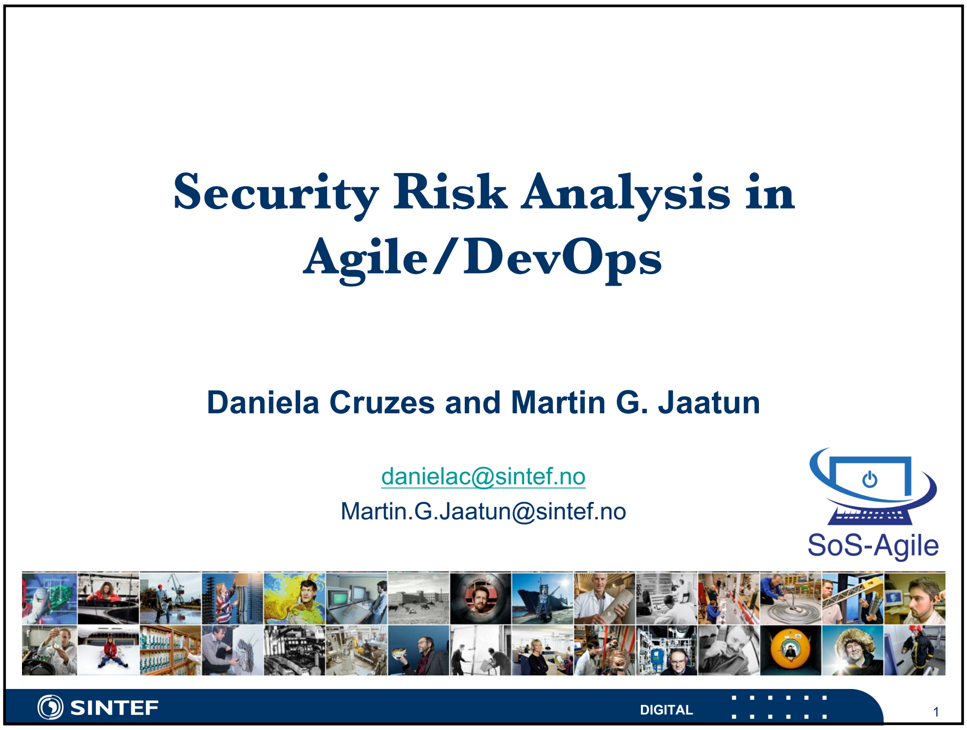
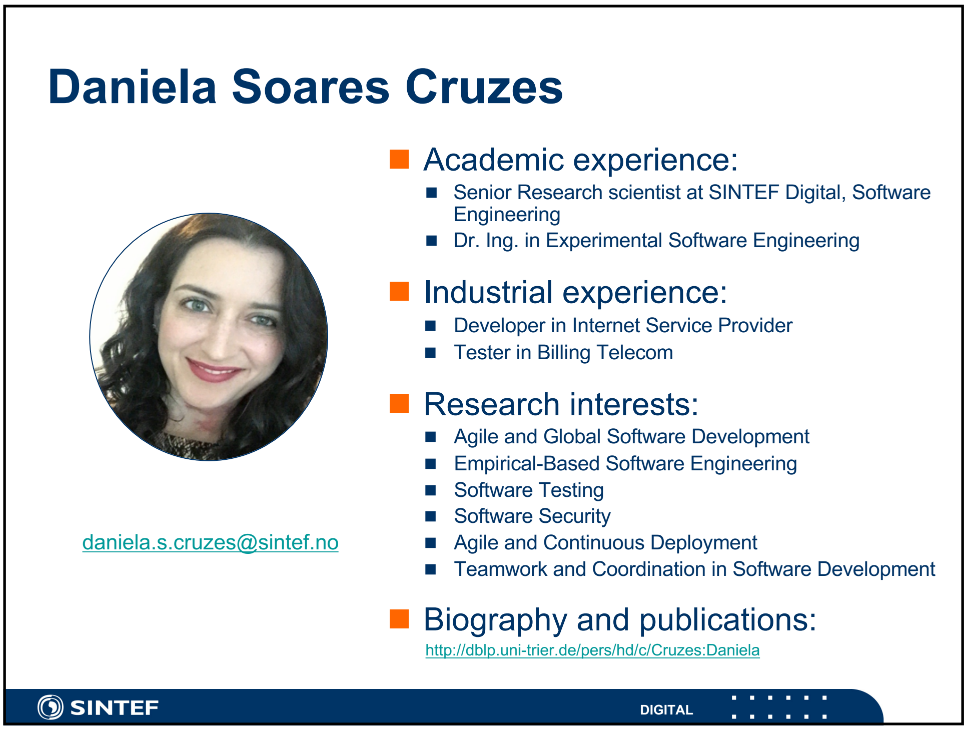


