Secure Software Development

Anna Vapen, IDA/ADIT

TDDC90 Software Security
2013-11-08

LiU

1

Agenda

- Software security terminology
- Software development + security
  - Development processes
  - Security activities from the processes
  - Detailed examples of activities: risk analysis, security requirements
- **Main goal of today:** See how security is applied in software development in different ways.

LiU

2

But I want to do technical work…?

- **Today:** broad overview of secure software development
  - Current practices in industry (and academia)
  - Development process == how people build software together
- Who needs to know this?
  - Developers, project leaders, auditors, testers… **you**!
  - When: Seeking financing for security projects, working with !security people, working with others in general

LiU 3

Software Security Terminology
Simple real-life example

Threat agent (burglar)

Threat (burglar breaking in)

Attack (the break-in)

Vulnerability (open window)

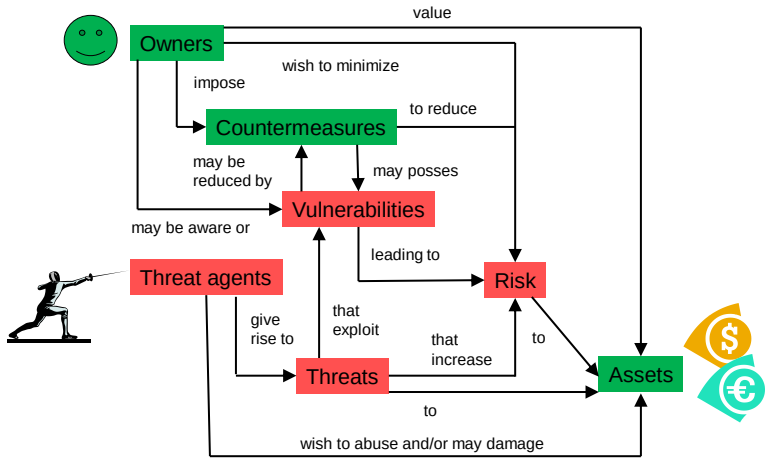
Countermeasure and/or mitigation (alarm)

Owner System

Risk = likelihood * impact

LiU 4

Software Security Terminology



LiU

What is software engineering?

- **Software development:** General, broad term for writing code
- **Software engineering:** The art of developing software
  - With some sort of structured method (development process)
  - In teams in large companies, open source projects, the lab series in this course… and all other places.
- Here we mainly discuss software engineering

LiU

The Software Lifecycle

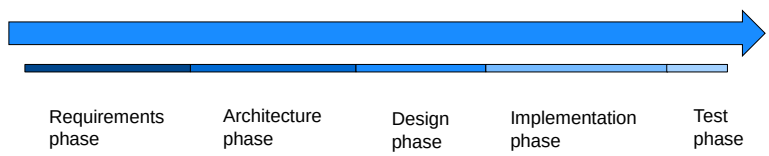
The life of a piece of software – not just built and released!

- **Requirements:** What to build, how should it work
- **Architecture and design:** Overall structure
- **Implementation:** Let's build it!
- **Test:** Does it work?
- **Release:** Distribute it, sell it, show it to the lab assistant.
- **Now what?** Continuous support until end-of-life.

LiU

The Software Lifecycle (contd.)

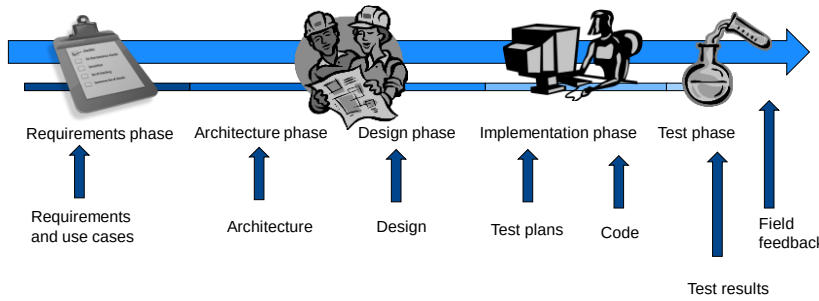
- The steps in the lifecycle (write requirements, design the software etc.) become **phases** in a *software development process*.
- Software development processes: many different approaches
  - Examples: SCRUM, waterfall model etc.
  - Below: very simplified toy-example of a SW dev. process
  - Different phases in different order, depending on the process



