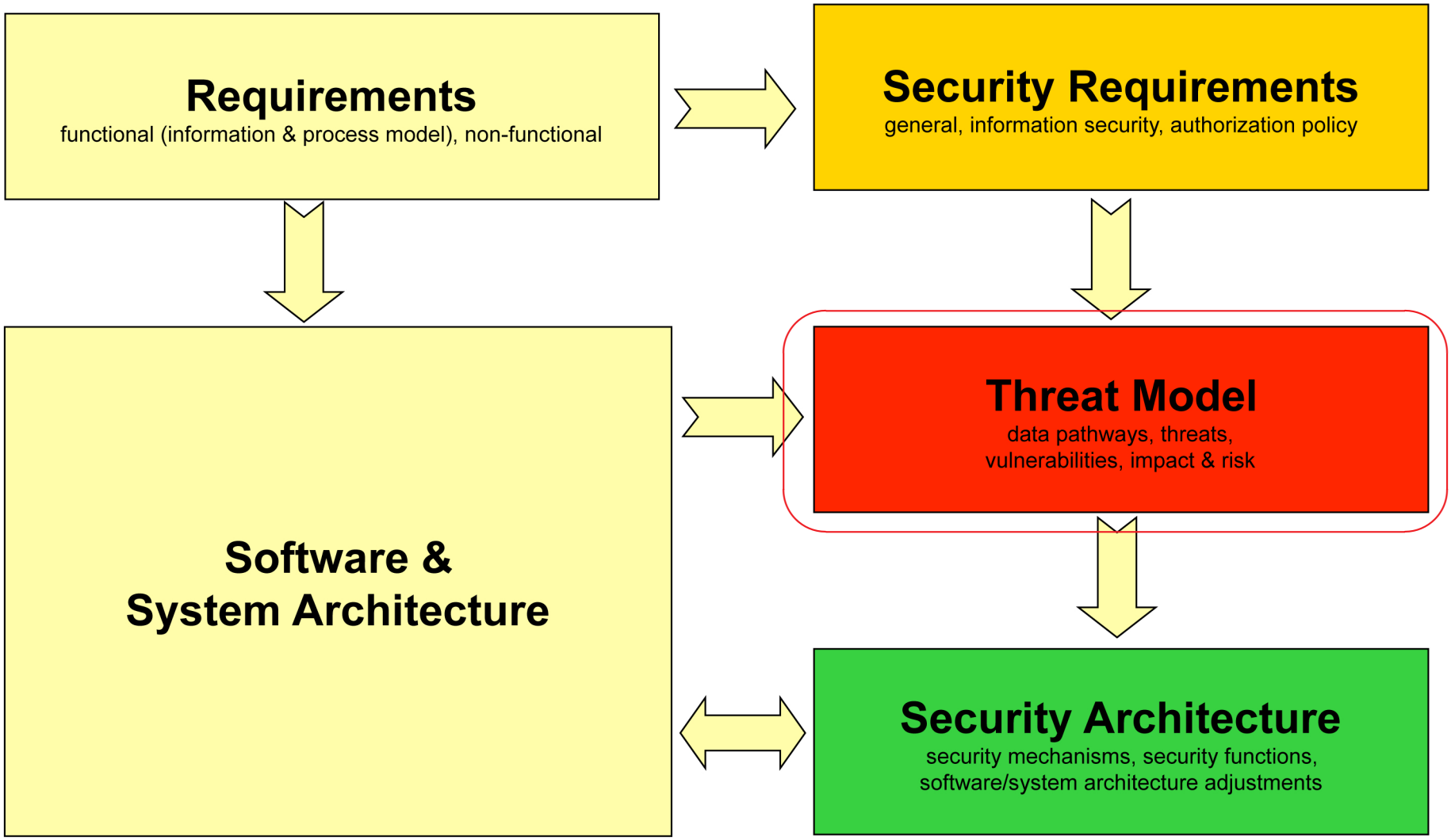
Threat Modeling

Security Engineering

David Basin
ETH Zurich

Context


Requirements
functional (information & process model), non-functional
Security Requirements
general, information security, authorization policy
Software &
System Architecture
Threat Model
data pathways, threats,
vulnerabilities, impact & risk
Security Architecture
security mechanisms, security functions,
software/system architecture adjustments

Recap

What must we know to assess a system's security?

1. Enterprise's assets and the system itself
2. **Vulnerabilities that can be exploited**
3. **Impact of exploits**
4. Effectiveness of countermeasures
5. Resulting risks

In this module we focus on (2) and (3)

Aim of module

- Understand how to identify threats and their impact
- See relationships to techniques from traditional safety analysis

  And understand key differences
- See the role of **models** in analyzing risks

  As well as different representation possibilities

Road map

☞ **Classical approaches to modeling faults**

- **FMEA**
- Fault trees

- Risk analysis using design models
- Attack modeling
- Experience with techniques

Why study safety engineering?

- Security engineering is not yet a mature discipline
- Safety engineering and associated methods are better established
  - **Failure Modes and Effects Analysis:** bottom-up, textual
  - **Fault Tree Analysis:** top-down, graphical
- Methods may be used individually or in combination
- Objective is to identify potential problems before they occur

FMEA

- **F**ailure **M**ode and **E**ffect **A**nalysis

  Also called FME**C**A, to emphasize **C**riticality (= risk assessment)

- Goal: identify possible root causes of faults early

- Bottom-up (inductive) addressing system, design, and process

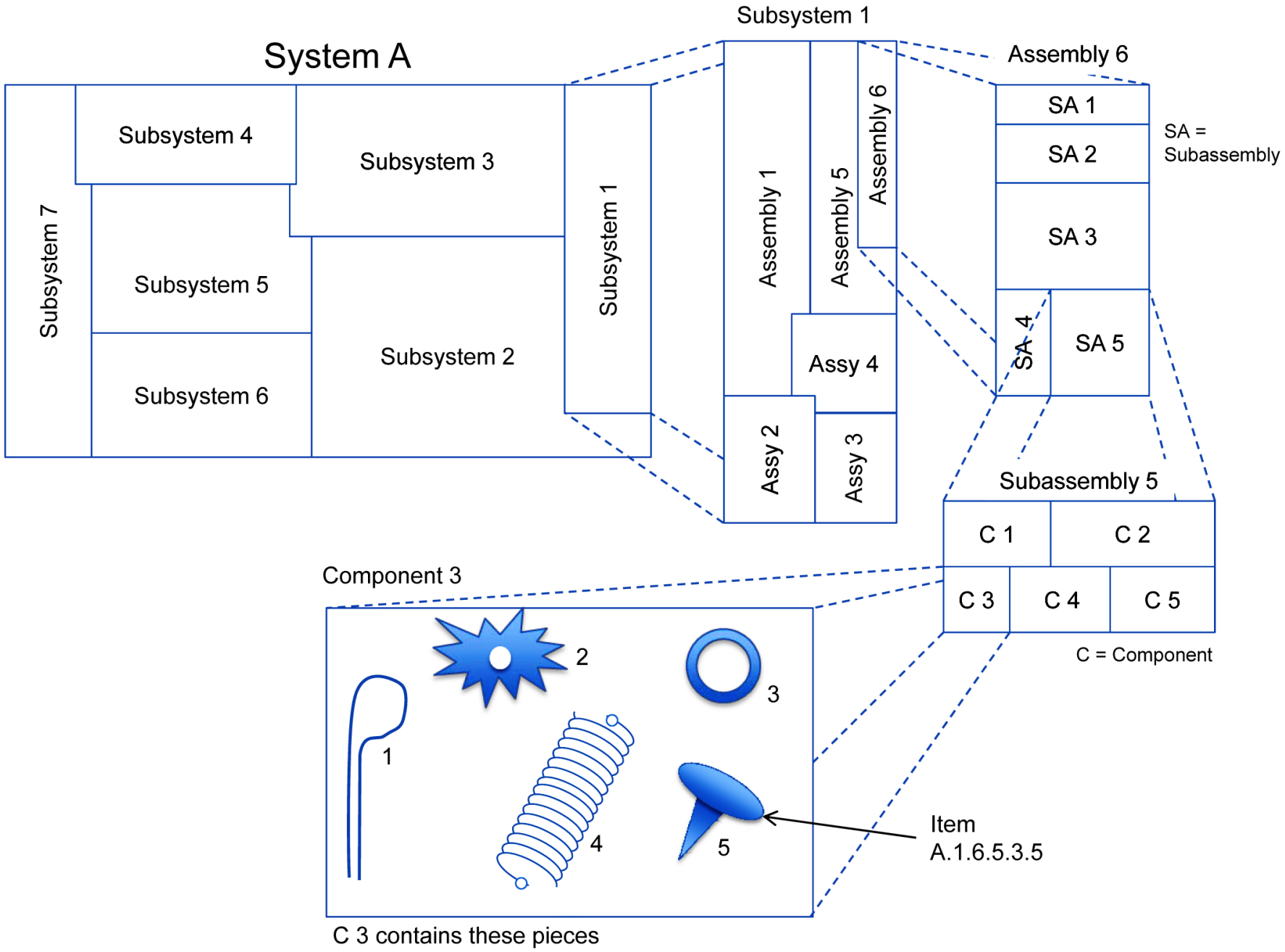
- Choose component/subsystem/parts and analyze possible failures and their effect on the rest of the system

  Yields qualitative risk assessment and actions for improvements

- Often used in mechanical or hardware-oriented systems
  - E.g., aerospace, automotive, and medical industries
  - Also in software industry, although less well established

Traditional example — mechanical systems

Traditional example — mechanical systems



FMEA elements

- Consider each part of the entire system
- How may it or its subsystems fail?

  Fault: Inability to function or an undesired functionality
  Failure: Occurrence of a fault
  Failure mode: manner which fault occurs, i.e., way element fails,
  e.g., breakage, compromised integrity, ...

- Analysis based on **historical data**, expert opinions, ...
- Rank failure modes, e.g., 1–10

  Occurrence: relative probability of malfunction occurring
  Severity: Relative severity of worst possible outcome w.r.t. system functions
  (Non)Detectability: Probability that failure will be detected/corrected

- **R**isk **P**riority **N**umber = Occurrence $\times$ Severity $\times$ Detectability
  E.g., RPN 1, ..., 1000

Occurrence

Notional probability of failure	Evaluated failure rates	Rank
Remote: failure is unlikely. No failure ever associated with almost identical processes	1 in 1,500,000	1
Very low: only isolated failures associated with almost identical processes	1 in 150,000	2
Low: isolated failures associated with similar processes	1 in 15,000	3
Moderate: generally associated with processes similar to previous processes failures, but not in "major" propositions	1 in 2,000	4
	1 in 400	5
	1 in 80	6
High: generally associated with processes similar to previous processes that have often failed	1 in 20	7
	1 in 8	8
Very high: failure is almost inevitable	1 in 3	9
	1 in 2	10

Severity

Effect	Criteria	Severity of effect	Rank
None		No Effect	1
Very minor	Minor disruption to production line	A portion of the product may have to be reworked. Defect not noticed by average customers; cosmetic defects	2
Minor	Minor disruption to production line	A portion of the product may have to be reworked. Defect not noticed by average customers; cosmetic defects	3
Very low	Minor disruption to production line	The product may have to be sorted and reworked. Defect noticed by average custmers; cosmetic defects	4
Low	Some disruption to product line	100% of product may have to be reworked. Customer has some dissatisfaction. Item is fit for purpose but may have reduced levels of performance	5
Moderate	Some disruption to product line	A portion of the product may have to be scrapped. Customer has some dissatisfaction. Item is fit for purpose but may have reduced levels of performace	6
High	Some disruption to product line	A portion of product may have to be scrapped. Item is useable but at reduced levels of performance	7
Very high	Major disruption to production line	100% of product may have to be scrapped. Loss of primary function. Item unusable. Customer very dissatisfied.	8
Hazard with warning	May endanger machine or operator	Failure occurs with warning. The failure mode affects safe operation and involves noncompliance with regulations	9
Hazard without warning	May endanger machine or operator	Failure occurs without warning. The failure mode affects safe operation and involves noncompliance with regulations	10

(Non)Detectability

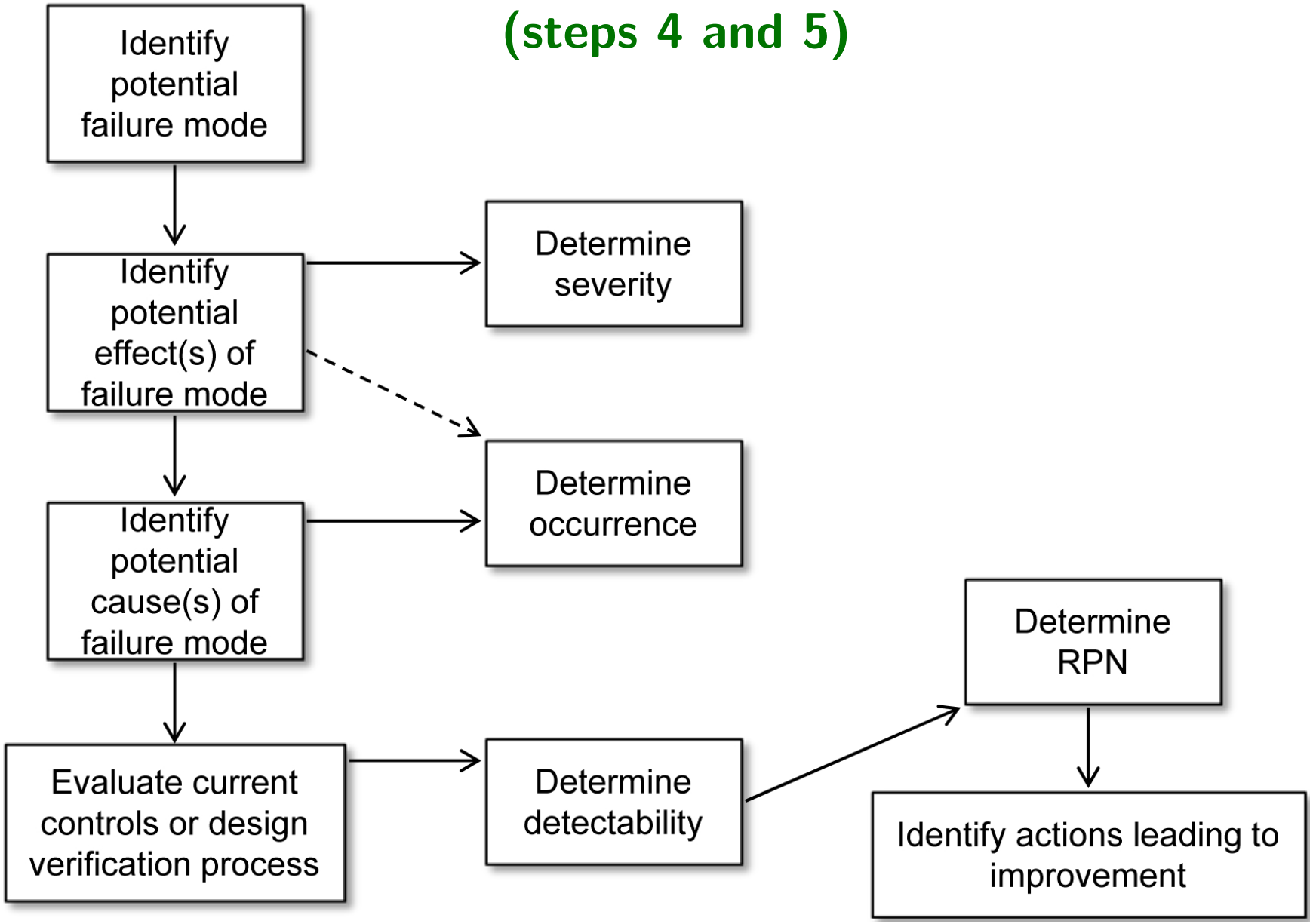
Detection	The likelihood the controls will detect a defect	Rank
Almost certain	Current controls are almost certain to detect the failure mode. Reliable detection controls are known with similar processes	1
Very high	Very high likelihood that the current controls will detect the failure mode	2
High	High likelihood that the current controls will detect the failure mode	3
Moderately high	Moderately high likelihood that the current controls will detect the failure mode	4
Moderate	Moderate likelihood that the current controls will detect the failure mode	5
Low	Low likelihood that the current controls will detect the failure mode	6
Very low	Very low likelihood that the current controls will detect the failure mode	7
Remote	Remote likelihood that the current controls will detect the failure mode	8
Very remote	Very remote likelihood that the current controls will detect the failure mode	9
Almost impossible	No known controls available to detect the failure mode	10

FMEA Procedure

Example: MIL-STD-1629A, Military Standard Procedure for performing an FMECA, 1980 (dating back to 1949!)

1. Define system to be analyzed
   - Mission, environment profiles
   - Internal and interface functions, expected performance, constraints, failure definitions
2. Construct block diagrams of systems
3. Identify potential item and interface definitions
4. Evaluate and rank failures
5. Identify possible causes and appropriate actions
6. Corrective design: take actions to eliminate/reduce high-risks FMs
7. Documentation

FMEA procedure
(steps 4 and 5)



Source: P. Hammett, FMEA, www.fmeainfocentre.com/download/umich.pdf

Failure categories: example MIL-STD-1629A
(alternative to previous severity scale)

Category I: Catastrophic

A failure which may cause death or weapon system loss
e.g., aircraft, tank, missile, ship, etc.

Category II: Critical

A failure which may cause severe injury, major property damage, or major system damage, resulting in mission loss.

Category III: Marginal

A failure which may cause minor injury, minor property damage, or minor system damage, resulting in delay or loss of availability or mission degradation.

Category IV: Minor

A failure not serious enough to cause injury, property damage, or system damage, resulting in unscheduled maintenance or repair.