Security Risk Analysis in Agile/DevOps

Daniela Cruzes and Martin G. Jaatun

danielac@sintef.no
Martin.G.Jaatun@sintef.no



Daniela Soares Cruzes

- Academic experience:
  - Senior Research scientist at SINTEF Digital, Software Engineering
  - Dr. Ing. in Experimental Software Engineering
- Industrial experience:
  - Developer in Internet Service Provider
  - Tester in Billing Telecom
- Research interests:
  - Agile and Global Software Development
  - Empirical-Based Software Engineering
  - Software Testing
  - Software Security
  - Agile and Continuous Deployment
  - Teamwork and Coordination in Software Development
- Biography and publications:
  http://dblp.uni-trier.de/pers/hd/c/Cruzes:Daniela

daniela.s.cruzes@sintef.no

Martin G. Jaatun

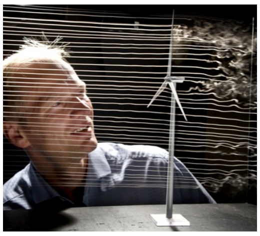
Martin.G.Jaatun@sintef.no

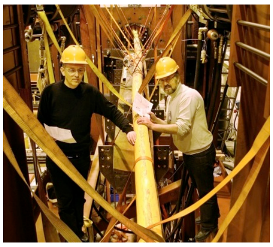
- Academic experience:
  - Senior Scientist at SINTEF ICT, Information Security
  - Dr. Philos. from UiS in Critical Information System Security
- Industrial experience:
  - Security Consultant in System Sikkerhet AS, Norway
  - Senior developer in Shield Data A/S
- Research interests:
  - Software security
  - Critical infrastructure security (SmartGrid, Oil&Gas)
  - Cloud security
- Biography and publications:

  https://scholar.google.no/citations?user=R8Fw41kAAAAJ&hl=no&oi=ao
  http://dblp.uni-trier.de/pers/hd/j/Jaatun:Martin_Gilje

SINTEF

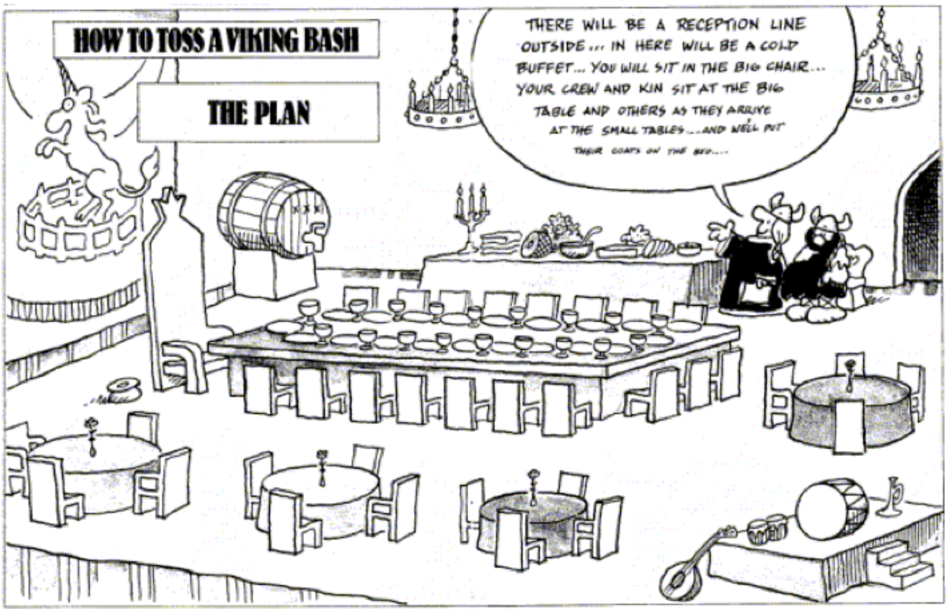
- A non-commercial research foundation with subsidiaries
- Largest independent research organisation in Scandinavia.
- Leading expertise in the natural sciences and technology, environment, health and social science
- 2100 employees from 68 countries.
- 1350 researchers
- 48% of our researchers hold doctorates
- Customers in 61 countries