LiU

Artifacts in the Software Lifecycle

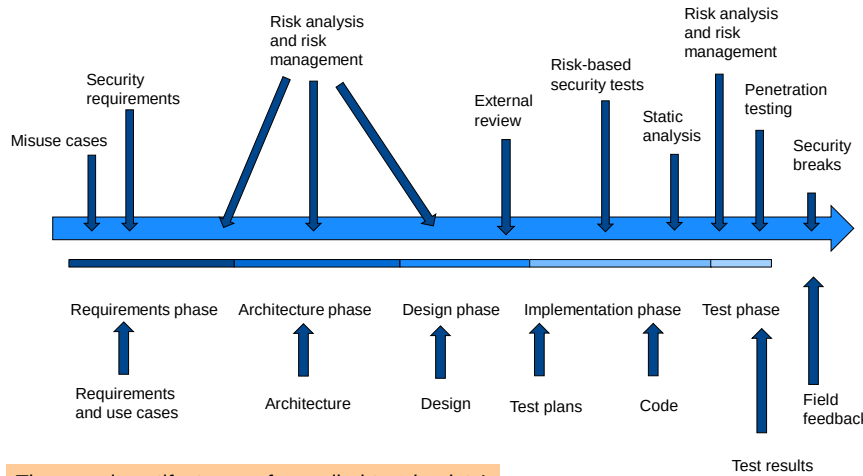
- Artifact: a *thing*, e.g. a test plan, a design document, some code
- Activity: something you do, e.g. write code, preform testing
- Below: example of artifacts in a software development process



LiU

9

Security Artifacts in the Software Lifecycle

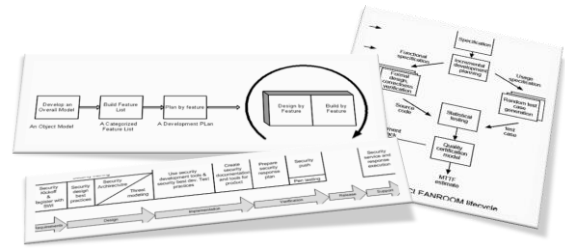


The security artifacts are often called *touchpoints*!

LiU

10

Software Development Processes



LiU

11

Process to Produce Secure Software

- Full lifecycle
- Precise
- Measurable – possible to test how well it works
- Tailored – fit for the purpose/organization/process
- Use current practices

- **Supported with:** training programs, tools, testing…

LiU

12

Examples of Software Development Processes

Three common types of software development processes:

- Incremental development processes
- Requirements-driven development processes
- Agile development processes

Note: We are talking about software development in general now!
Not about secure development (yet!)

LiU 13

Software Development Processes
Incremental development processes

Example: Waterfall model

Requirements

(… other phases …)

Implementation

Test

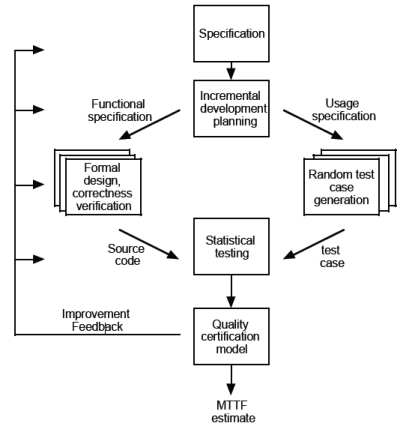
Problems?
Go back and start over!

LiU 14

Software Development Processes
Requirements-driven processes

Example: CLEANROOM

The requirements are
the most important part!
They drive what you build.

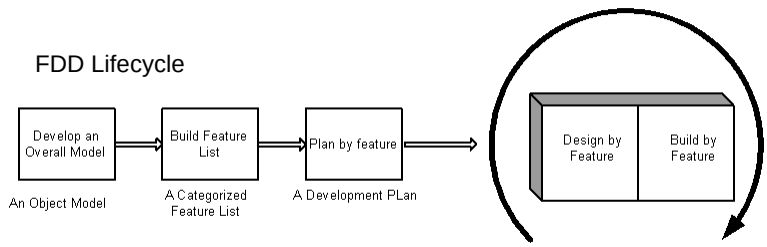


The CLEANROOM lifecycle

Software Development Processes
Agile Processes

- Examples:
  - SCRUM
  - Extreme programming (XP)
  - Feature driven development (FDD)

FDD Lifecycle



Security and Software Development Processes

Three types of approaches to security in software development:

- **Process-specific solutions**
  - A development process specifically designed to include security from the beginning.
  - Designed from scratch of a “secure” version of an existing process.
- **Security plug-ins**
  - A small process of it’s own, to be added to an existing development process.
- **Ad-hoc application of best practices**
  - Adding security practices the way we feel like... Where it fits!