Example: air bags

Part of process name: Automotive passenger air bag system					Suppliers & plants affected:				
Design / Mfg responsibility:					Model date:				
Other areas involved:					Engineering change level:				
Process operation, product function or purpose	Potential failure mode	Potential effect(s) of failure	SEV	Potential cause(s) of failure	OCC	Current controls evaluation method	DET	RPN	Recommended action(s)
Inflate air bag	Bag does not open on impact	Injure passenger	8	Sensor is not functioning properly	2	Light to notify that system is malfunctioning	6	96	Add redundant sensor to monitor impact
Restrain passenger	Occupant unable to withstand inflation force	Injure lightweight passenger	8	Passenger not wearing seat belt	4	None	10	320	1) Install switch which deactivates air bag system unless seat belt is worn
		Bruise passenger in crash	3	Force regulator not working	2	Repeatablility test in lab	3	18	2) Consumer education of air bag system potential failures

Source: P. Hammett, FMEA, www.fmeainfocentre.com/download/umich.pdf

FMEA discussion

- It appears to work well in practice
- Like all other techniques, no guarantees
  - Garbage in $\Rightarrow$ garbage out
  - Easy to overlook human error, effect of hostile environments, ...
- Prerequisites
  - Understanding how the system works
  - Done when design/architecture is available
  - Experience with possible problems (historical data)
- Leads to facilitated discussions with different group members
- Unique to FMEA: probability of detection (relevant for security!)

Road map

☞ **Classical approaches to modeling faults**

- FMEA
- **Fault trees**

- Risk analysis using design models
- Attack modeling
- Experience with techniques

Fault tree analysis

- Goal: identify conditions leading to system failure (top level event)
- Developed at Bell Labs (1960s) for missile launch-control systems

  Now also used in nuclear, medical, and aerospace industries
- DIN and IEC standards
- Aims at finding the **sources** of a system failure
- Deductive top-down method
- Quantitative and qualitative
- Graphical representation of causal relationships

Prerequisites

- Understand how the system works

  E.g. from an FMEA

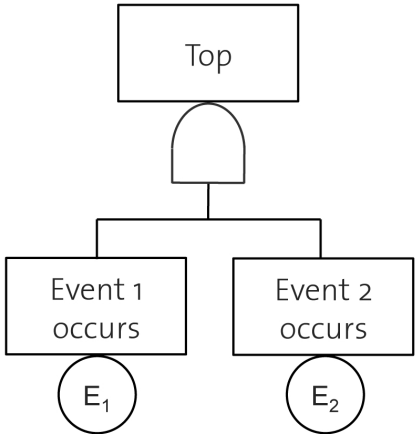
- Done when design/architecture is available

- Facilitated discussions with different group members

Procedure

- Input: plan of the system and FMEA, if existent
- Determine undesired (top-level) events, i.e., a failure/malfunction
- Identify event(s) that lead to the top-level event
  - These are lower-level failures
  - Apply recursively
- Leaves: possible causes, e.g., a component failure or environmental condition
- Symbols used: AND/OR; possibly probabilities
- Identify cut sets: those events that together lead to system failure
- Perform quantitative or qualitative analysis on resulting tree

Kinds of events



- **Primary events** (leaves)

  Basic events: no precursor; probabilistic

  e.g., bits flipped by cosmic rays

  Undeveloped events: no major effect alone on the system

  e.g., indication lamp fails

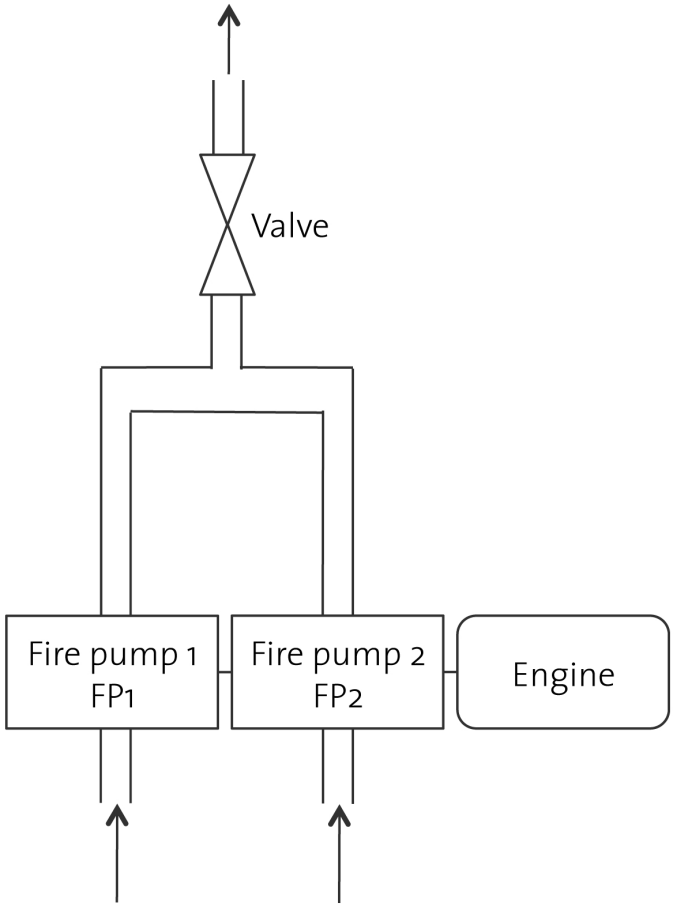
  External events: expected to happen; not a fault

- **Intermediate events:** Link primary or intermediate events via AND/OR gates

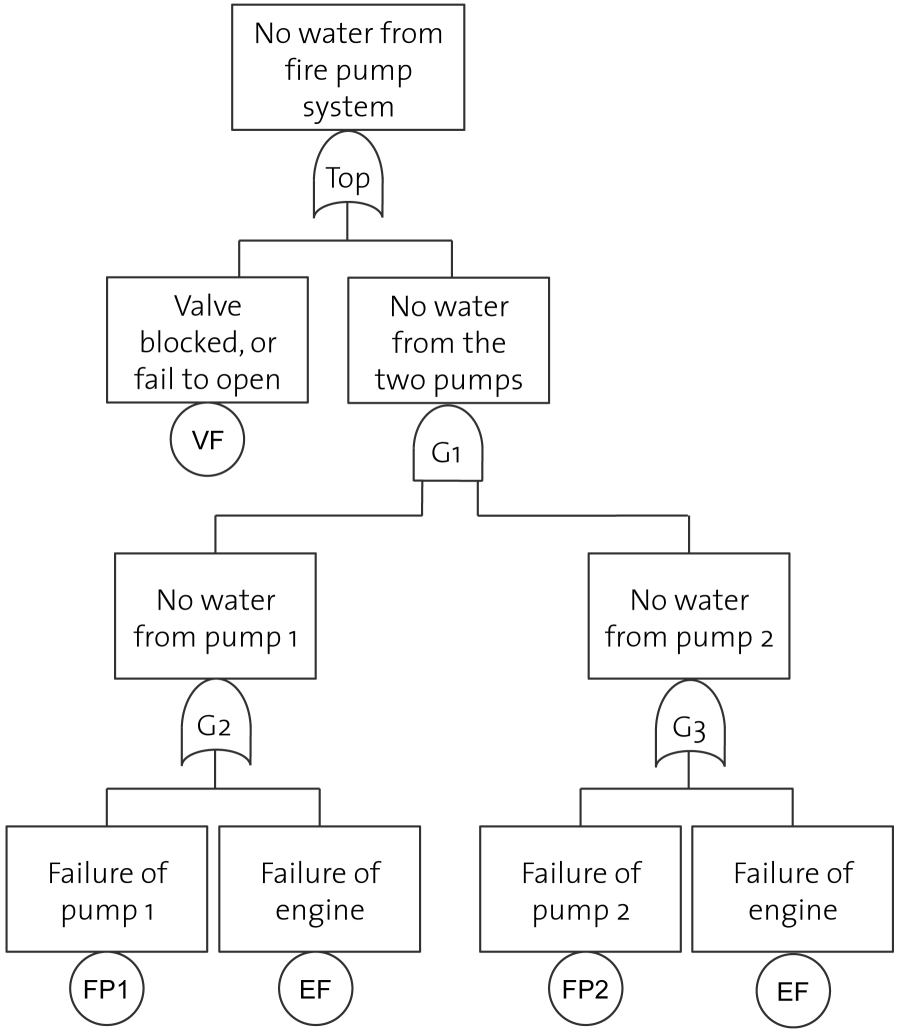
- **Expanded events:** Need a separate fault tree to explain

Example

System Schematic

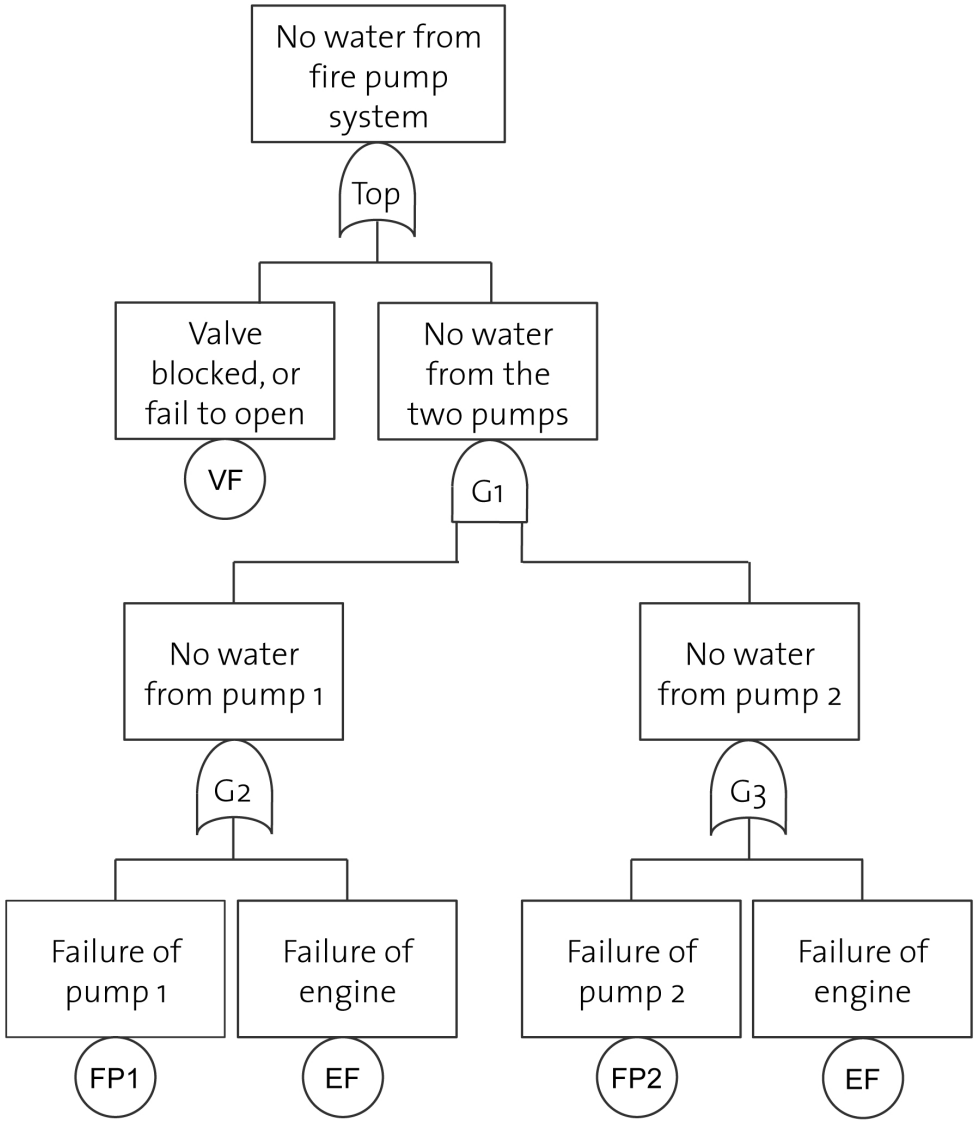


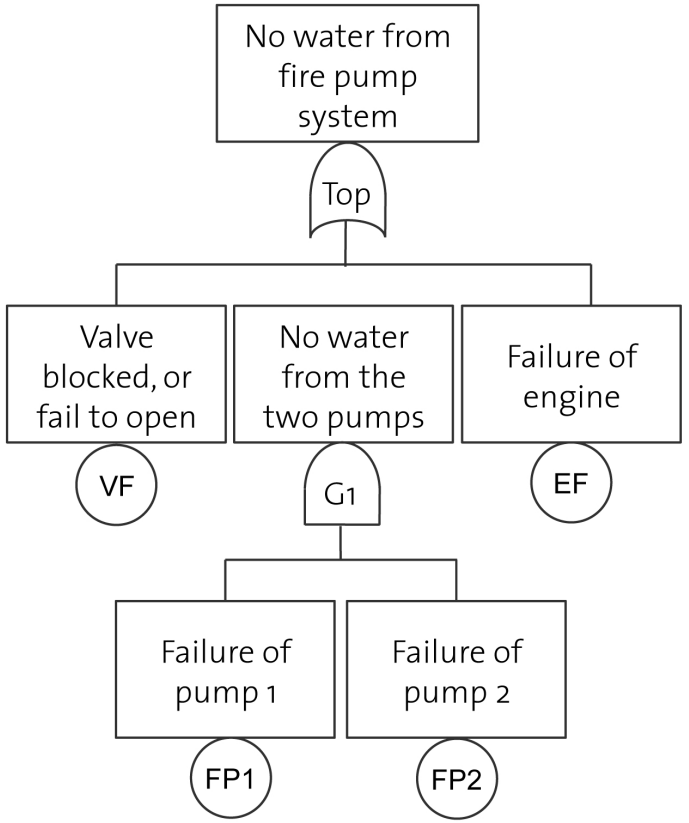
Fault Tree



Source: Source: M. Rausand, Fault Tree Analysis, 6/2004
http://www.ntnu.no/ross/srt/slides/fta.pdf

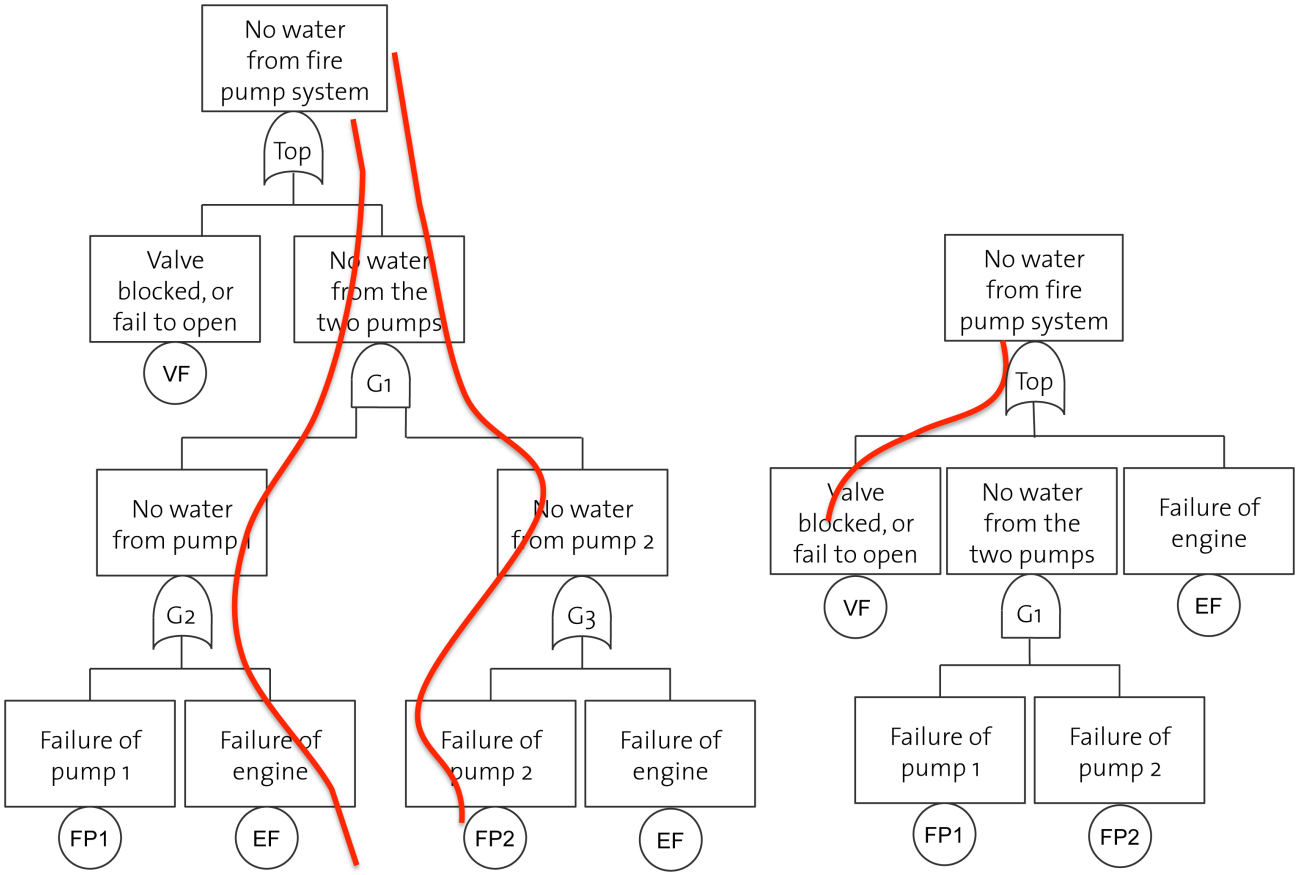
Equivalent fault trees





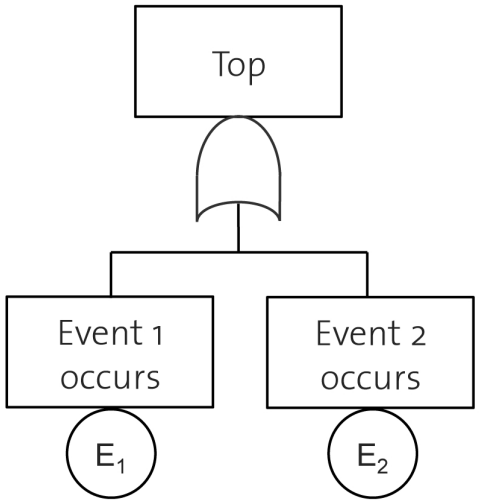
Source: M. Rausand, Fault Tree Analysis, 6/2004
http://www.ntnu.no/ross/srt/slides/fta.pdf