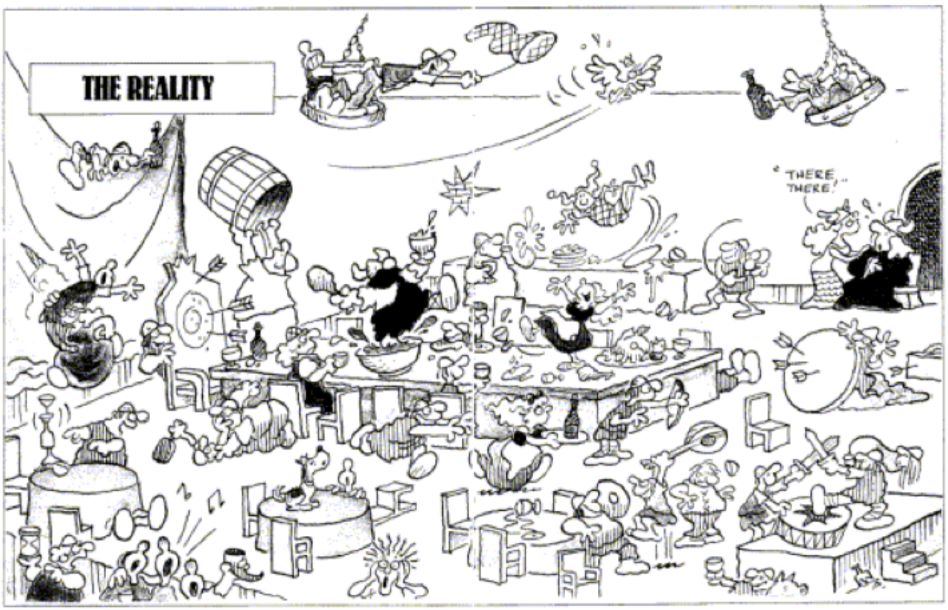


Agenda

- Introduction to Software Security
  - Software Security and Agile
- Threat Modeling and Risks
  - Assets
  - Threats Categorization
    - STRIDE
    - Top Ten Owasp
  - Trust Levels
    - Attack Surface Analysis
    - Attack Tree
  - Threat Analysis with Data Flow Diagrams


HOW TO TOSS A VIKING BASH
THE PLAN
THERE WILL BE A RECEPTION LINE OUTSIDE... IN HERE WILL BE A COLD BUFFET... YOU WILL SIT IN THE BIG CHAIR... YOUR CREW AND KIN SIT AT THE BIG TABLE AND OTHERS AS THEY ARRIVE AT THE SMALL TABLES... AND WE'LL PUT THEIR COATS ON THE BED...

Slide from Tore Dybå: What means to be agile? FARA Workshop, Jan 2015

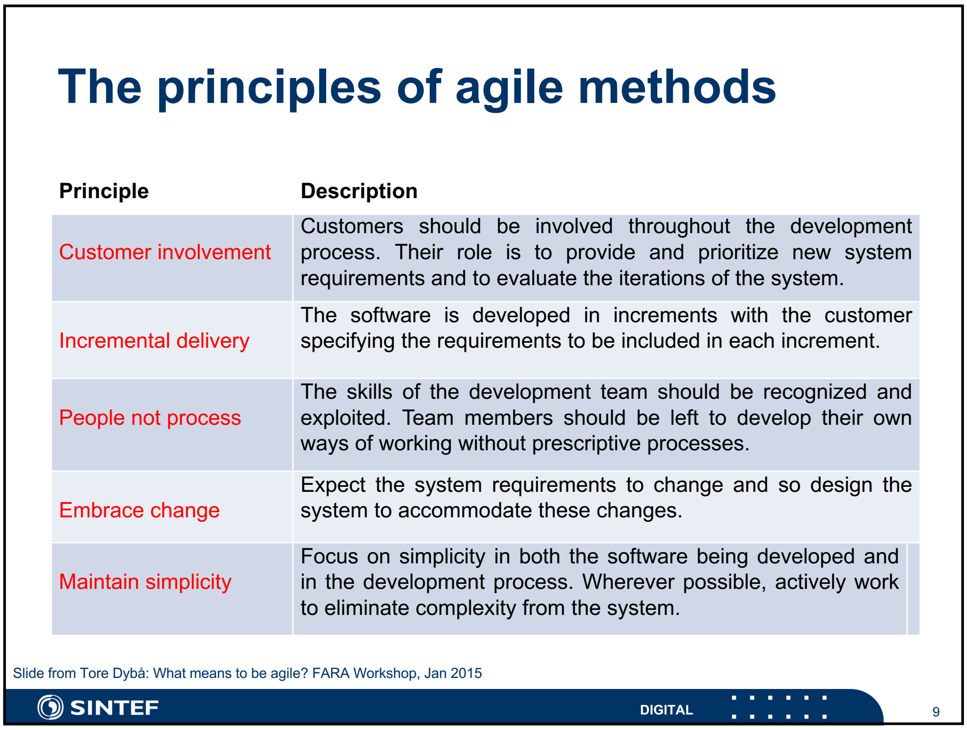


Slide from Tore Dybå: What means to be agile? FARA Workshop, Jan 2015

Plan-driven and agile development

- Plan-driven development:
  - A plan-driven approach to software engineering is based around separate development stages with the outputs to be produced at each of these stages planned in advance.
  - Not necessarily waterfall model – plan-driven, incremental development is possible
  - Iteration occurs within activities.
- Agile development:
  - Specification, design, implementation and testing are inter-leaved and the outputs from the development process are decided through a process of negotiation during the software development process.
  - Reduce overheads in the software process and to be able to respond quickly to changing requirements without excessive rework