LiU 17

Ex: (In)secure Software Development

Bad Software Inc. wants to improve their development process.

- Current process:

Make big plans Write bad code Patch software

- How to secure it? Can we help them out?

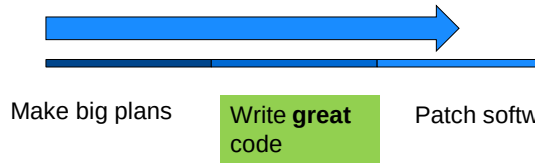
This example is (hopefully) unrealistic!

LiU 18

Ex: (In)secure Software Development (2)

Bad Software Inc. wants to improve their development process.

- Let's try: **Process-specific solution** for secure SW development
  - Throw away the old development process
  - Pick a development process with security included



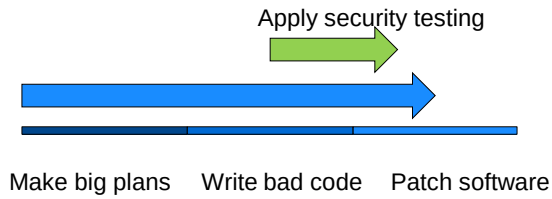
LiU

19

Ex: (In)secure Software Development (3)

Bad Software Inc. wants to improve their development process.

- Let's try: **Security plug-in**
  - Keep the old development process
  - Add a small development process (security only)



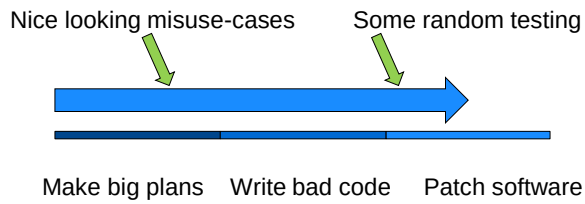
LiU

20

Ex: (In)secure Software Development (4)

Bad Software Inc. wants to improve their development process.

- Let's try: **Add-hoc application of best practices**
  - Keep the old development process
  - Add security touch-points to your liking



LiU 21

Secure Software Development

And now for some real examples on how this should be done!

- SDL
- TSP and TSP-Secure
- CLASP
- S^3P

LiU 22

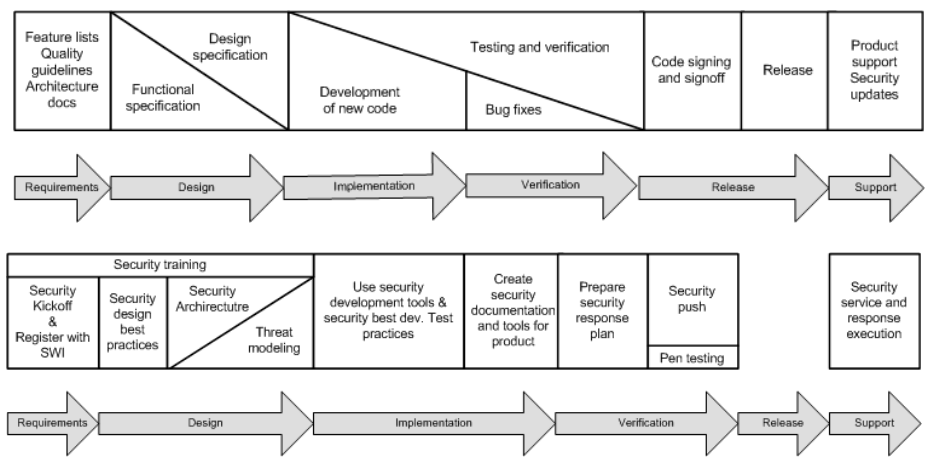
Example: SDL

Type: Process specific solution

- Development of software that needs to withstand attacks:
  - Threat modeling
  - Static analysis
  - Code reviews and security testing
  - Final security review (by another team)

LiU 23

SDL



LiU 24

Example: Team Software Process (TSP)

Type: Application of best practices

- High-level guidance for development team
  - Manage and remove defects
  - Measurements and quality management
  - Monitor the process
  - Use predictive measures for remaining defects

Note: TSP does not include security, but… TSP-Secure does!