Cuts

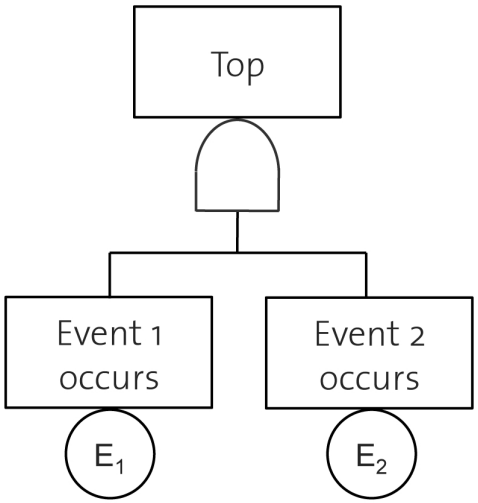


- A **cut** is a set of events that, taken together, lead to the top level event
- A **minimal cut** is a cut that is no longer a cut if an element is removed
- In general, there are many cuts and many minimal cuts

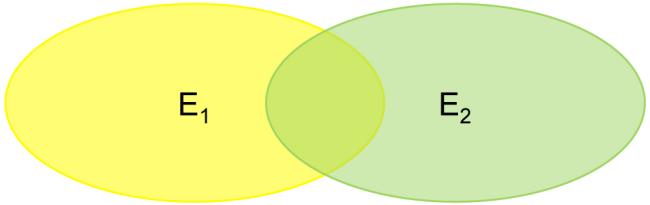
Quantitative analysis



$$1 - \prod_{j=1}^{M} (1 - P(E_j))$$



$$\prod_{j=1}^{M} P(E_j)$$



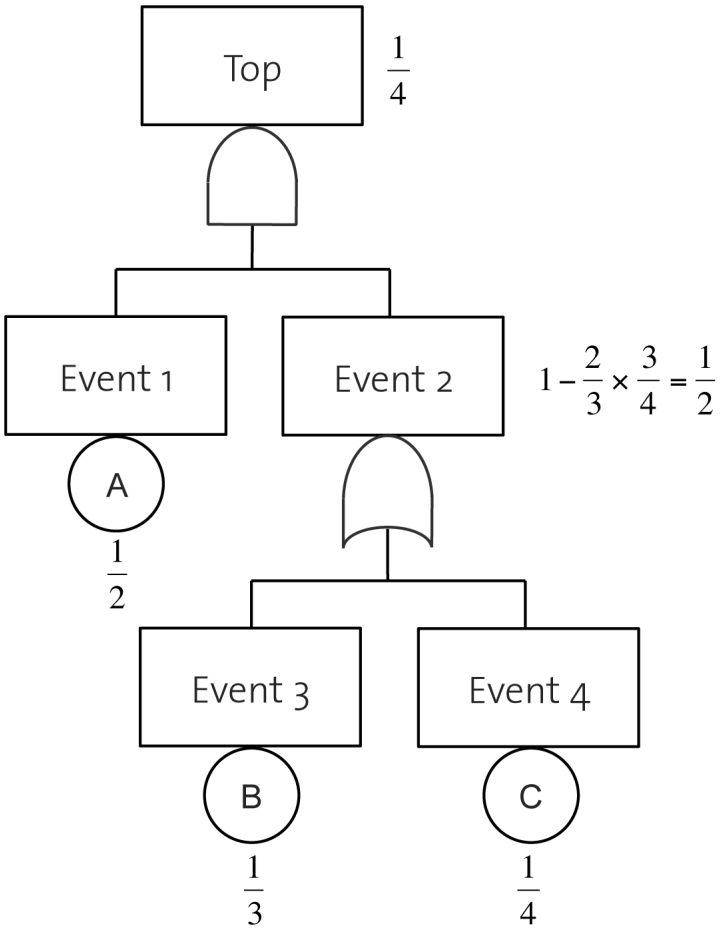
- Assume E_1 and E_2 are independent

  AND: $P(E_1 \cap E_2) = P(E_1) \times P(E_2)$
  OR: $P(E1 \cup E_2) = P(E_1) + P(E_2) - P(E_1) \times P(E_2)$

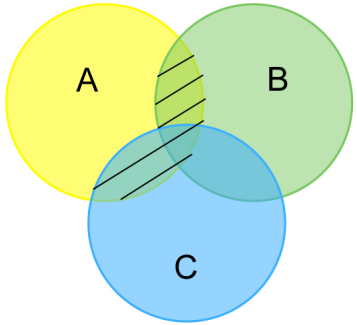
- For (minimal) cut: multiply probabilities

  This gives probability of failure happening this way.

Example



$P(\{A, B\})=\frac{1}{6}$ $P(\{A, C\})=\frac{1}{8}$ $P(\{A, B, C\})=\frac{1}{24}$



P({A, B} or {A, C})

= P({A, B}) + P({A, C}) – P({A, B, C})

$$=\frac{1}{6}+\frac{1}{8}-\frac{1}{24}=\frac{1}{4}$$

Relationship to security

- For security: focus on what should **not** happen, i.e. **misuse cases**
- **FTA** starts with an undesired top-level event representing a violated security requirement
  - Decomposing it into possible causes may lead to lower-level causes and suggest new security requirements
  - This is the basis of **attack trees** (coming up!)
- **FMEA** may indicate conditions that are not controllable by the system, e.g., security breach at one particular point
  - This gives rise to new requirements for the involved components
  - Closest analogy is **data pathways** where we aggregate requirements as we move from classes to components to systems (coming up!)

Road map

- Classical approaches to modeling faults
  - FMEA
  - Fault trees

☞ **Risk analysis using design models**

- Attack modeling

- Experience with techniques

Safety versus security

- **Safety:** failures arise from faults, e.g., when a critical part breaks.
- **Security:** "failures" are the unwanted events that occur when a threat agent attacks a system by exploiting a vulnerability
- How do we determine the relevant threats against a system?

  We begin by showing how to identify the critical parts

Requirements for understanding threats

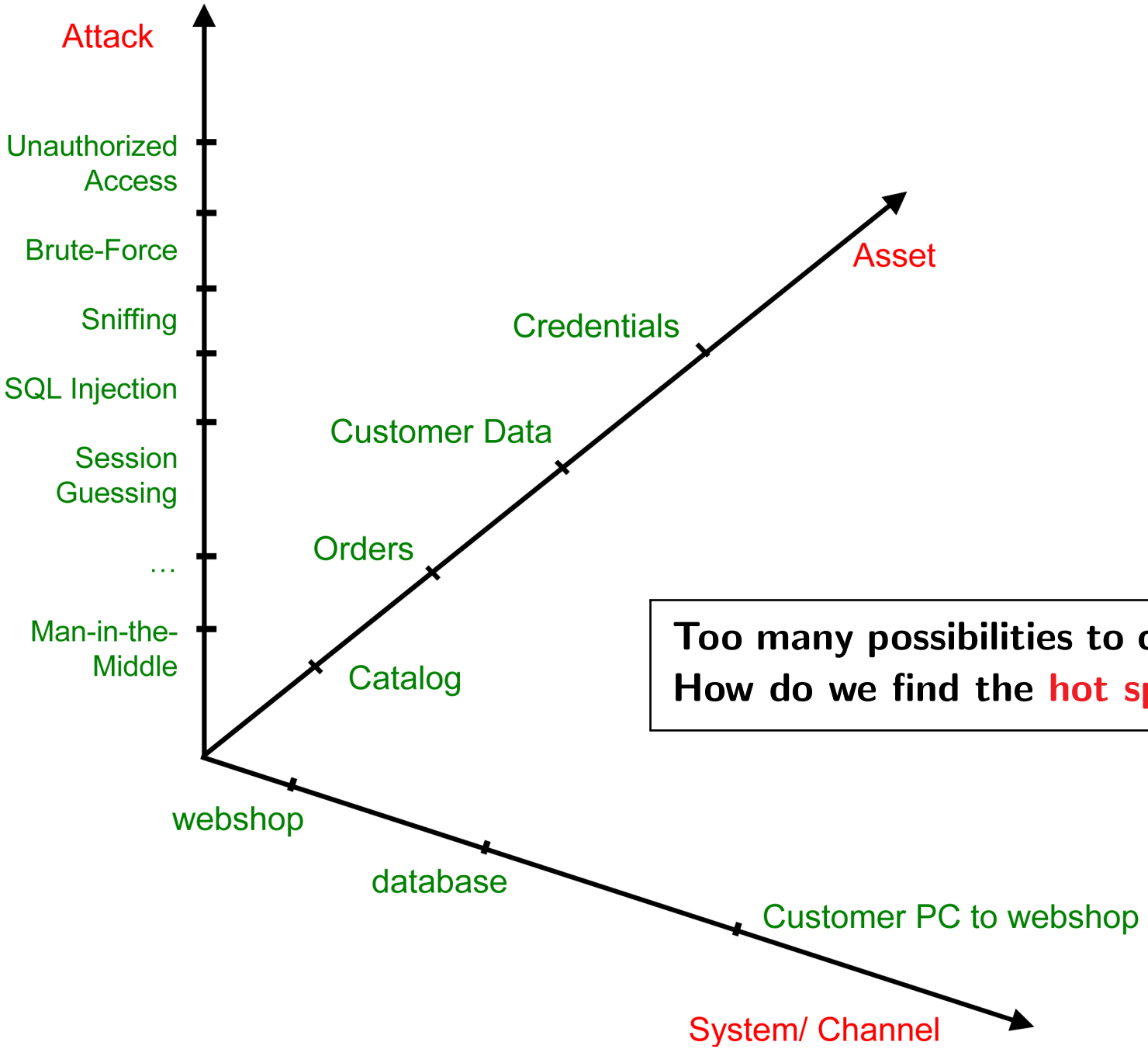
- Identify **relevant** threat agents
  - Network attacker vs. inside attacker
  - Script kiddies vs. organized criminals
  - Curious competitors vs. government-supported espionage

  Which assets might they be after? How skilled are they?

- Understand the system
  - Design and implementation
  - Configuration, patch level, ...
  - Organizational procedures, system usage, ...

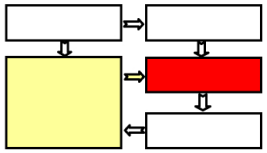
  Where might vulnerabilities arise?
  How might they be exploited, alone or in combination?

Dimensions to consider



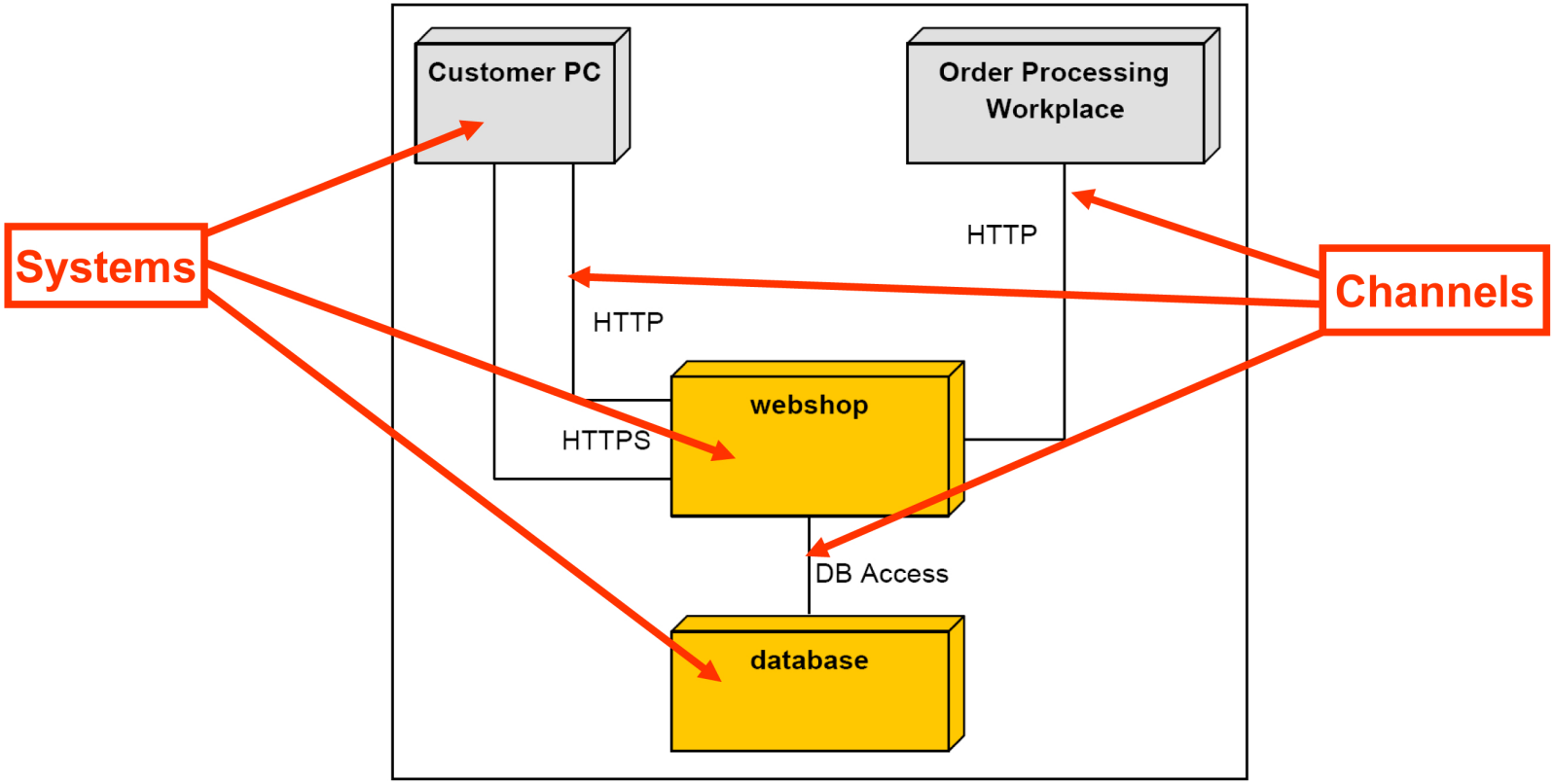
Too many possibilities to consider them all.
How do we find the hot spots?

Systems and Channels

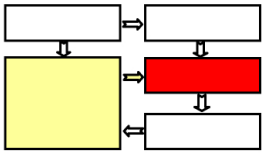


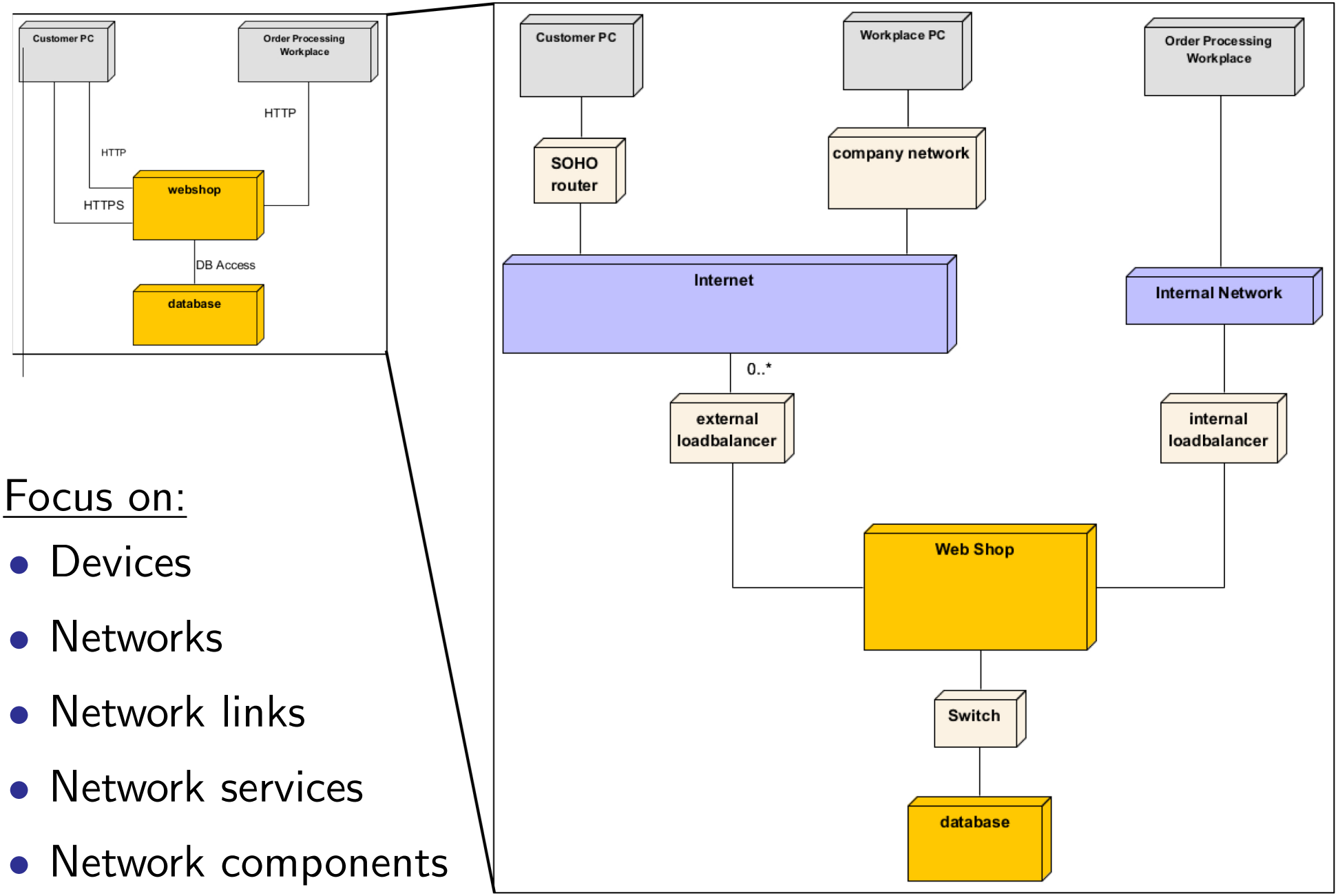
- What do we need to know? Systems and communication channels
- Where do we get this information from? **UML Deployment Diagrams**
- Abstraction level should be appropriate