Slide from Tore Dybå: What means to be agile? FARA Workshop, Jan 2015

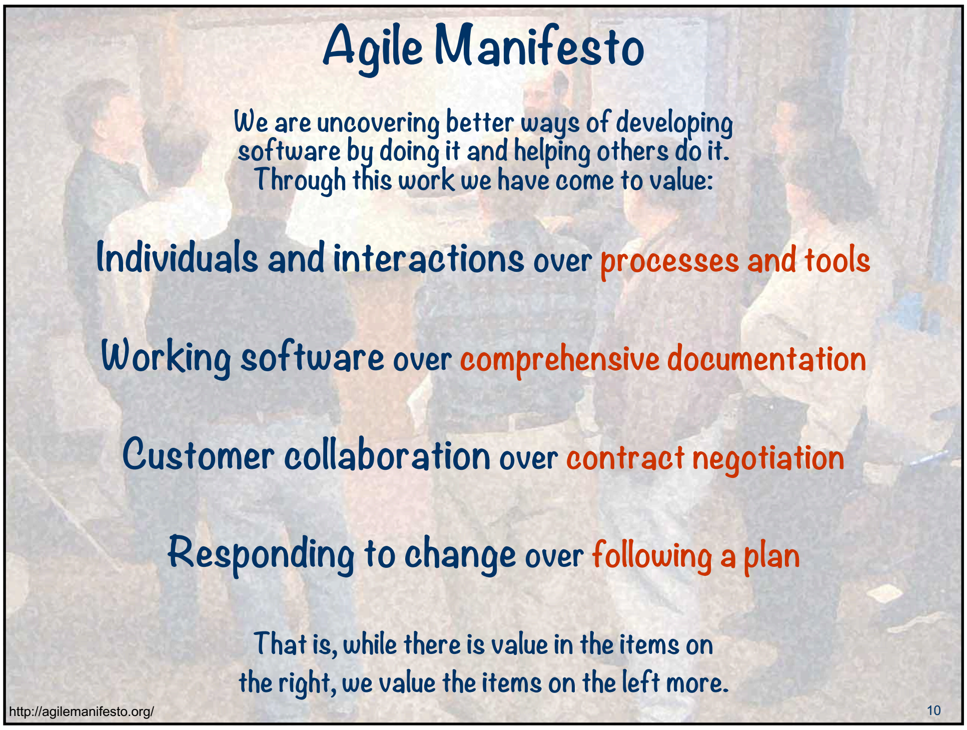


The principles of agile methods

Principle	Description
Customer involvement	Customers should be involved throughout the development process. Their role is to provide and prioritize new system requirements and to evaluate the iterations of the system.
Incremental delivery	The software is developed in increments with the customer specifying the requirements to be included in each increment.
People not process	The skills of the development team should be recognized and exploited. Team members should be left to develop their own ways of working without prescriptive processes.
Embrace change	Expect the system requirements to change and so design the system to accommodate these changes.
Maintain simplicity	Focus on simplicity in both the software being developed and in the development process. Wherever possible, actively work to eliminate complexity from the system.

Slide from Tore Dybå: What means to be agile? FARA Workshop, Jan 2015

SINTEF DIGITAL 9



Agile Manifesto

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

Individuals and interactions over processes and tools

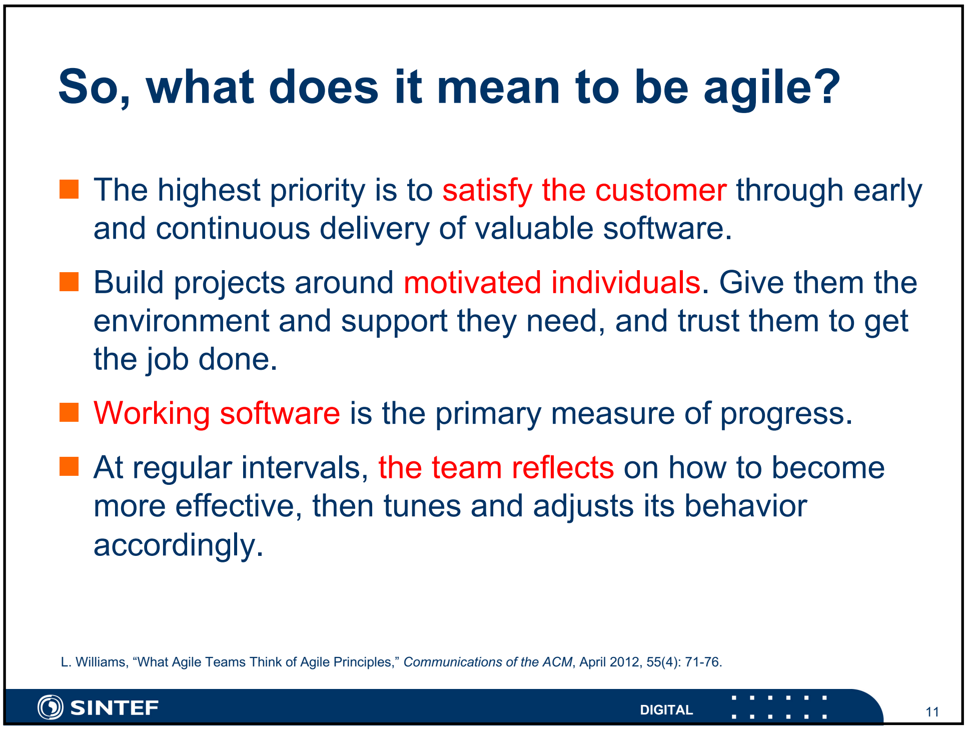
Working software over comprehensive documentation