LiU

25

Example: TSP-Secure

- TSP-Secure: augment TSP with security practices
  - Additional training in security issues
  - Security oriented design
  - Security conscious implementation
- Difficulties:
  - Training
  - Disciplined methods

LiU

26

Example: CLASP

Type: Security plug-in

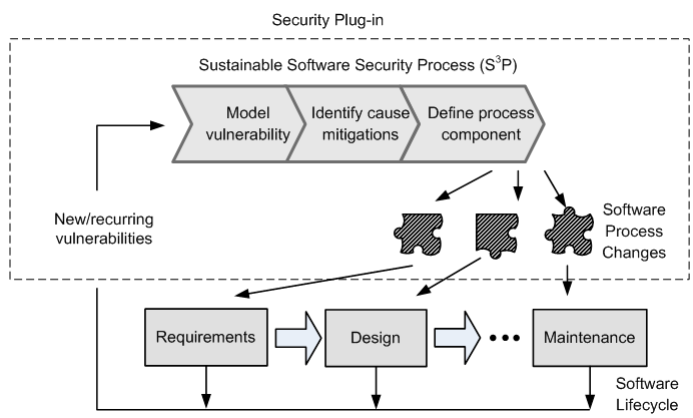
- CLASP (Comprehensive Lightweight Application Security Process)
  - Activities for development team members
  - Plug-in for Rational Unified Process (RUP)

LiU

27

Example: Sustainable Software Security Process S^3P

Type: Security plug-in

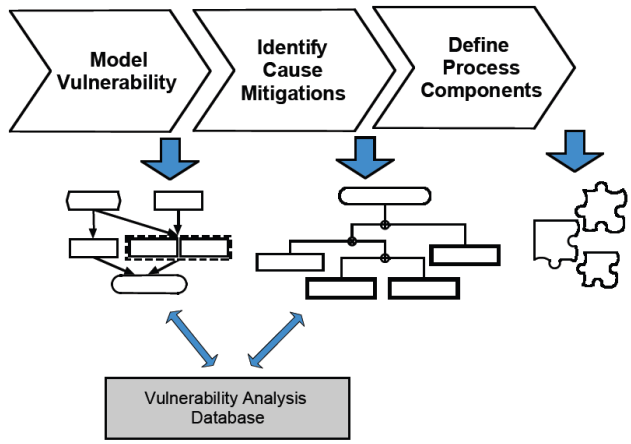


LiU

28

S^3P

This will be covered more in detail at the modeling lecture.



LiU

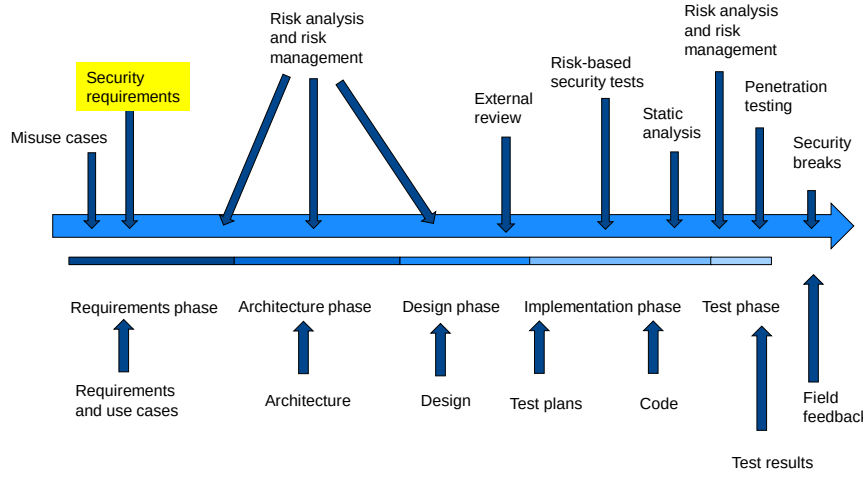
29

Security Requirements

LiU

30

Security artifacts in the Software Lifecycle



Security Requirements

- Early in development (*not* after deployment)
- Methodologies for security requirement engineering
  - SQUARE
  - Common Criteria (CC)
  - …and more.

Security Requirements (contd.)

- Functional and non-functional requirements
  - **What** the system must do (testable)
  - **How** the system must do it

LiU

33

Security Requirements (contd.)