  Example: logical system view (services, comm. protocols and endpoints)



Example (2): network view

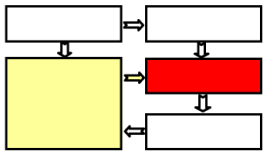


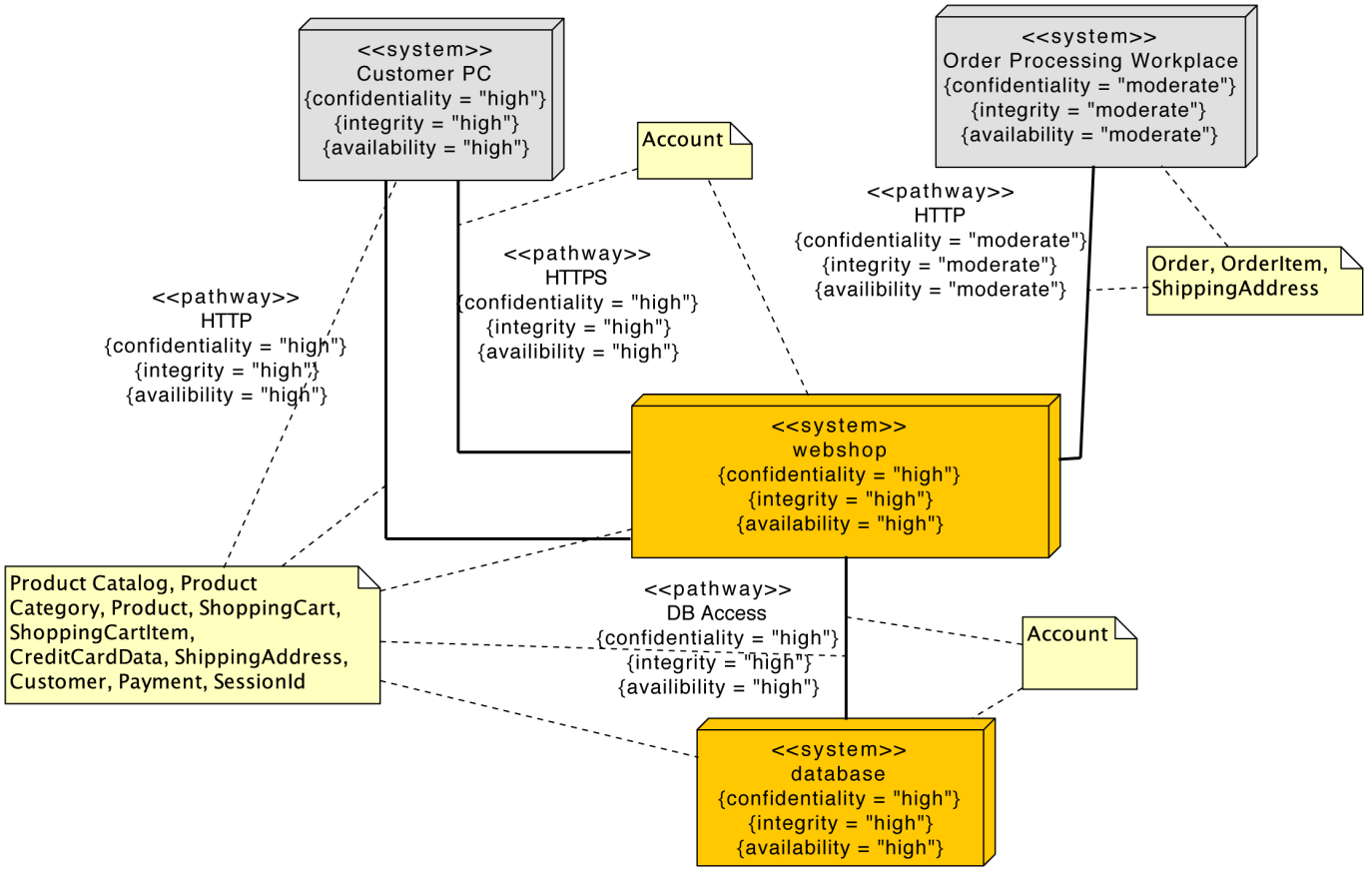


Focus on:

- Devices
- Networks
- Network links
- Network services
- Network components

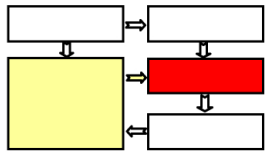
Incorporating assets: data pathways

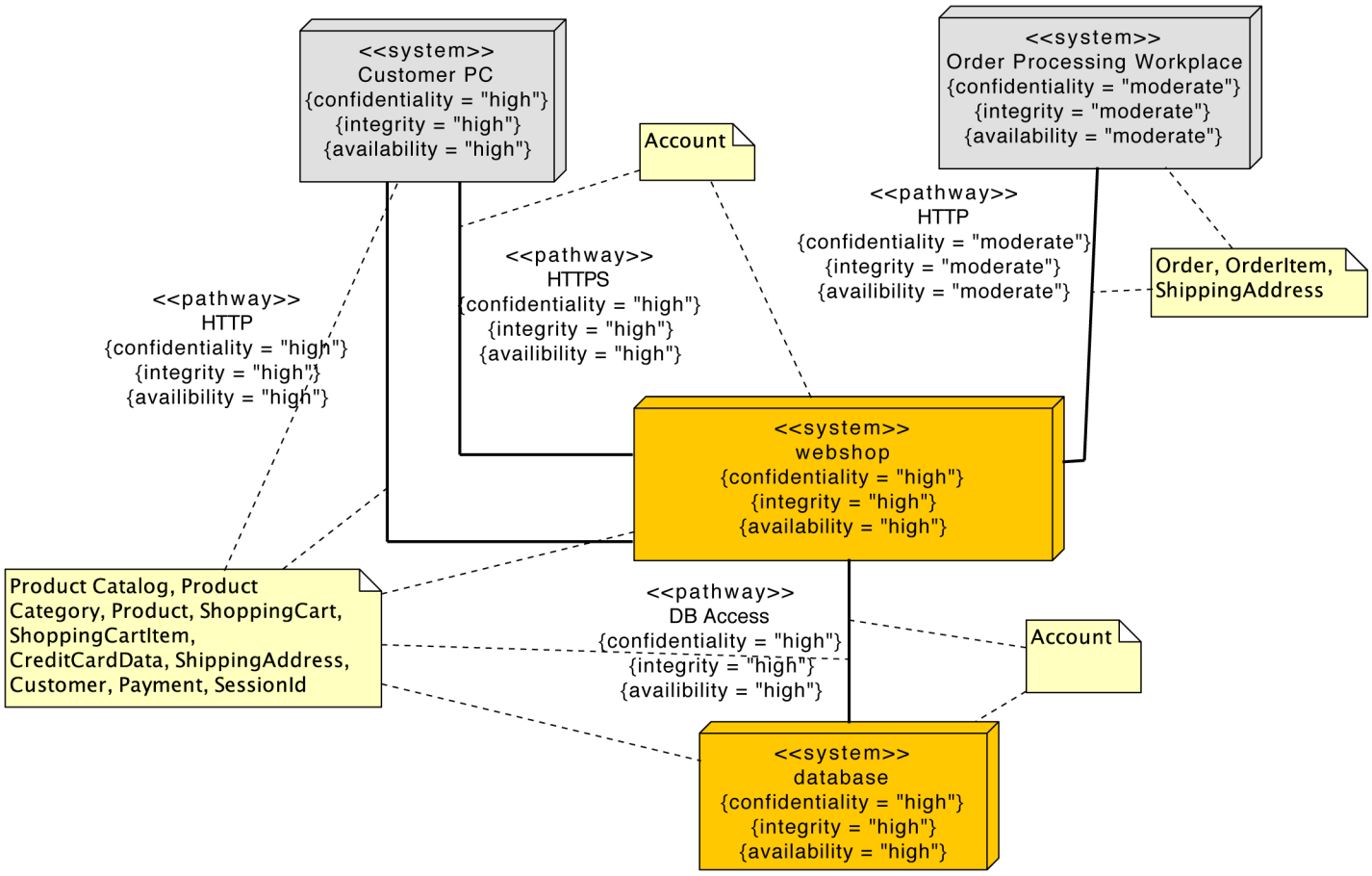




- **Visualize pathways data takes through the system**
  - Which pathways do data take through the system?
  - Which systems process and store what data?
  - What security requirements can be derived for channels and systems?
- Reveals most sensitive systems and channels and provides foundation for risk assessment (impact analysis) and focus on hot spots

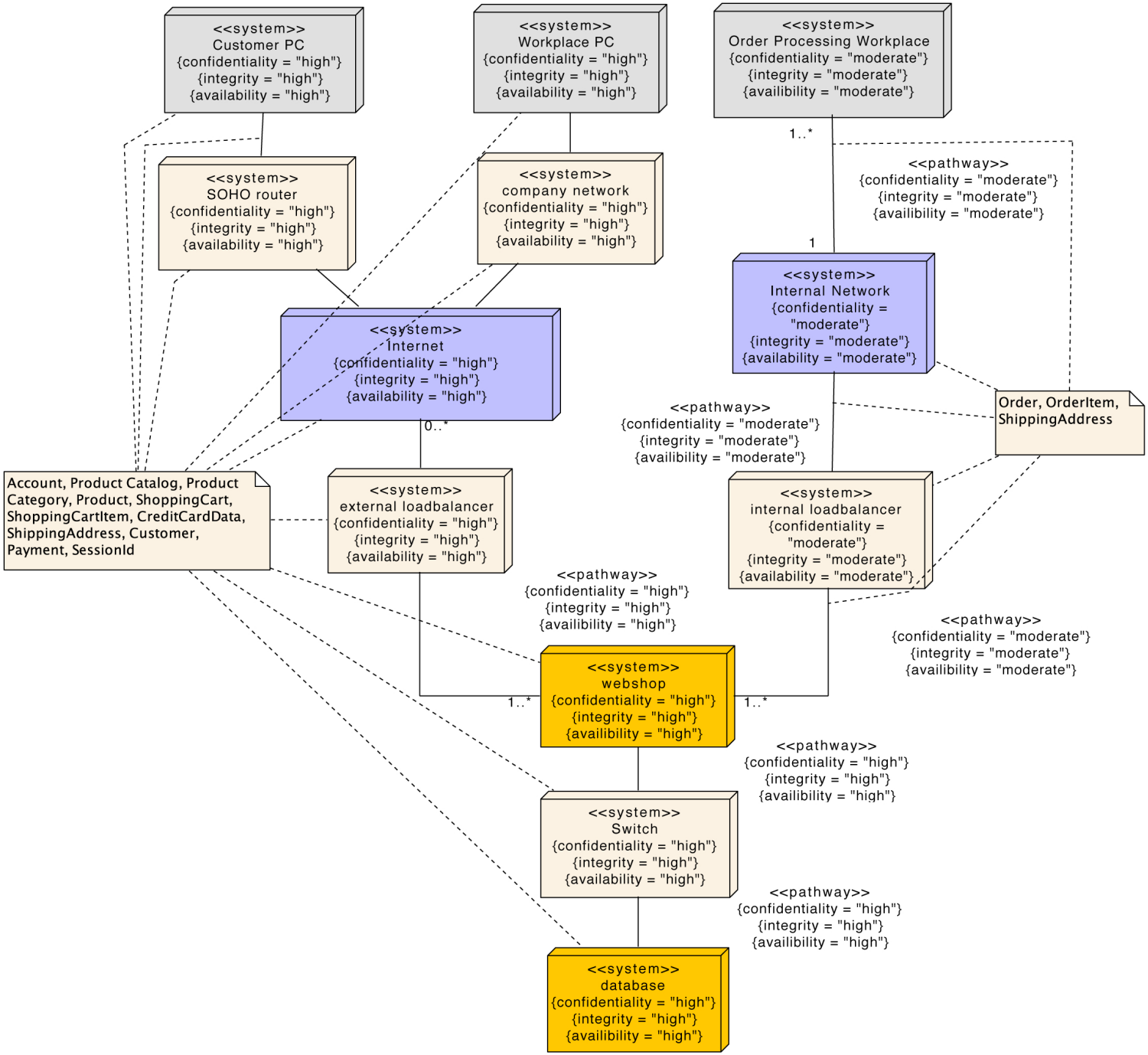
Data pathways: example & syntax



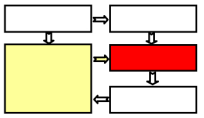


- Data types listed in annotation
- Stereotypes
  - ⟨⟨system⟩⟩ on nodes
  - ⟨⟨pathway⟩⟩ on associations
- Corresponding tagged values:
  - Confidentiality
  - Integrity
  - Availability

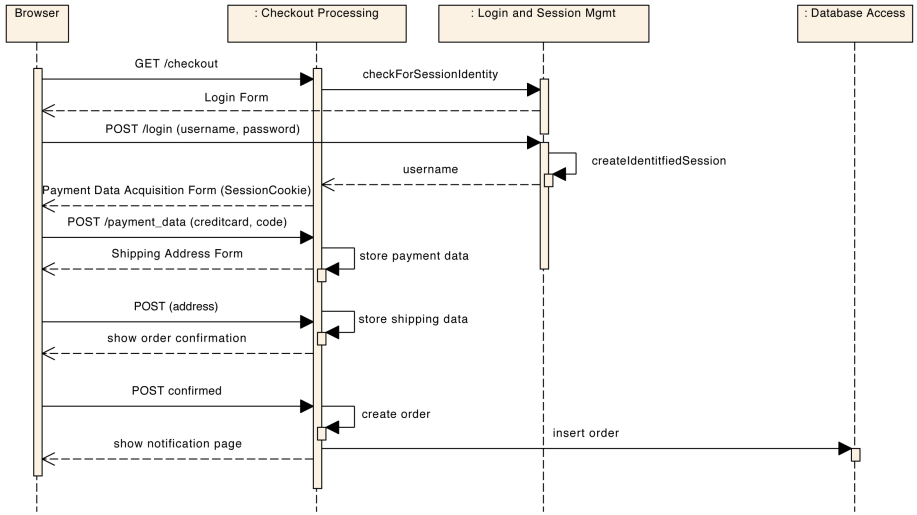
Data pathways: network view

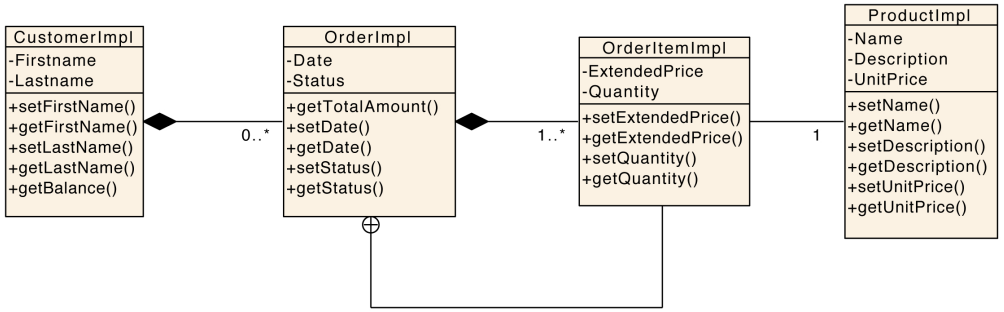


Constructing data pathways

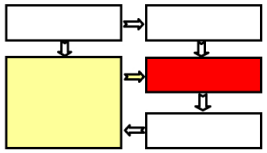


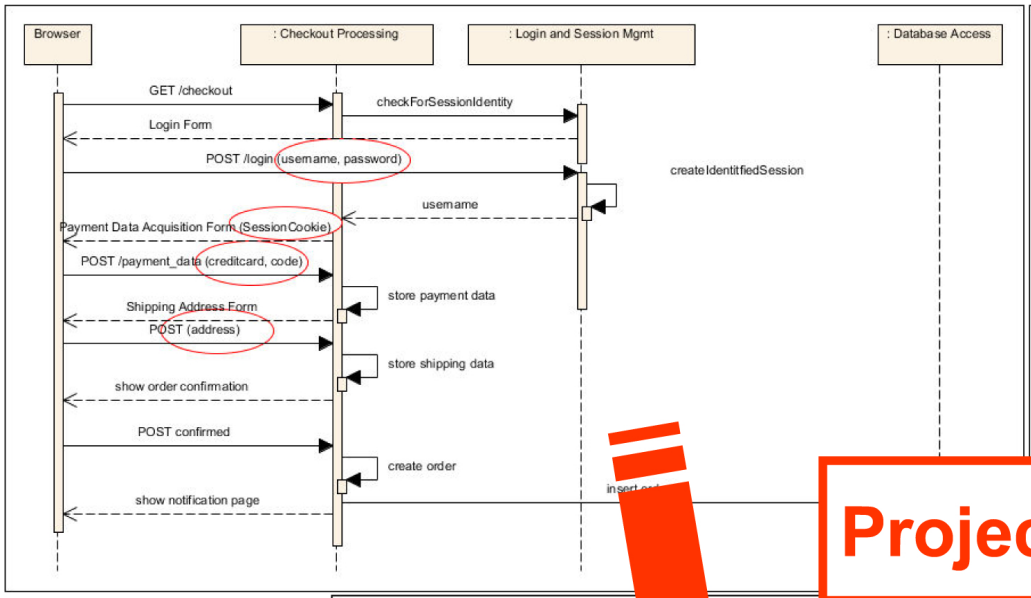
- Study of system documentation
- Interview of architects and developers
- Source code analysis
- Derivation from design models