Customer collaboration over contract negotiation

Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

http://agilemanifesto.org/ 10

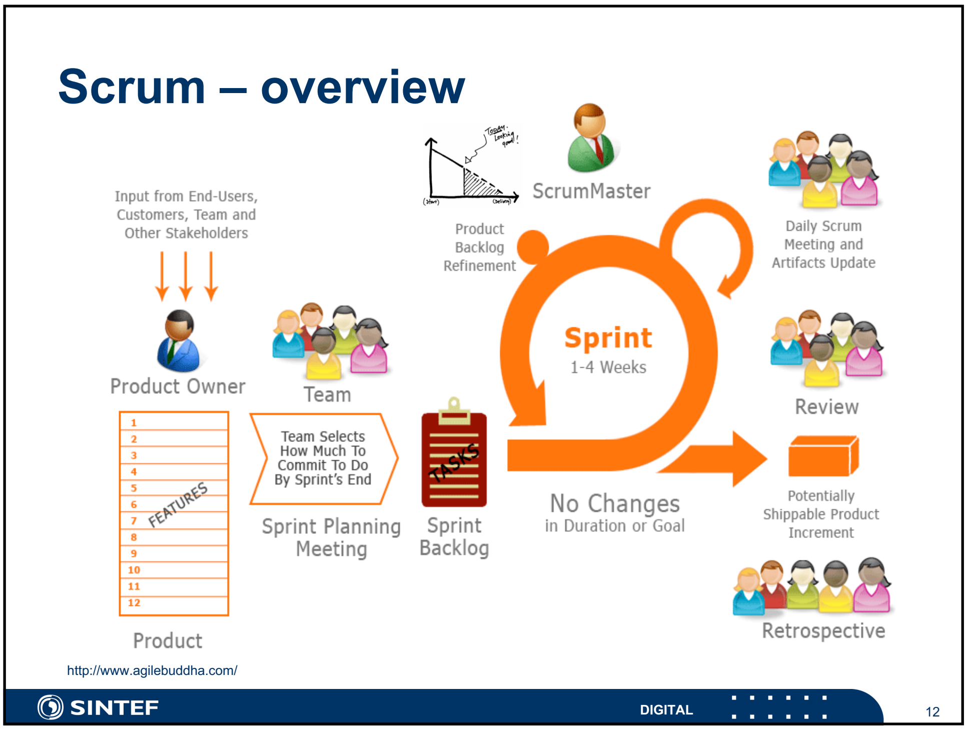


So, what does it mean to be agile?

- The highest priority is to satisfy the customer through early and continuous delivery of valuable software.
- Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
- Working software is the primary measure of progress.
- At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

L. Williams, “What Agile Teams Think of Agile Principles,” *Communications of the ACM*, April 2012, 55(4): 71-76.

SINTEF DIGITAL 11



Scrum – overview

Input from End-Users, Customers, Team and Other Stakeholders

Product Owner

Team

Team Selects How Much To Commit To Do By Sprint’s End

Sprint Planning Meeting

Product Backlog Refinement

ScrumMaster

Sprint Backlog

Sprint 1-4 Weeks

No Changes in Duration or Goal

Daily Scrum Meeting and Artifacts Update

Review

Potentially Shippable Product Increment

Retrospective

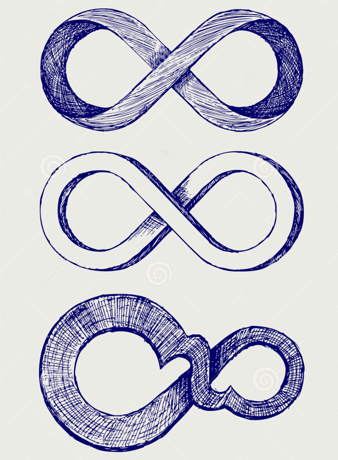
Product

http://www.agilebuddha.com/

SINTEF DIGITAL 12

Agile: SW development becomes continuous

- Continuous Planning
- Continuous Budgeting
- Continuous Integration
- Continuous Deployment
- Continuous Delivery
- Continuous Verification
- Continuous Innovation
- Continuous Experimentation



How do these values affect security?