Examples of fields in which we may need security requirements:

- Identification and authentication
- Authorization
- Immunity
- Integrity
- Intrusion detection
- Non-repudiation
- Privacy
- Physical protection

LiU

34

Security Requirement Method: CC

- Common Criteria
  - Security *functional* requirements
  - Security *assurance* requirements
- Evaluation and certification

LiU

35

Security Requirement Method: SQUARE

- Nine steps
  - Agree on **definitions**
  - Identify safety and security **goals**
  - Select elicitation **techniques**
  - Develop **artifacts** to support elicitation techniques
  - **Elicit** safety and security requirements
  - **Categorize** requirements
  - Perform **risk assessment**
  - **Prioritize** requirements
  - Requirements **inspection**

LiU

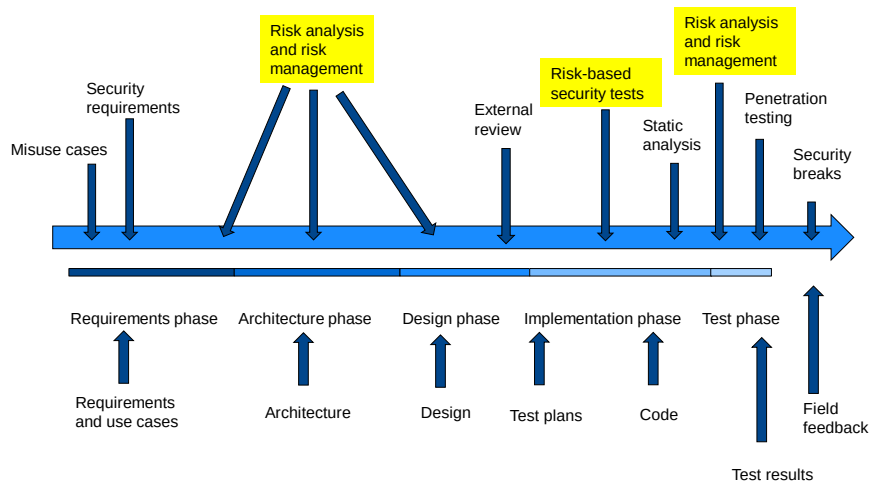
36

Risk Management and Risk Analysis

LiU

37

Security artifacts in the Software Lifecycle



LiU

38

Risk Management vs. Risk Analysis

- Key concepts: **Threat – vulnerability – damage**
- Risk = likelihood * impact

- **Risk analysis:** *Identifying* risks
- **Risk management:** *Dealing* with risks found during risk analysis

- Risk analysis is part of the overall risk management work!

LiU 39

Security Risk Analysis
Doing risk analysis with security in mind

- **Think as an attacker:** Learn about the target
- **Teamwork:** Discuss security issues
- **Rank risks:** High likelihood and high impact = high risk!
  - Step 1: Determine probability of compromise (attack success)
  - Step 2: Perform impact analysis (level of damage)
- **Mitigation strategy:** How to mitigate risks?
- **Report:** What did we find?

LiU 40

Risk Management Steps

- Identify **assets**
- Identify **vulnerabilities** in the asset and in the systems directly interacting with asset
- For every asset and associated vulnerabilities:
  - Estimate the **consequence** of loosing the asset (the cost of replacing or restoring an asset)
  - Estimate the expected **rate of occurrence** of the vulnerability
- **Defend** against risks:
  - Reduce the value of asset to attackers (e.g. encrypt data)
  - Mitigate vulnerabilities
  - Prevent attacks

LiU 41

Example: Annual Loss Expectancy (ALE)

DoS attack on the mail server of Company A

Item Description	Estimated Cost
Recovery: External consultant, 4 hours * $150	$600
Lost productivity: 5 employees * 2 hours * $34	$340
Long distance phone calls	$10
Total cost of the attack:	**$950**

- Estimated likelihood of the DoS attack: ~0.5 incident per year
- Estimated loss: *950 ($/incident) x 0.5 (incident/year) = 475 ($/year)*

ALE is an activity that could be part of risk management!