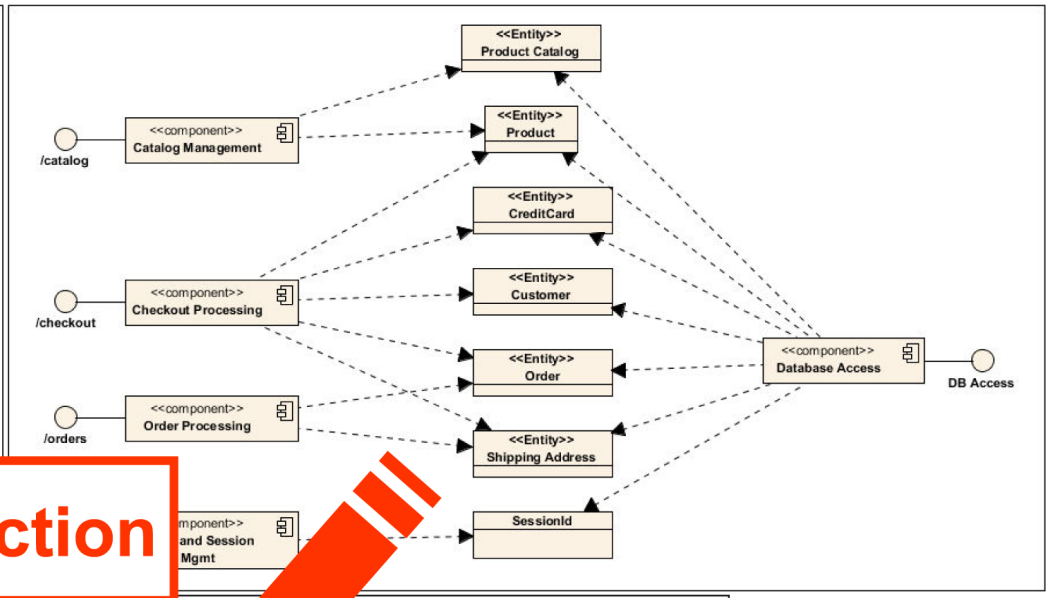




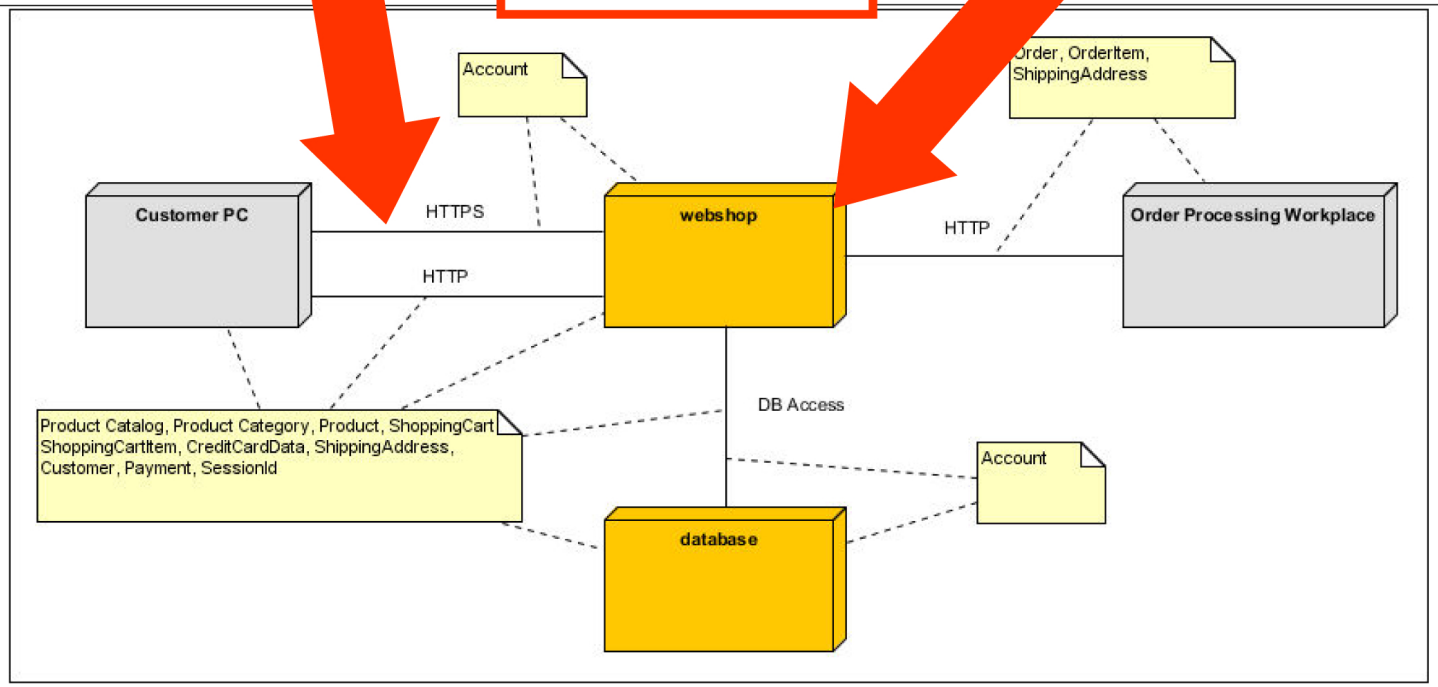
Deriving data pathways from design models



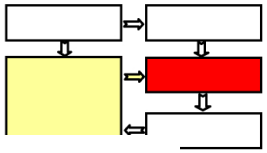




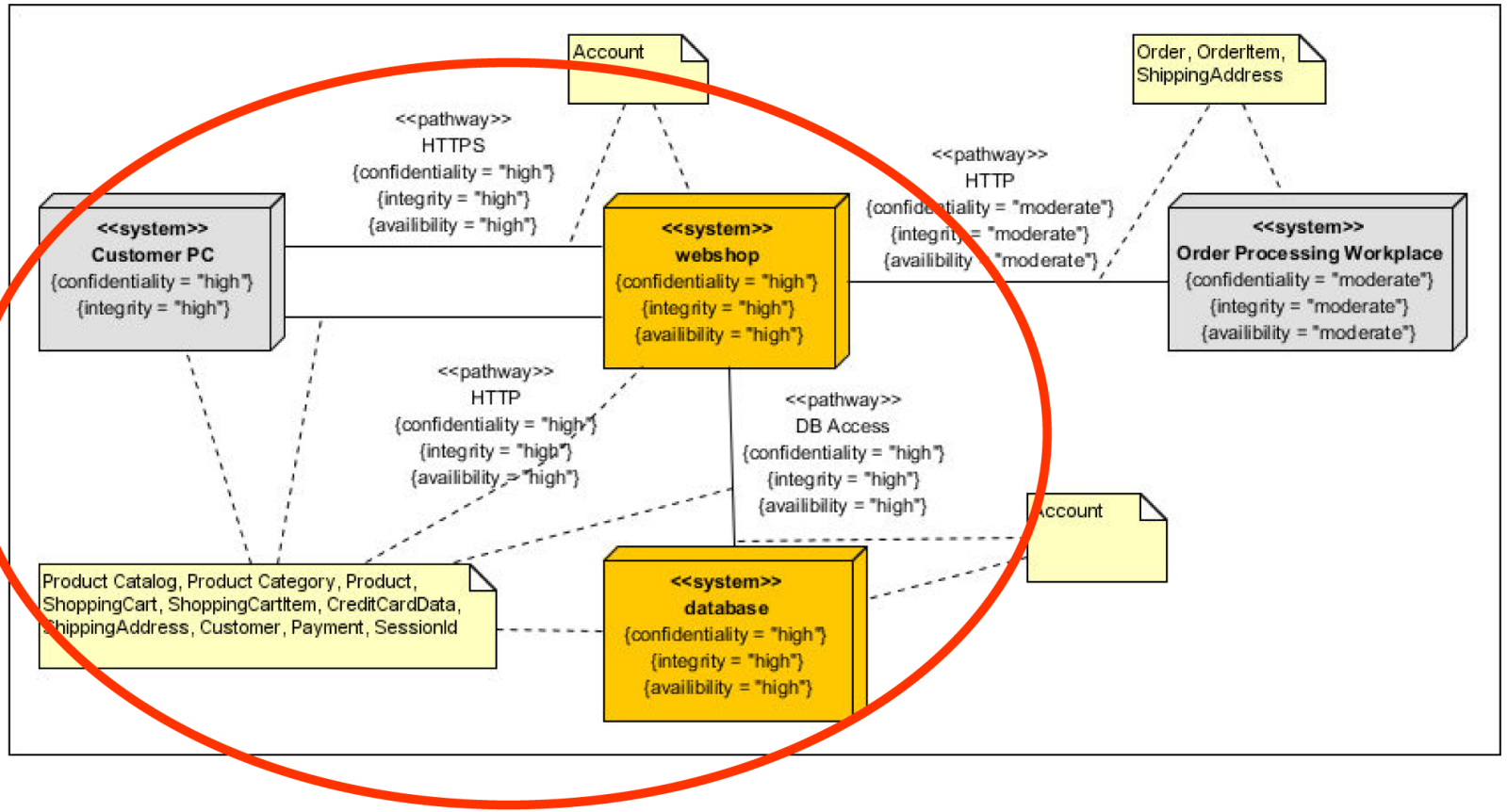
Projection



Data pathways: security categorization



Data Item	Confidentiality	Integrity	Availability
Catalog	N/A	Low	High
Category	N/A	Low	High
Product	N/A	Moderate	High
Item	N/A	Moderate	High
Person	Low	Low	High
OnlineAccount.Password	High	Moderate	High
OnlineAccount.A/R Balance	Moderate	High	High
OnlineAccount.default	Low	Low	High
ShoppingCart	Low	Moderate	High
CreditCard	High	Moderate	High
Payment	Low	High	Low
Order	Low	Moderate	Moderate
OrderedItem	Low	Moderate	Moderate
ItemPurchased	Low	Low	Low



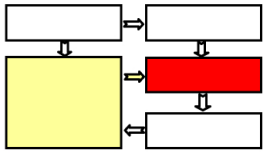
Road map

- Classical approaches to modeling faults
  - FMEA
  - Fault trees
- Risk analysis using design models

☞ **Attack modeling**

- Experience with techniques

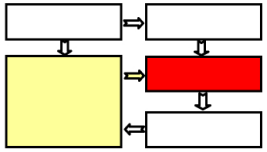
Identifying threats: attacks



- Use existing vulnerability catalogs and checklists from practice
  - BSI IT-Grundschutz-Kataloge/ Gefährdungskataloge
  - Open Web Application Security Project, Top Ten Project
  - Common Vulnerabilities and Exposures (`http://cwe.mitre.org`)
- Ask security specialists!
- Be creative

  Identifying threats requires creativity as many systems have unique requirements and therefore unique threats. One must look at each service and ask "If I were an attacker, how could I possibly exploit this security service?" Any answer constitutes a threat (OWASP)
- Consider misuse cases as a starting point

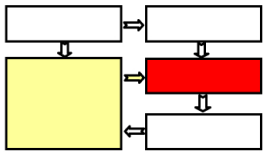
Tables



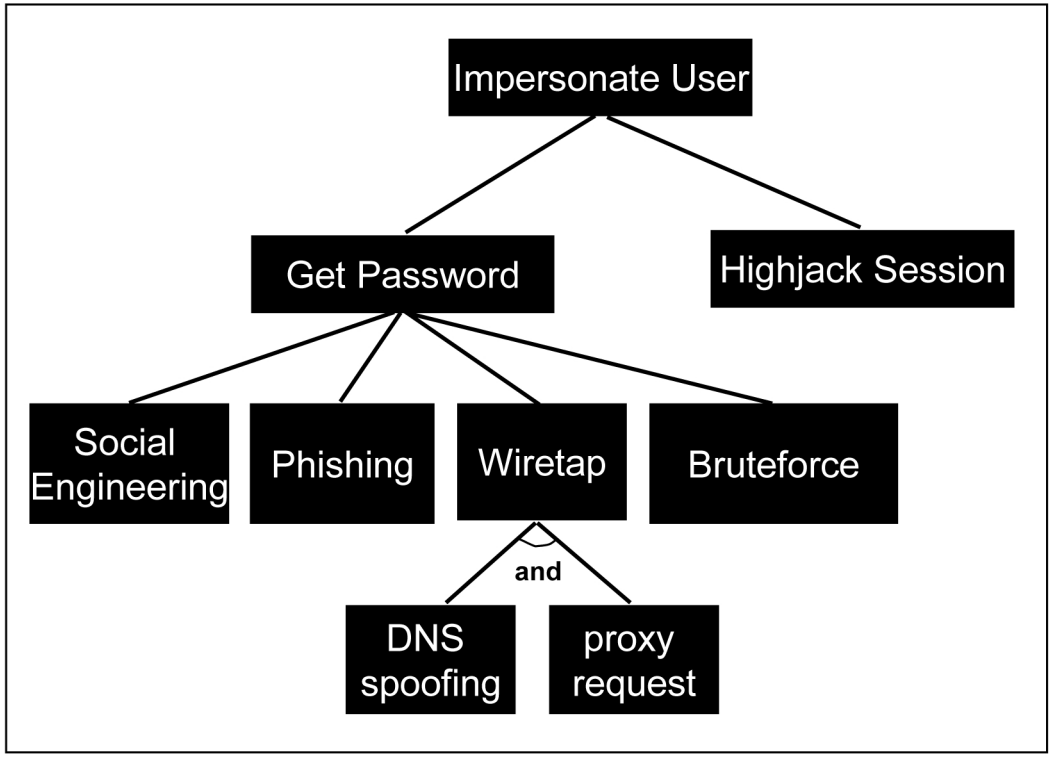
Target	Attacker	Attack	Impact Description	Countermeasure
shop (Internet)	External Attacker	Access w/ forged user id	Potentially loss of confidentiality, integrity, and availability of all accessible data (arbitrary user)	user login
shop (Internet)	External Attacker	Password guessing (Brute force)	see above (single user)	
shop (Internet)	External Attacker	wiretap password	see above (single user)	login via HTTPS
database	Internal Attacker	read user credentials	see above (arbitrary user)	
shop (Internet)	External Attacker	session guessing	Take over of an active session, potentially loss of confidentiality, integrity, and availability of a particular users data	
shop (Internet)	External Attacker	wiretap session token	see above	
shop (Internet)	External Attacker	wiretap user data on transport	Potentially loss of confidentiality, integrity, and availability of all accessible data (single user)	
database	Internal Attacker	read user data from database	Potentially loss of confidentiality, integrity, and availability of all accessible data (arbitrary user)	
shop (Internet)	Customer	Unauthorized Access	access to premium offers and other users data	user access control on login, filter links according to user authorization
shop (Internet)	Customer	URL guessing	see above	
shop (Order Processing)	Warehouse Clerk	Unauthorized Access	access to user data	

- Well suited for communicating threats
- Less helpful for identifying and assessing threats!