Agile Manifesto

Individuals and interactions

over processes and tools

Working software

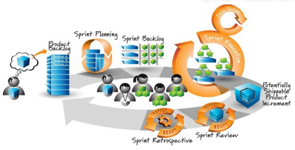
over comprehensive documentation

Customer collaboration

over contract negotiation

Responding to change

over following a plan



Devops Manifesto

by Ernerst Mueller

Individuals and interactions

over processes and tools

Working systems

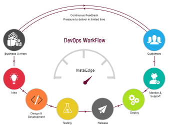
over comprehensive documentation

Customer, developers and operations collaboration

over contract negotiation

Responding to change

over following a plan



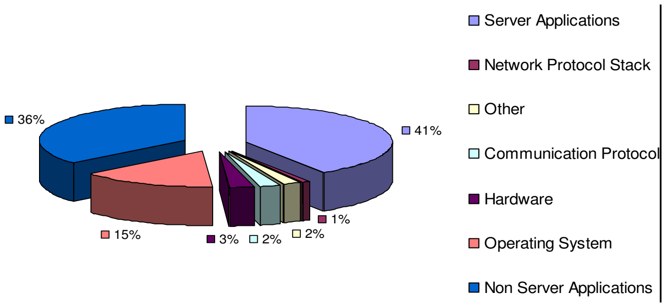
https://theagileadmin.com/2010/10/15/a-devops-manifesto/

Software Security and Agile

What is the problem then?

Over 70% of security vulnerabilities exists at the application layer, not the network layer

Around 50% of the vulnerabilities that can be introduced during the implementation phase are consequences of design flaws (CWE Database)



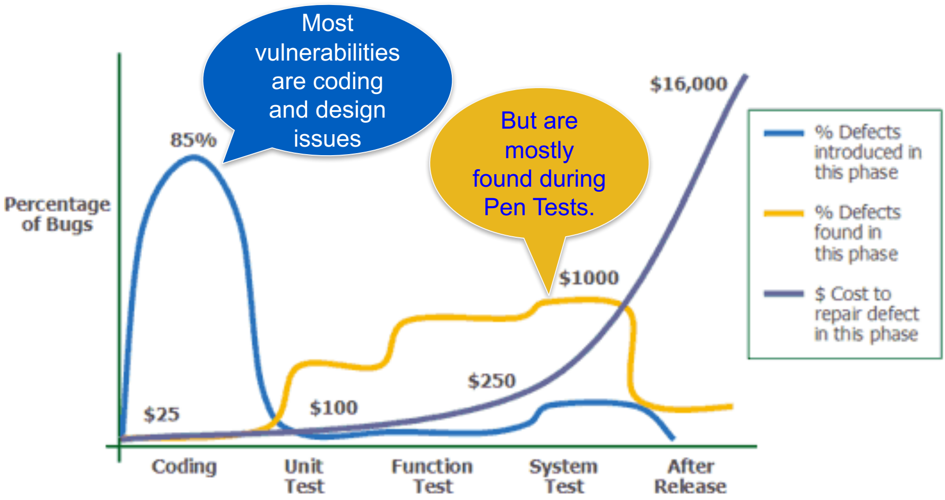
Marco M. Morana "Building Security Into The Software Life Cycle A Business Case"

If you haven't reviewed your code for security holes, the likelihood that your application has problems is virtually 100%.

- Over the last 10 years, the team involved with the OWASP Code Review Project has performed thousands of application reviews, and found that **every single application has had serious vulnerabilities.**

OVER
70%
of organizations report having been compromised by a successful cyberattack in the past 12 months.

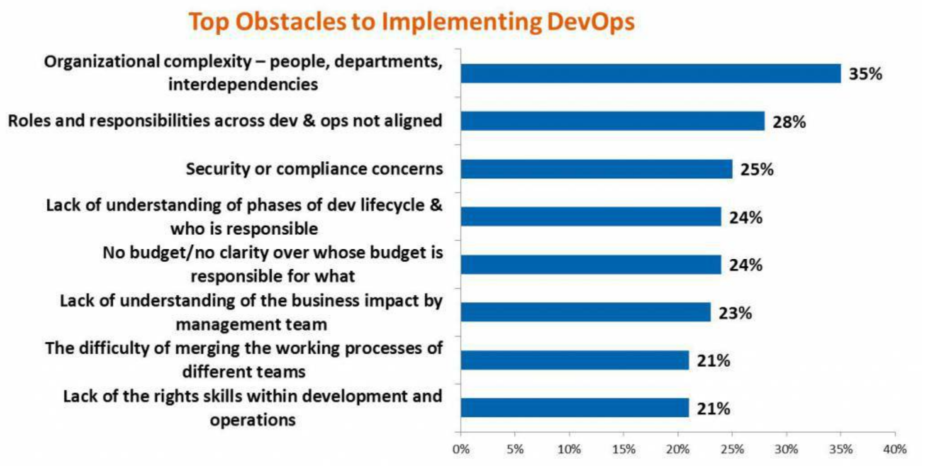
The cost of fixing a bug in in Production is 100x bigger than during coding/design