LiU 42

Methodologies for Handling Risks
Risk analysis and/or risk management

Standard-based

- **OCTAVE** (Operationally Critical Threat, Asset, and Vulnerability Evaluation) from SEI
- **COBIT** (Control Objectives for Information and Related Technology) from ISACA (Information Systems Audit and Control Association)

Commercial

- **STRIDE** (Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, Elevation of privilege)
- **RMF** (Risk management Framework)

LiU 43

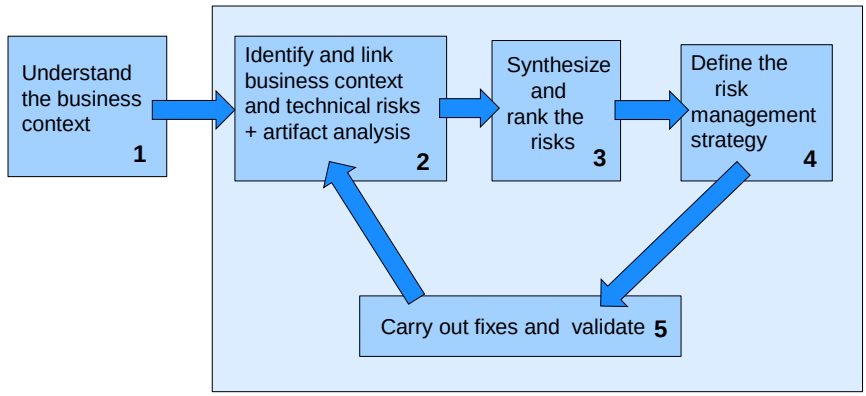
Risk Management Framework (RMF)

- Continuous software risk management process
- A full lifecycle activity
- Manages software-induced **business risks**

- Software security risks
  - Risks introduced by insufficient processes
  - Risks introduced by people

LiU 44

RMF (contd.)



Risk analysis: steps 1, 2, and 3
Implementation and operation: steps 4 and 5

LiU

45

Architectural Risk Management

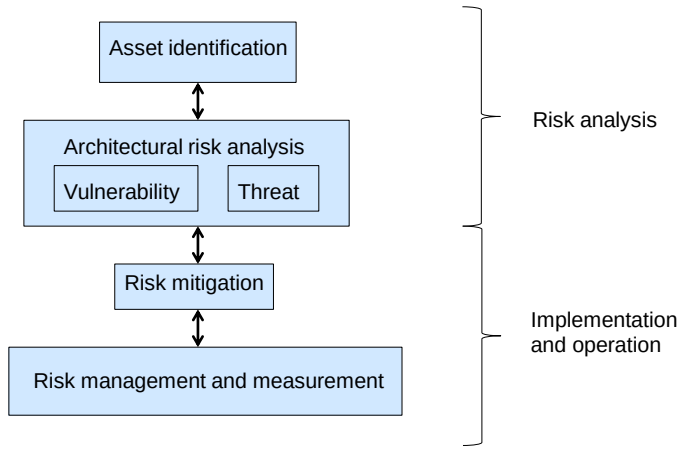
- RMF: specific risk management method
- Architectural risk management: wider approach
- A risk management process to:
  - Identify flaws in a software architecture
  - Determine risks to business information assets that results from those flaws

LiU

46

Architectural Risk Analysis

Part of architectural risk management



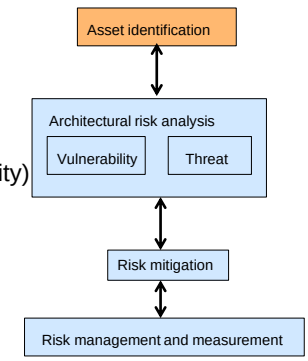
LiU

47

Architectural Risk Management

Step by step: Asset identification

- Identify assets to be protected e.g. databases, credentials, audit records, financial information…
- Gain initial information about assets
  - Ex: Properties to be maintained on the asset (confidentiality, integrity, availability)
  - Business impact if the property is not maintained
- Identify software modules that manipulate the assets

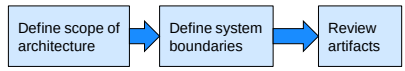


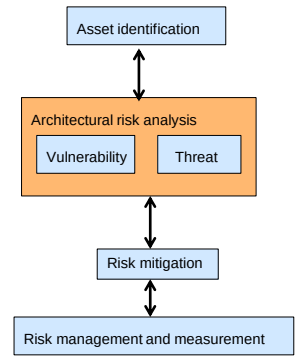
LiU

48

Architectural Risk Analysis
Step by step: Actual ARM

- Application characterization





- Architectural vulnerability assessment
  - Known vulnerability analysis
  - Ambiguity analysis
  - Underlying platform vulnerability analysis