Attack trees

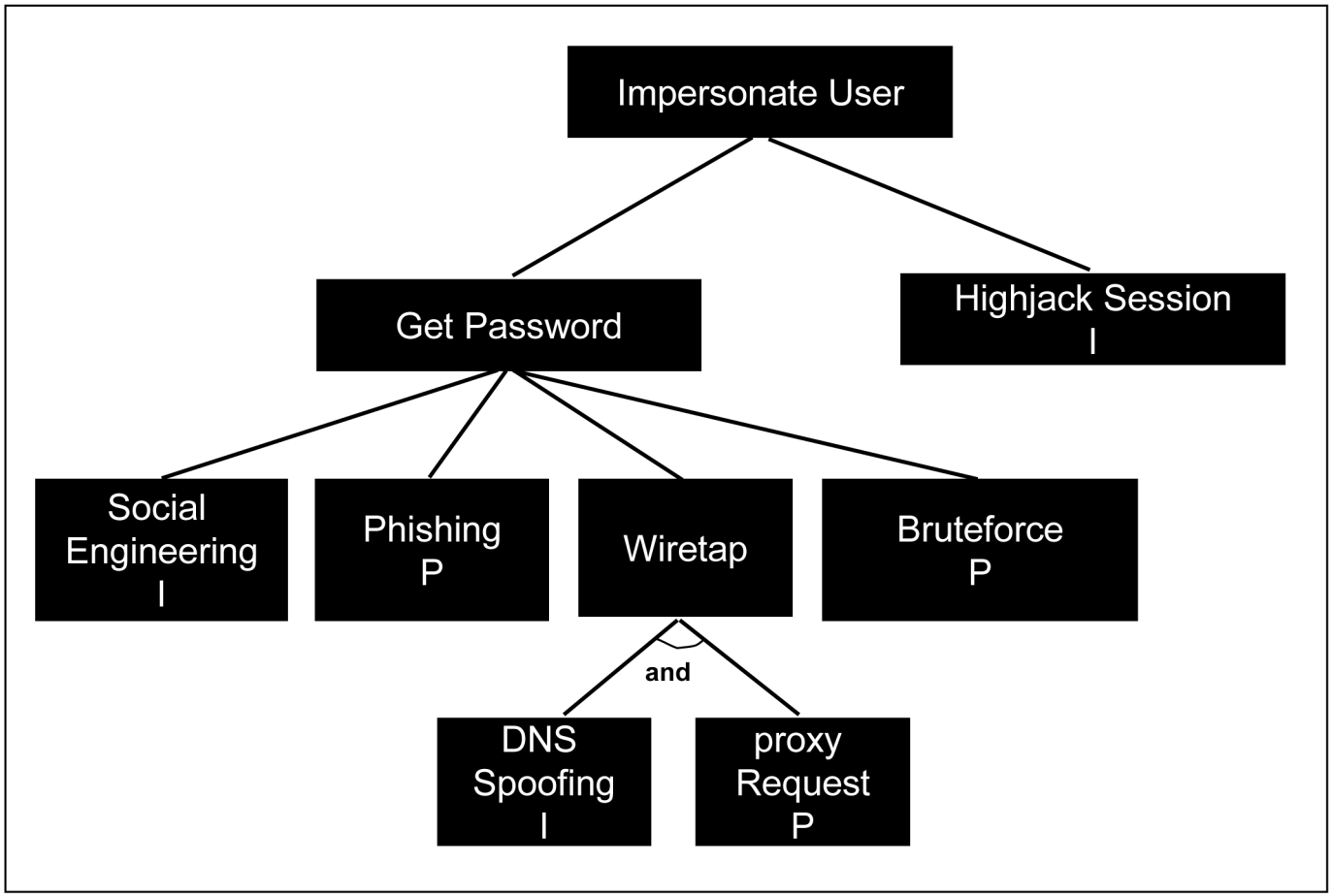


- Nodes are attacks (threats)
- Top-level goal may be obtained from misuse cases
- Refine as needed
  - Alternative attacks
  - Composite attacks (AND)
- Assign attributes to nodes
  - Probabilities
  - Estimated impact
  - Compute probabilities, impact, or risk of cut sets



- **Basically fault trees, plus special roles/motives of attackers**

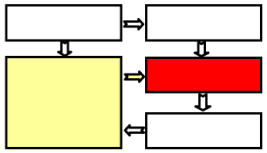
Attack tree with attributes

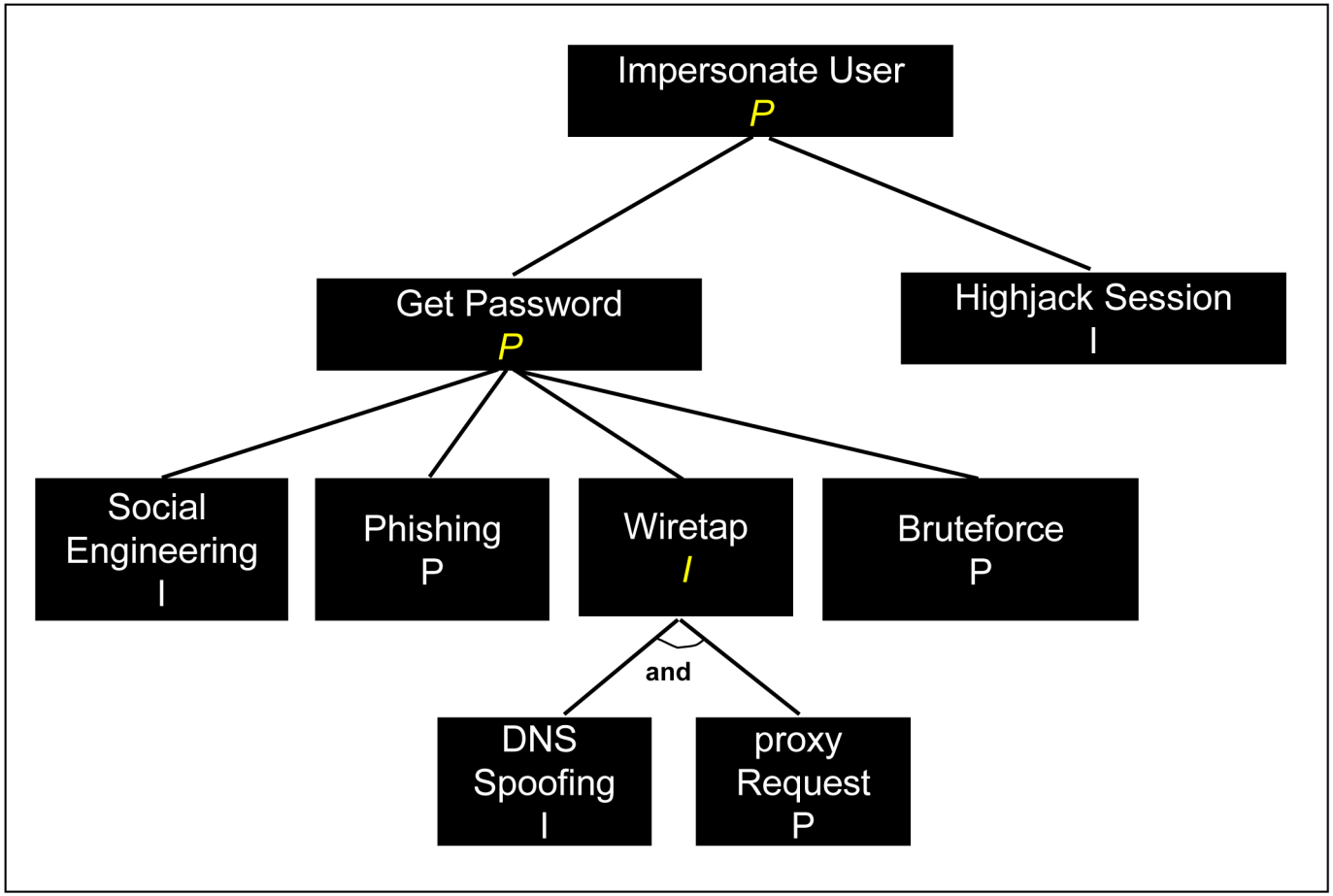


P = Possible

I = Impossible

Attack tree with computed attributes





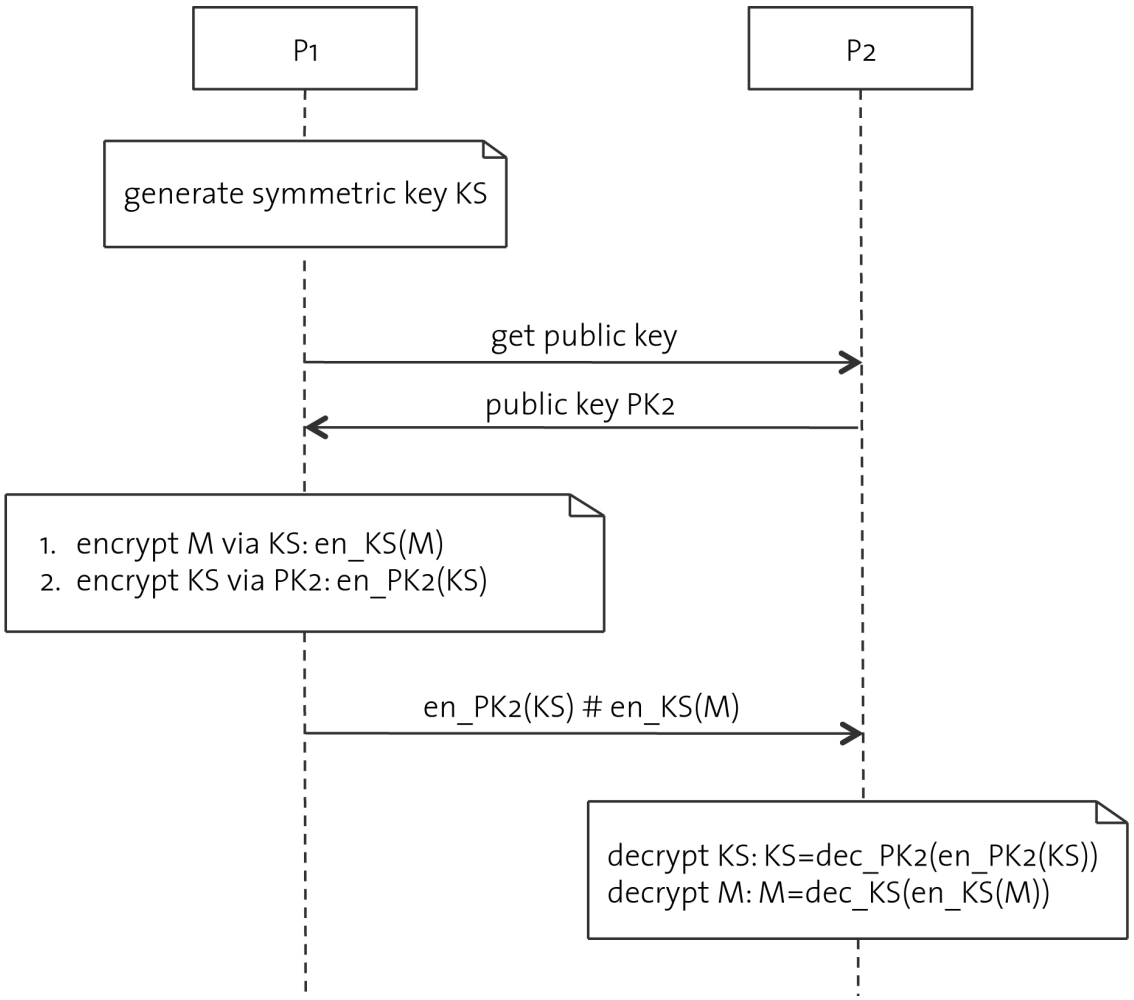
P = Possible

I = Impossible

Alternatives: propagate up probabilities, cost of attack, ...

Example: PGP

- (Get authentic public key)
- Encrypt message with a symmetric session key
- Encrypt symmetric key with partners public key
- Send both (symmetrically) encrypted message and (asymmetrically) encrypted symmetric key to partner
- Partner decrypts key and subsequently message



PGP attack tree (partial!)

Goal: Read a message encrypted with PGP

1. Decrypt the message itself
    - 1.1. Break asymmetric encryption
    - OR 1.2. Break symmetric-key encryption

OR 2. Determine symmetric key used to encrypt the message via other means
    - 2.1 Fool sender into encrypting message using public key whose private key is known
    - OR 2.2 Monitor the sender's computer memory
    - OR 2.3 Monitor the receiver's computer memory
    - OR 2.4 Determine key from pseudorandom generator
    - OR 2.5 Implant virus that exposes the symmetric key

OR 3. Get recipient to (help) decrypt message
    - 3.1 Chosen ciphertext attack on symmetric key
    - OR 3.2 Chosen ciphertext attack on public key
    - OR 3.3 Read message after it has been decrypted

OR 4. Obtain private key of recipient
    - 4.1 Factor RSA modulus
    - OR 4.2 Monitor recipient's memory
    - OR 4.3 Implant virus to expose private key
    - OR 4.4 Generate insecure public/private key pair for recipient

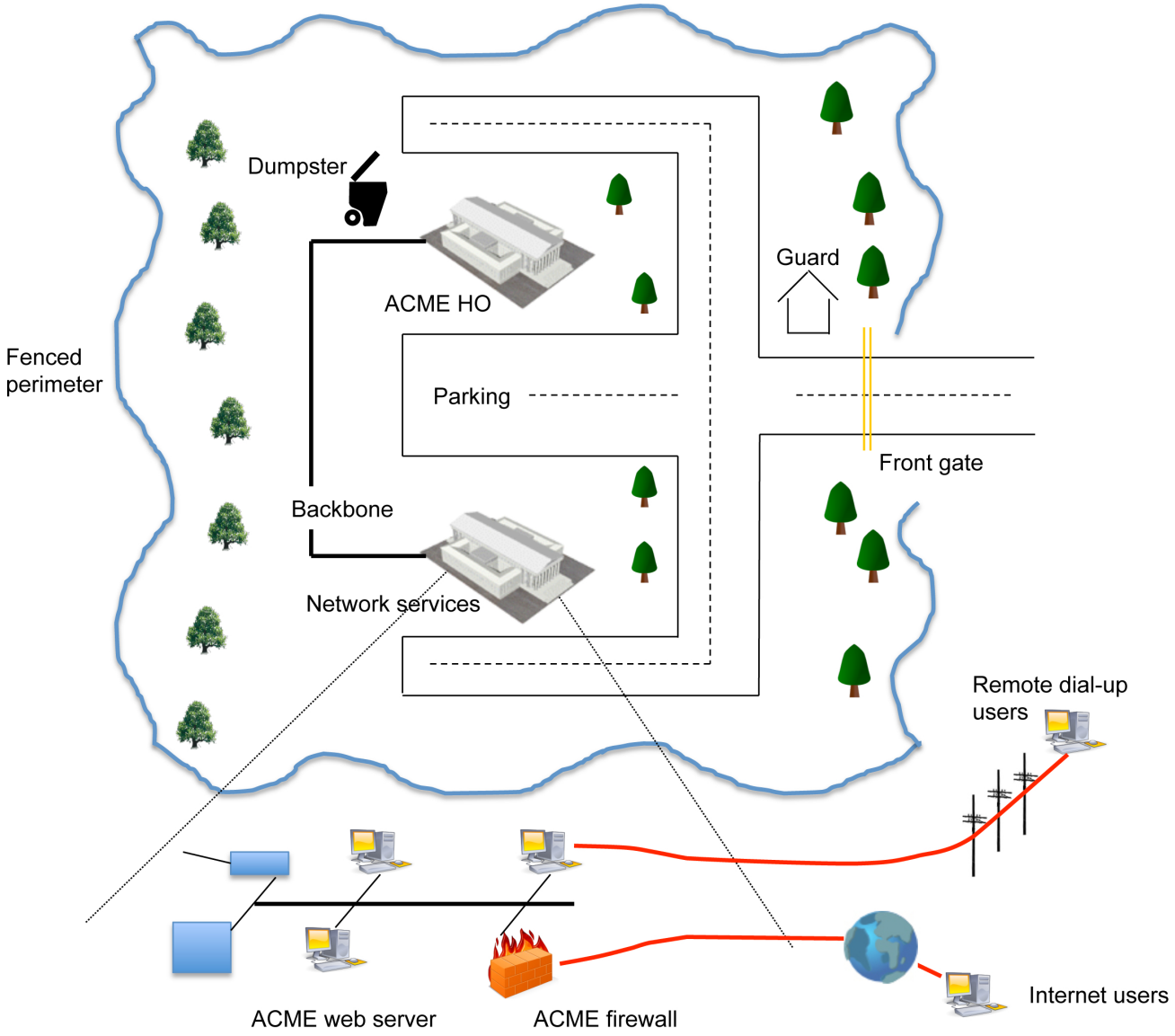
What is the point?

- "Use 1024 vs. 2048-bit key" is but one problem
- Breaking crypto is just one possibility
  - Capture screen using a Trojan
  - Use a key-logger (sniffer)
- Different attack trees for different attackers
- Reuse in larger applications
- As with FTAs and FMEAs:

  Key is to think about possible problems!

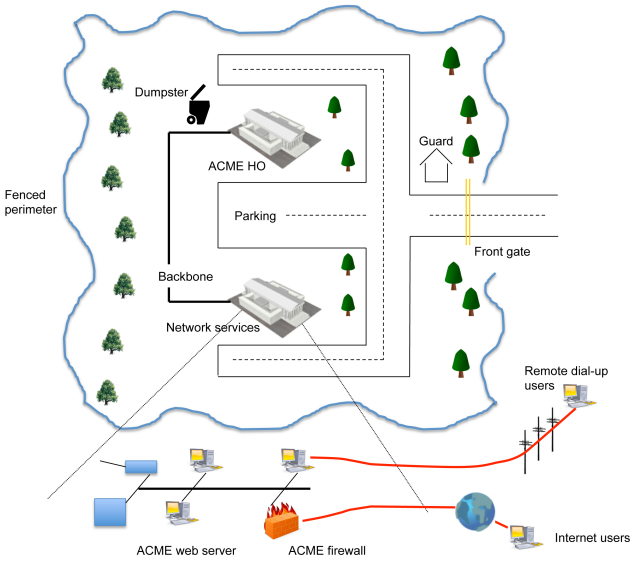
Attack trees

Use structure of system and environment



How would you attack Acme's computers?