Most vulnerabilities are coding and design issues

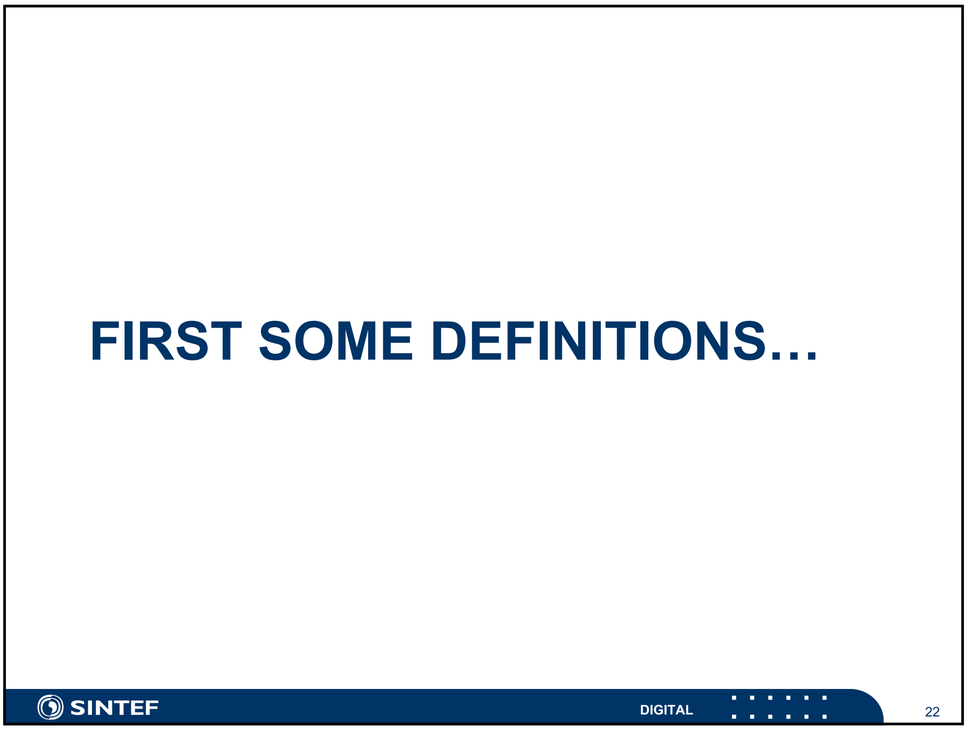
But are mostly found during Pen Tests.

Source: Applied Software Measurement,



http://blogs.ca.com/2013/09/12/survey-devops-an-overnight-success-because-it-works/

What are the challenges for software security in agile development?



FIRST SOME DEFINITIONS…

Managing Risks

- Probability of occurrence (PO) is the product of one manageable value and one nearly unmanageable value: vulnerabilities and threats.

$$\text{Risk} = \frac{\text{Probability of Occurrence} \times \text{Business Impact}}{\text{Controls}}$$

http://resources.infosecinstitute.com/enterprise-security-book-chapter-1/

Vulnerabilities and Threats

- **Vulnerabilities** are weaknesses in a system, network, or process.
  - A more business-focused definition would be weaknesses in people, processes, or technology.

- **Threats** are technical, human, or natural events—either accidental or intentional—that exploit vulnerabilities.

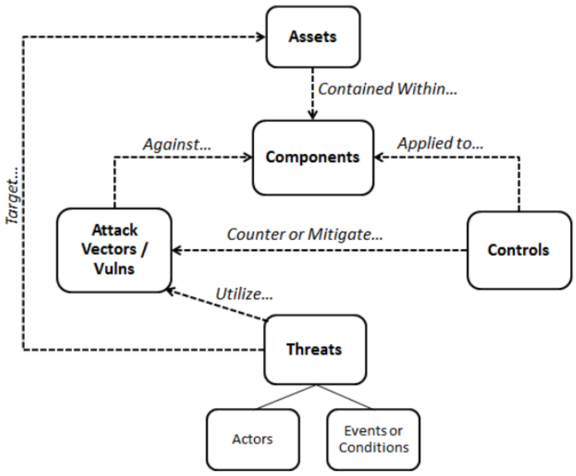
http://resources.infosecinstitute.com/enterprise-security-book-chapter-1/