LiU

49

Architectural Risk Analysis
Step by step: Threat analysis

- Assume given access and skill level for the attacker
- Map vulnerabilities to threats to understand how system may be exploited

Threat source	Motivation	Threat Actions
Malware writer	Economic profit	Write and release malicious software
[Who?]	[Why?]	[What?]
…	…	…

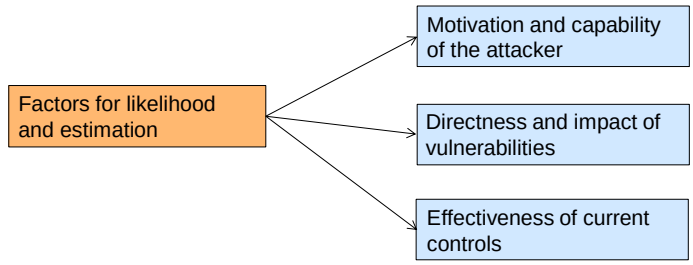
Threat source == threat agent

LiU

50

Architectural Risk Analysis
Step by step: Risk likelihood determination

- **Likelihood:** qualitative estimation of how likely a successful attack will be, based on analysis and past experience
- Not for new types of attacks



LiU

51

Architectural Risk Analysis
Step by step: Risk calculation

Likelihood \ Impact	High	Medium	Low
High	High	High	Medium
Medium	High	Medium	Low
Low	Medium	Low	Low

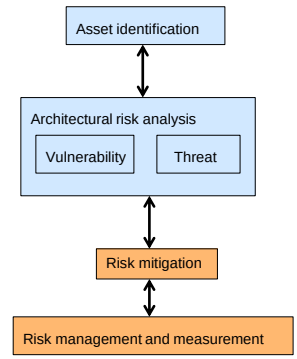
LiU

52

Architectural Risk Management

Step by step: Mitigation, management and measurement

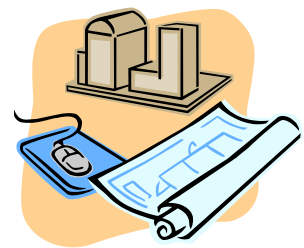
- Risk mitigation
  - Reduce likelihood
  - Reduce impact
- Risk management and measurement
  - Not all risks can be mitigated
  - Accepting the risk
  - Outsourcing risk via insurance
  - Partial mitigation
  - Monitoring risk exposure over time



LiU

53

Capability Models



LiU

54

Capability Maturity Model (CMM)

- Born from military research in the 80's to avoid:
  - Project failure
  - Over budget
  - Finished to late
- Objective **evaluation** of software development **processes**
- Variations: SW-CMM, CMMI, iCMM, SSE-CMM

LiU 55

Capability Maturity Model (CMM)

- Helps organizations to improve their capability to perform a particular process
- Also used to **evaluate organizations**:
  - **Process capability** → measures performance
  - **Process maturity** → measures how defined, managed, measured, and controlled a process is

LiU 56

Capability Maturity Model (CMM)

- Five levels of software process maturity, based on an organization's support for certain process areas (PAs).
  - Level 1: Initial
  - Level 2: Repeatable
  - Level 3: Defined
  - Level 4: Managed
  - Level 5: Optimized

LiU 57

Capability Maturity Model (CMM)

- **Key process areas:**
  - Requirement activities and artifacts
  - Documentation & specifications
  - Audits & inspections
  - Documented processes and procedures
- **Not in CMM:**
  - The software itself
  - Technical artifacts (use cases, design models, code…)

LiU 58

Other CMM Models: CMMI

- CMM Integration (CMMI) integrates various CMMs
  - CMMs from different fields → Set of integrated models

LiU

59

Other CMM Models: SSE-CMM

- System Security Engineering Maturity Model (SSE-CMM)
- Scope:
  - Security engineering practices
  - Throughout development and **support**
- Process areas (PAs)
  - Engineering
  - Project
  - Organizational

LiU

60

Summary

- Terminology
- Software engineering and security
- Development processes for adding security
  - SDL, TSP and TSP-Secure, CLASP, S3P
- Security requirements
  - SQUARE, CC
- Risk analysis and risk managements
  - Overview, Architectural risk management, RMF
- Capability maturity models
  - CMM and its variations

LiU

61

What's next?

- Introduction – why software security?
- Secure software development
- Vulnerabilities, exploits and testing
- Software inspections
- Static analysis
- Security modeling
- Guest lecture from industry

LiU

62