Attack tree



Disclosure of ACME proprietary secrets (partial)

OR 1. Physically scavenge discarded items from ACME
- OR 1. Inspect dumpster content on-site
- 2. Inspect refuse after removal from site

2. Monitor emanations from ACME machines
- AND 1. Survey physical perimeter to determine best monitoring location
- 2. Acquire necessary monitoring equipment
- 3. Monitor emanations from site

3. Recruit help of an ACME insider
- OR 1. Plant spy as insider
- 2. Use existing insider

4. Physically access ACME networks or machines
- OR 1. Get physical, on-site access to Intranet
- 2. Get physical access to external ACME machines

5. Attack ACME intranet using its Internet connections
- OR 1. Monitor communications over Internet for leakage
- 2. Plant a process to send attacker sensitive information over Internet
- 3. Gain privileged access to ACME web server

Attack tree: refinement

5.3 Gain privileged access to ACME web server

AND 1. Identify ACME domain name

2. Identify ACME firewall IP address

OR 1. Interrogate domain name server

2. Scan for firewall identification

3. Trace route through firewall to web server

3. Determine ACME firewall access control

OR 1. Search for specific default listening ports

2. SCAN ports broadly for any listening port

4. Identify ACME web server operating system and type

OR 1. Scan OS services' banners for OS identification

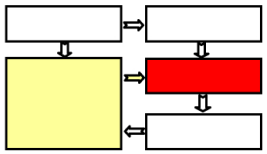
2. Probe TCP/IP stack for OS characteristic information

5. Exploit ACME web server vulnerabilities

OR 1. Access sensitive shared intranet resources directly

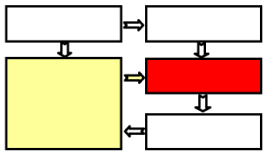
2. Access sensitive data from privileged account on web server

Attack trees in UML models

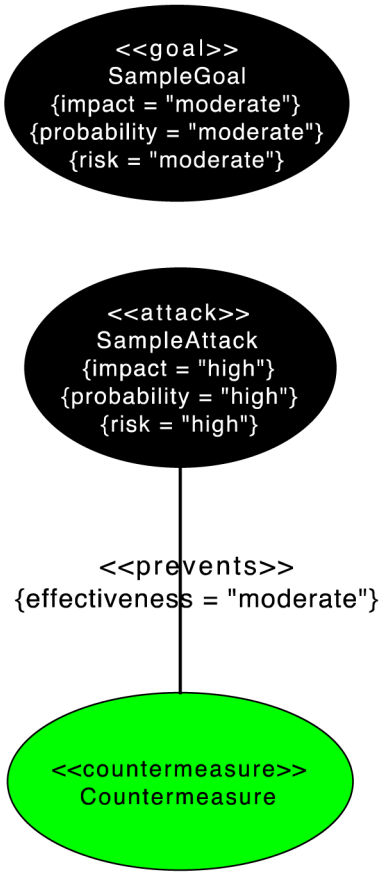


- Why UML?
  - Tool support
  - Integration with system requirements and design models
- Extension to attack trees
  - Include countermeasures giving a direct link to design model
- How?
  - Use case diagrams
  - UML profile
  - Trees can be decomposed in multiple diagrams
  - One diagram per goal

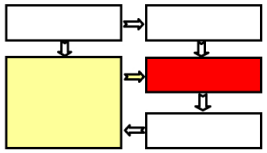
UML profile for extended attack trees

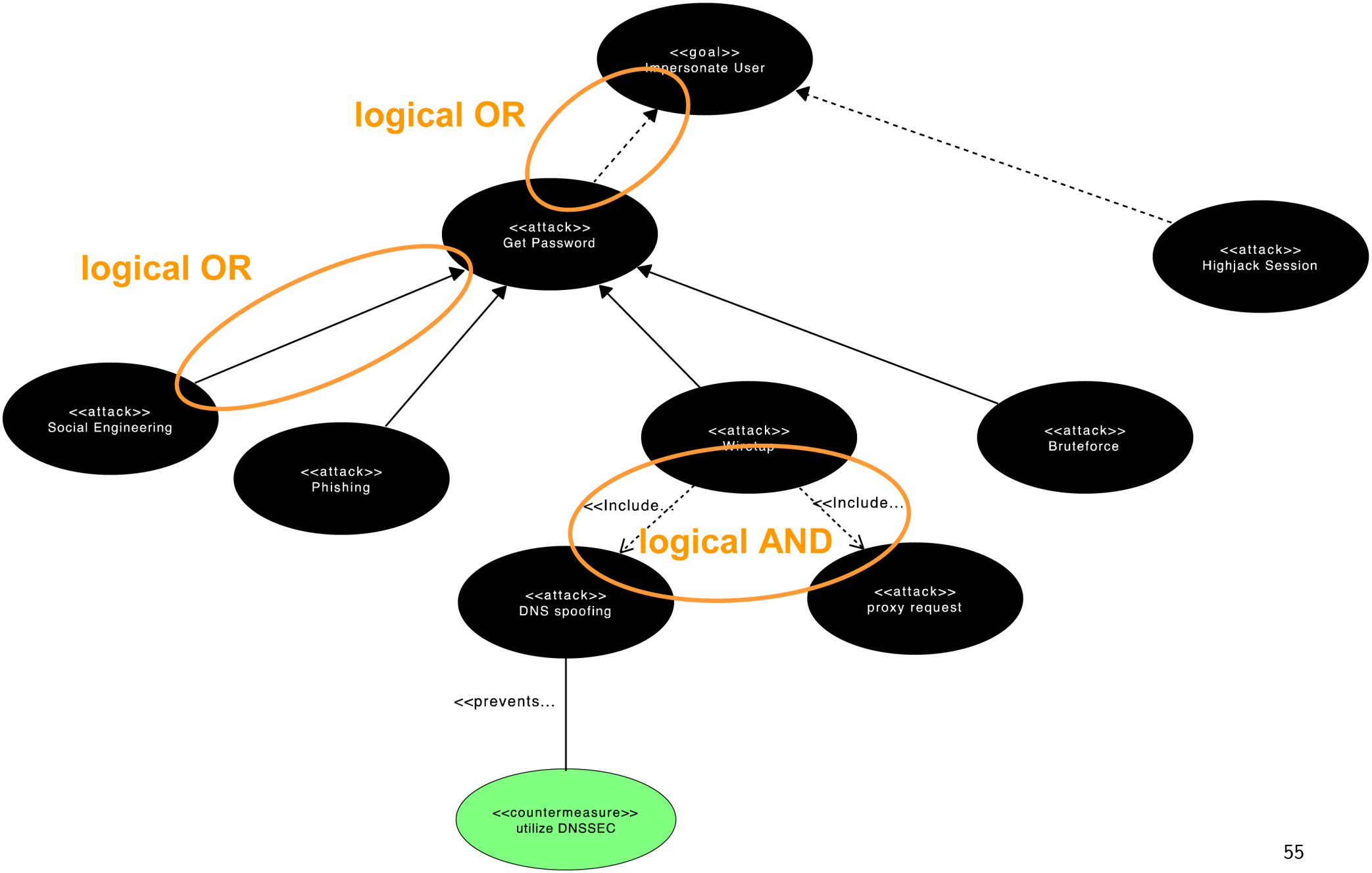


- Based on use case diagrams (and possibly color codes)
- Use Case stereotypes
  - ⟨⟨**goal**⟩⟩ with tagged values: **impact**, **probability**, **risk**
  - ⟨⟨**attack**⟩⟩ with tagged values: **impact**, **probability**, **risk**
  - ⟨⟨**countermeasure**⟩⟩
- Dependency stereotype
  - ⟨⟨**prevents**⟩⟩ with tagged value **effectiveness**
    Expresses relationship between attack and countermeasure
- Relations
  - **Realization:** Attacks realize goals (logical OR)
  - **Specialization:** Attacks specialize other attacks (logical OR)
  - **Include:** Attacks are composed of other attacks (logical AND)
- Qualitative assessment with three levels: low, moderate, high

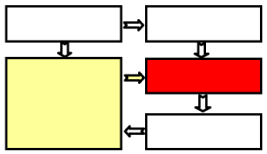


UML attack trees: example





UML attack trees: risk analysis

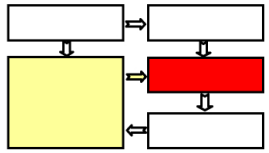


- Attack probability takes into account countermeasure effectiveness

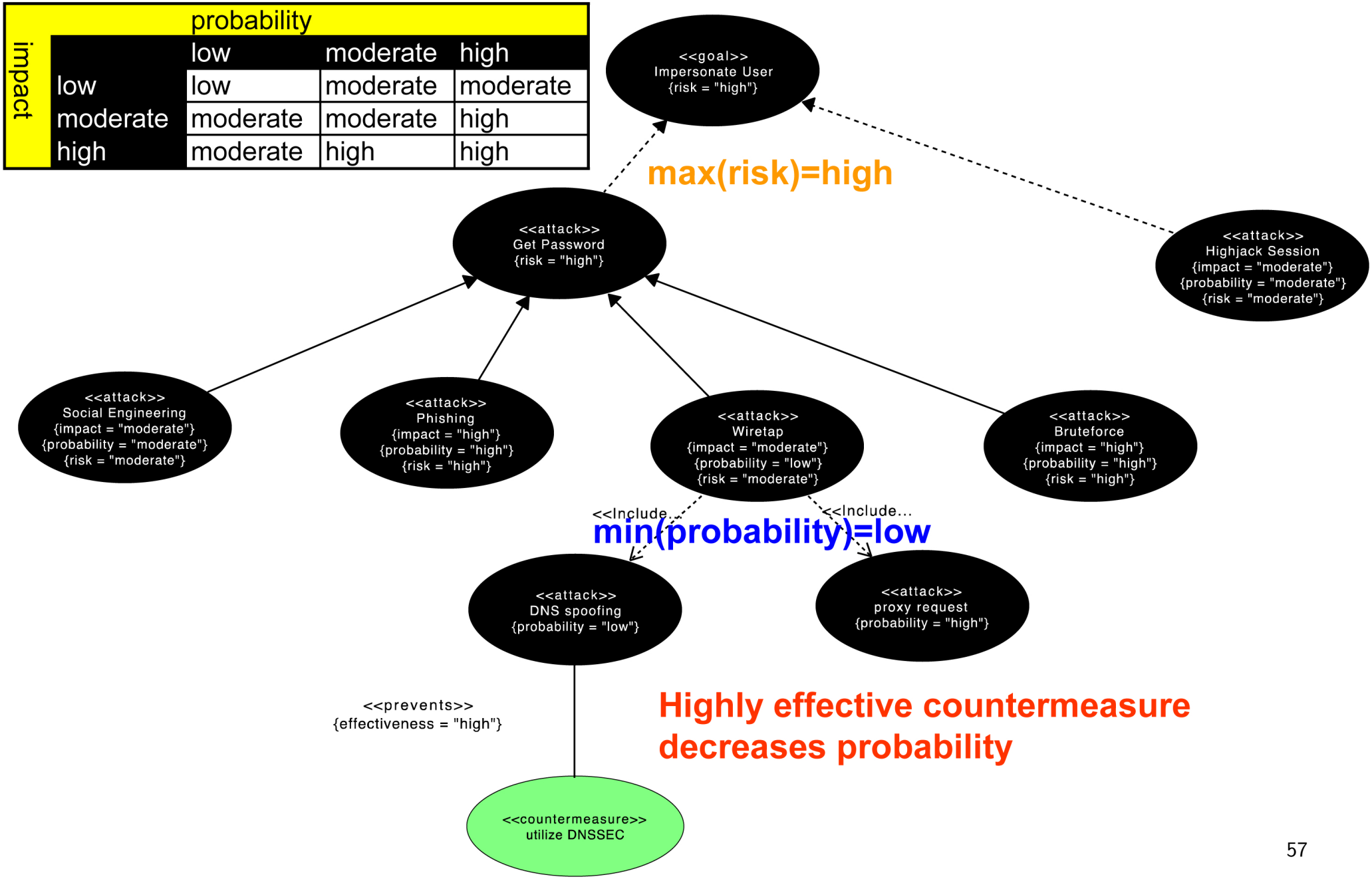
impact \ probability	low	moderate	high
low	low	moderate	moderate
moderate	moderate	moderate	high
high	moderate	high	high

- Combine probabilities and risks. E.g.,
  - $\wedge$-composed attacks: probability = minimum of subnodes
  - Goals & high-level attacks ($\vee$): risk only = maximum of subnodes
- Above qualitative risk oversimplifies computations!
  Still useful for starting point for analysis

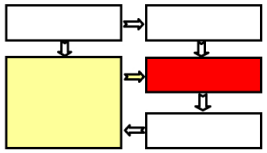
Risk analysis: example

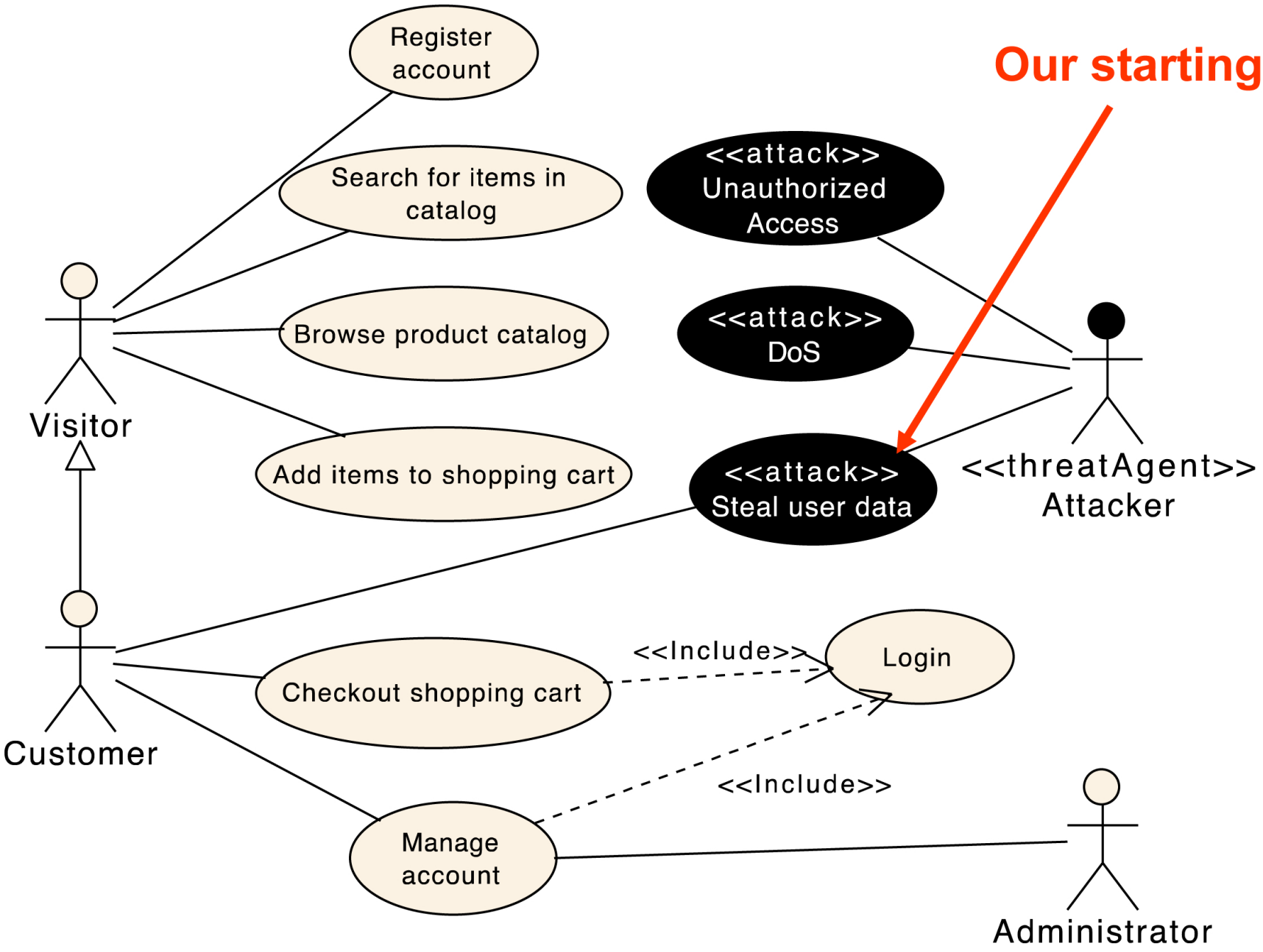


impact \ probability	low	moderate	high
low	low	moderate	moderate
moderate	moderate	moderate	high
high	moderate	high	high

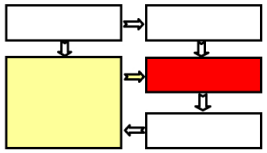


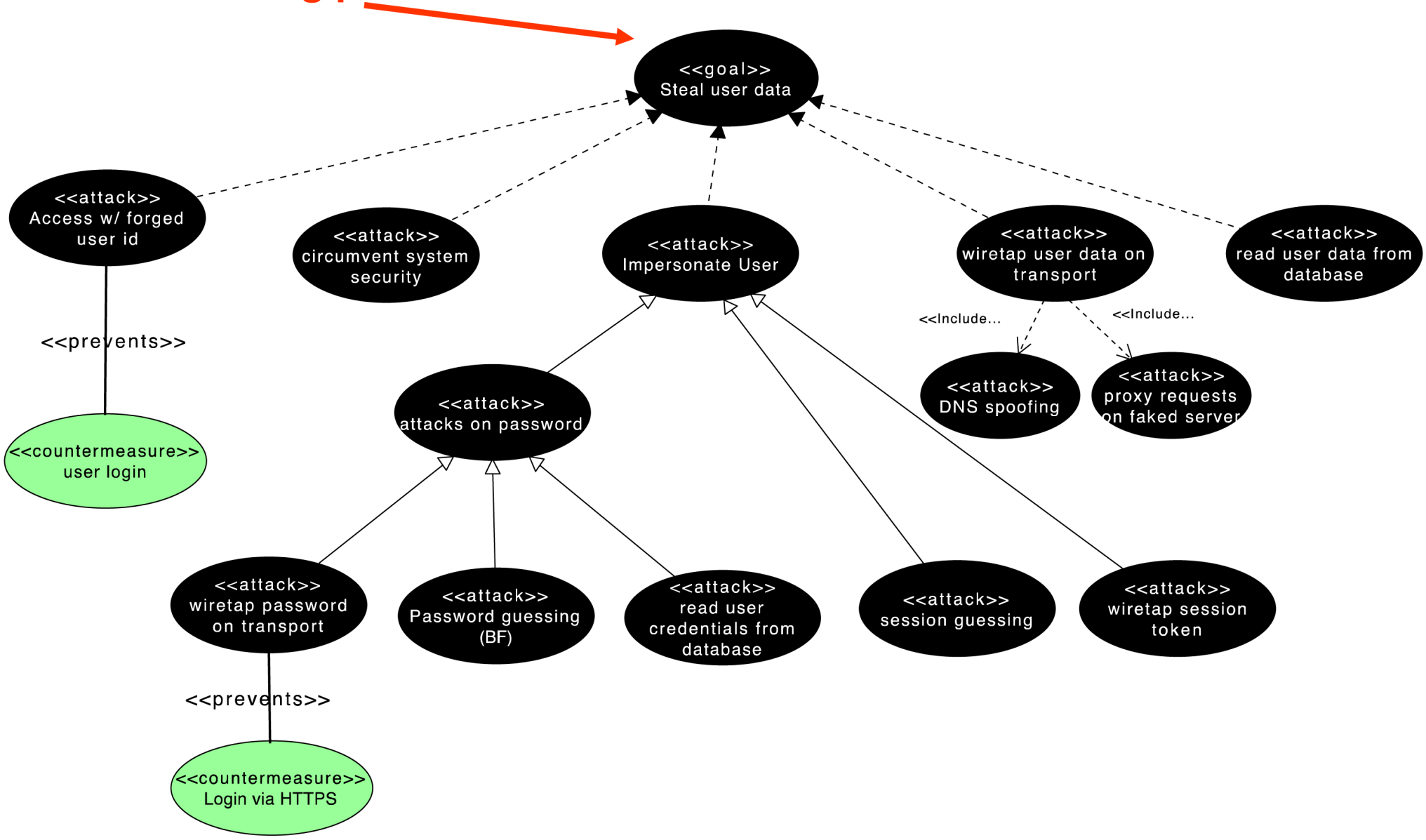
Capturing threats: example



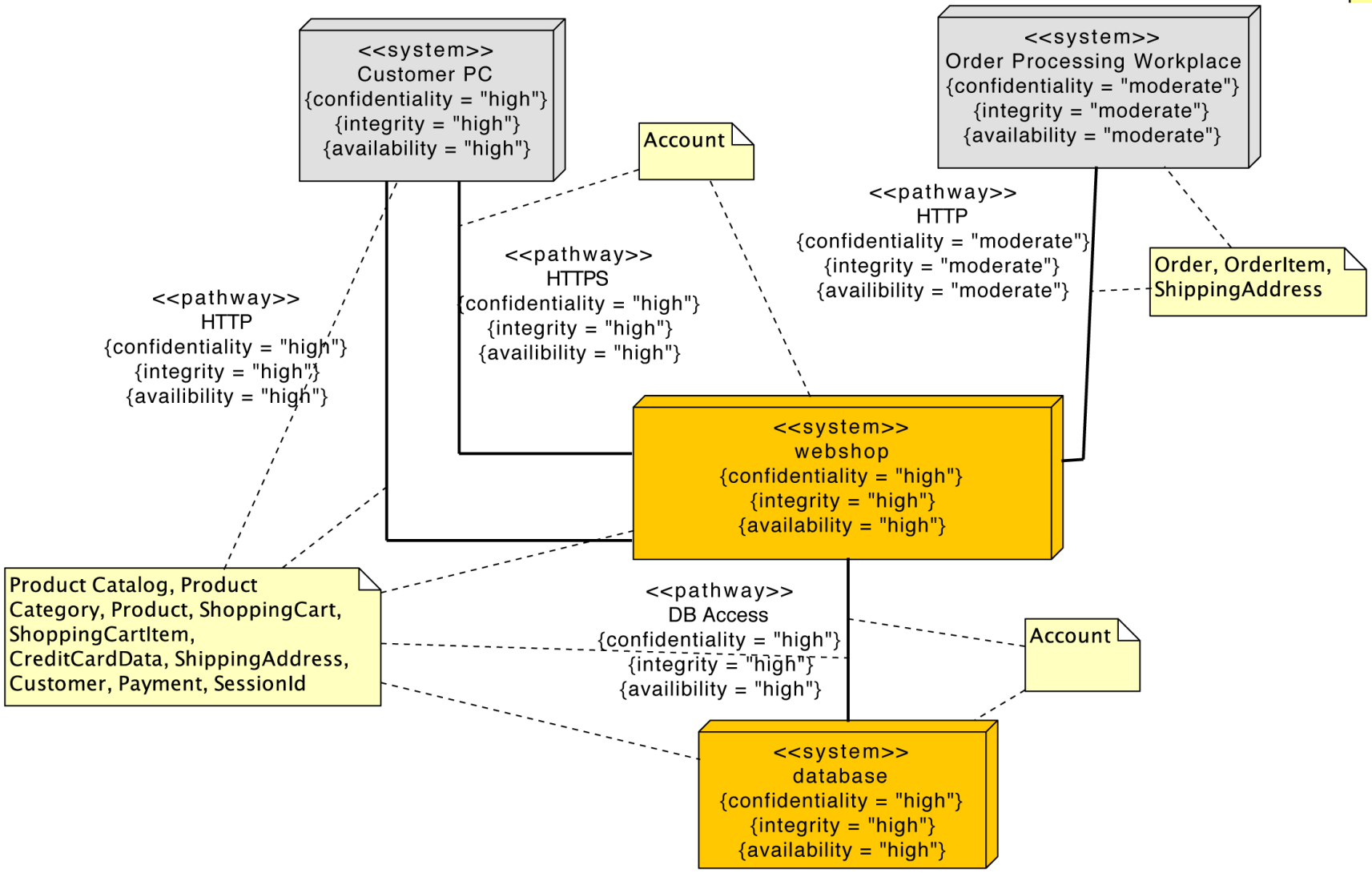


Capturing threats: example (II)



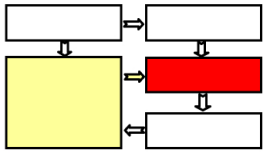


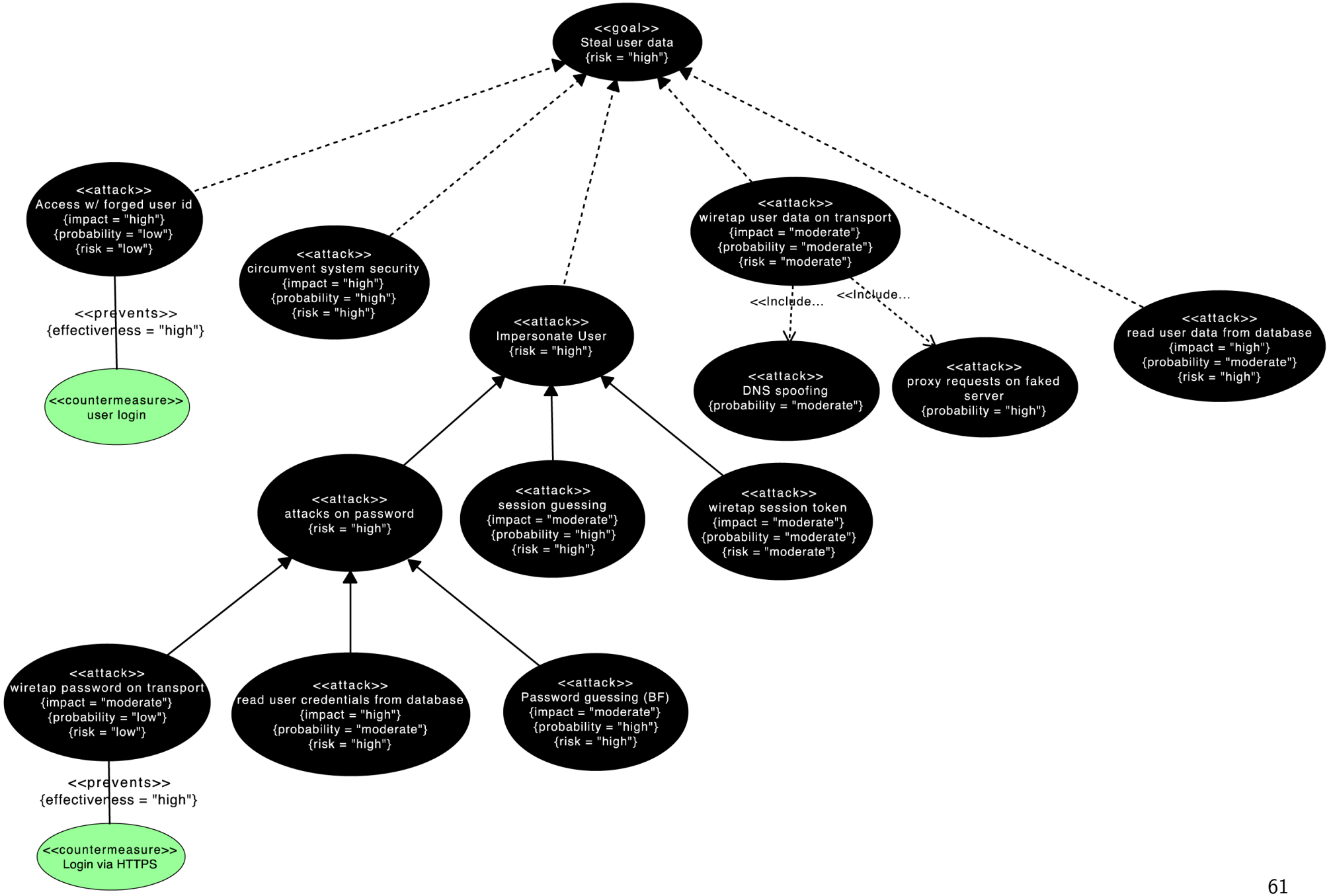
Assessing threat risks



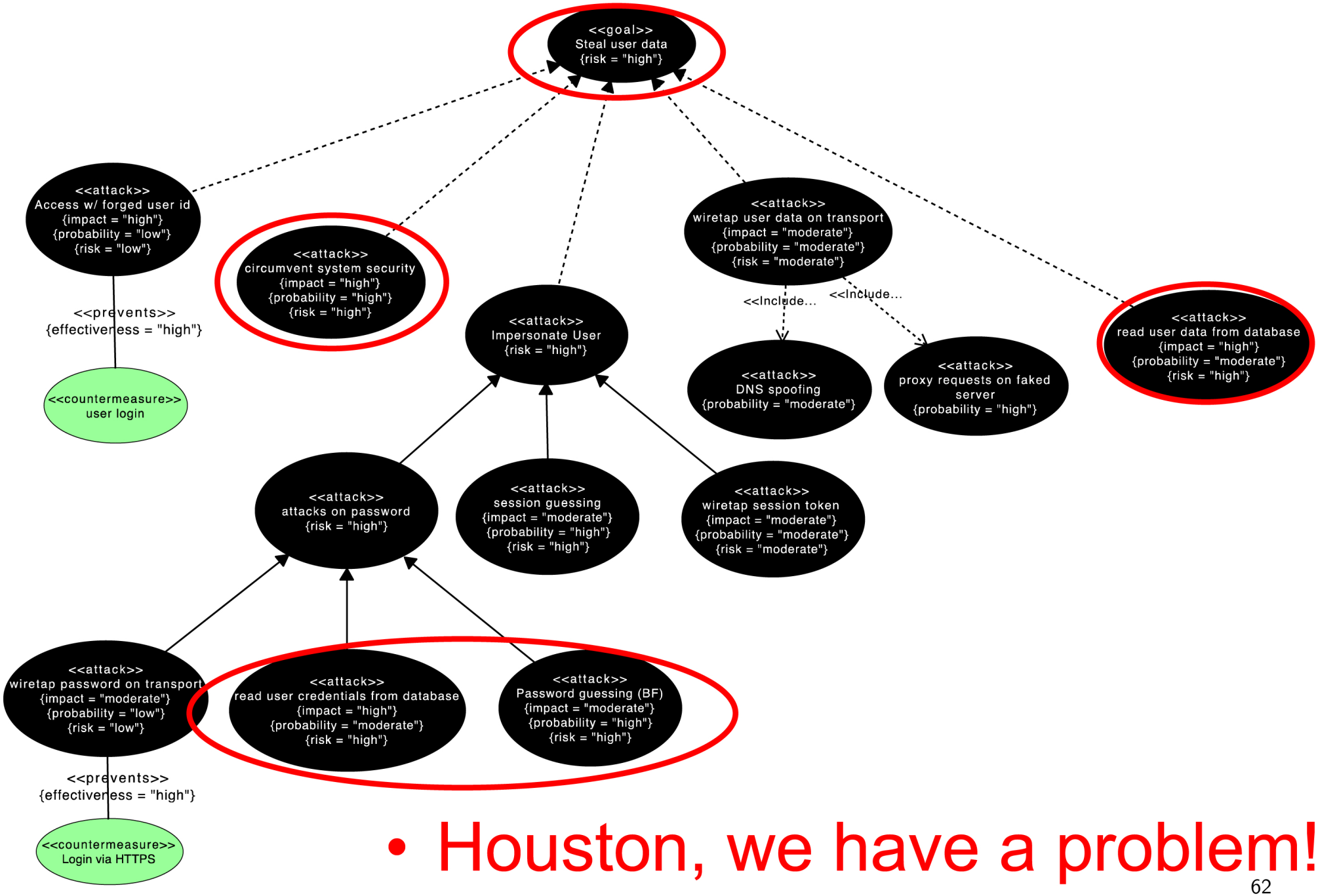
- Probability of attacks are based on expert evaluation
- Impact assessed based on pathways

Assessing threat risks: example





Assessing threat risks: example



- Houston, we have a problem!

Road map

- Classical approaches to modeling faults
  - FMEA
  - Fault trees
- Risk analysis using design models
- Attack modeling

☞ **Experience with techniques**

Practical experience

- Presented methods have been applied in real-world projects
  - Passenger information system, NYC subway
  - Security design of identity management systems at Deutsche Telekom
  - Security threat analysis of OAuth 2.0 protocol (IETF, RFC 6819)
- Data pathways
  - Make critical system components transparent to all involved parties (engineering, management, system administration)
  - Support comprehensive security reviews instead of selective discussions of apparently open issues (we need to prevent wiretapping)
- Attack trees
  - Excellent experiences for brainstorming about threats and countermeasures
  - Risk assessment sometime shows the critical points where no one had expected them (what about backup?)
  - Tend to get complex quickly $\Rightarrow$ decompose!
  - Combination of diagrams (overview) and HTML (description and hyperlinks)

Capturing threats: (hyper)text

- Threat and countermeasures described in different sections
- Links from threats to respective countermeasures and vice versa
- Table of contents represents attack tree

Source: OAuth 2.0 Threat Model and Security Considerations, http://tools.ietf.org/html/draft-lodderstedt-oauth-security/

Threat description

TOC

4.4.1.1. Threat: Malicious client obtains authorization

A malicious client could counterfeit a valid client and obtain an access authorization that way. The malicious client could even utilize screen scraping techniques in order to simulate the user consent in the authorization flow.

Countermeasures:

- The authorization server should authentication the client, if possible (see **Section 5.2.3.4**). Note: the authentication takes place after the end-user has authorized the access.
- The authorization server should validate the client's redirect_uri against the pre-registered redirect_uri, if one exists (see **Section 5.2.3.5**). Note: The validation of the redirect_uri is the only technical mean to recognize a malicious client id in advance of the authorization process. Further note this does not work for native applications because in contrast to web applications this uri is not bound to a single communication endpoint. The valid client's redirect_uri (typically with custom scheme) can be used by a malicious on any device.
- After authenticating the end-user, the authorization server should ask him/her for consent. In this context, the user shall be explained the purpose, scope, and duration of the authorization. Moreover, the authorization server must view to the end-user the meta data it associates with the particular client. It is up to the user to validate this data and approve the authorization request. (see **Section 5.2.4.3**).
- The authorization server must not perform automatic re-authorizations for clients it is unable to reliably authenticate or validate (see **Section 5.2.4.1**).
- If the authorization server automatically authenticates the end-user, it may nevertheless require some user input in order to prevent screen scraping. Examples are CAPTCHAs or user-specific secret like PIN codes.
- The authorization server may also limit the scope of tokens it issues to clients it cannot reliably authenticate (see **Section 5.1.5.1**).

Source: OAuth 2.0 Threat Model and Security Considerations, http://tools.ietf.org/html/draft-lodderstedt-oauth-security/

Countermeasure description

TOC

5.2.3.5. Validation of pre-registered redirect_uri

An authorization server may require clients to register their redirect_uri or a pattern (TBD: make definition more precise) thereof. The way this registration is performed is out of scope of this document. Every actual redirect_uri sent with the respective client_id to the end-user authorization endpoint must comply with that pattern. Otherwise the authorization server must assume the inbound GET request has been sent by an attacker and refuse it. Note: the authorization server MUST NOT redirect the user agent back to the redirect_uri of such an authorization request.

- Session fixation: allows to detect session fixation attempts already after first redirect to end-user authorization endpoint (**Section 4.4.1.7**).
- For clients with validated properties, this measure also helps to detect malicious apps early in the end-user authorization process. This reduces the need for a interactive validation by the user (**Section 4.4.1.1**, **Section 4.4.2.3**).

The underlying assumption of this measure is that an attacker must use another redirect_uri in order to get access to the authorization code. Deployments might consider the possibility of an attacker using spoofing attacks to a victims device to circumvent this security measure.

Note: Pre-registering clients might not scale in some deployments (manual process) or require dynamic client registration (not specified yet). With the lack of dynamic client registration, it only works for clients bound to certain deployments at development/configuration time. As soon as dynamic resource server discovery gets involved, that's no longer feasable.

Source: OAuth 2.0 Threat Model and Security Considerations,
http://tools.ietf.org/html/draft-lodderstedt-oauth-security/

Lessons learned

- System design models help to identify threats
- Data pathways as means to
  - identify critical system parts and
  - support risk assessment
- Use of attack trees to capture threats

Literature

- A. Shostack, Threat Modeling: Desining for Security, Wiley 2014
- T. Peltier, Information Security Risk Analysis, Auerbach Publications, 2001
- C. Alberts, A. Dorofee: Managing Information Security Risks-The OCTAVE approach, Addison Wesley, 2003
- B. Schneier: Attack Trees, Dr. Dobbs Journal, 12/1999, http://www.schneier.com/paper-attacktrees-ddj-ft.html
- A. Moore, R. Ellison, R. Linger: Attack Modeling for Information Security and Survivability, Tech. Report CMU/SEI-2001 TN-001, 3/2001
- S. Mauw, M. Oostdijk: Foundations of Attack Trees, Proc. FOSAD, 2005, http://www.sti.uniurb.it/events/fosad05/attacktrees.pdf
- P. Hammett: FMEA, http://www.fmeainfocentre.com/download/umich.pdf