Assets and Attacks

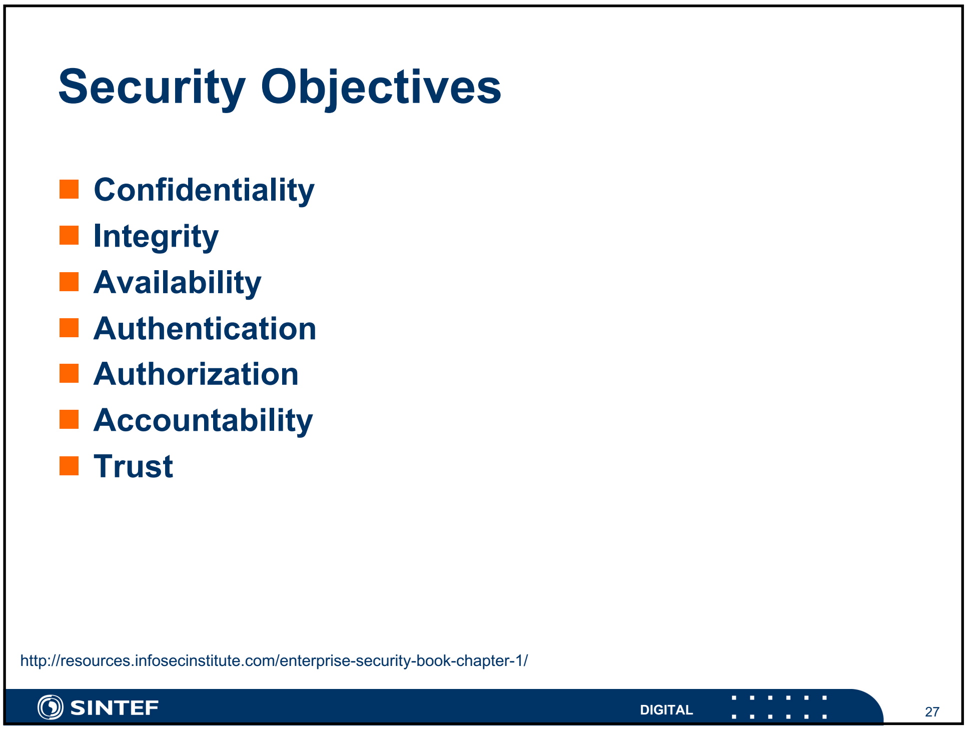
- **Asset.** Something that is of value and that you want to protect. A resource of value, such as the data in a database or on the file system.
- **Attack (or exploit).** An action taken by someone or something that harms an asset. This could be someone following through on a threat or exploiting a vulnerability.
- **Countermeasure/Control.** A safeguard that addresses a threat and mitigates risk.

http://resources.infosecinstitute.com/enterprise-security-book-chapter-1/

Threats, Assets and Controls Relationship Model



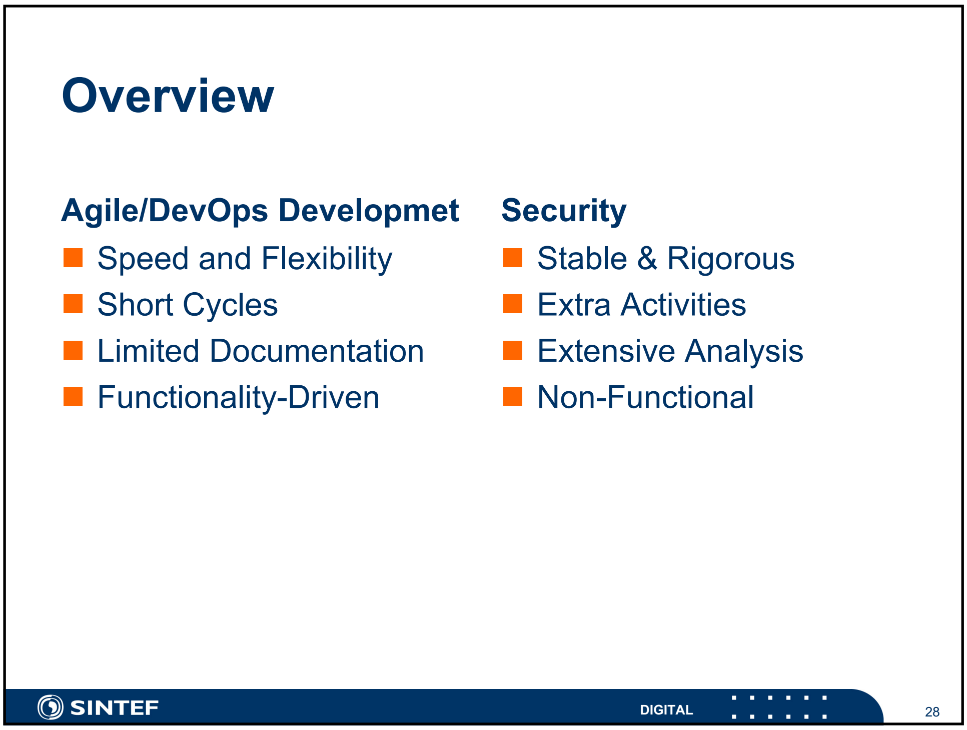
source:http://lockheedmartin.com/content/dam/lockheed/data/isgs/documents/Threat-Driven%20Approach%20whitepaper.pdf



Security Objectives

- Confidentiality
- Integrity
- Availability
- Authentication
- Authorization
- Accountability
- Trust

http://resources.infosecinstitute.com/enterprise-security-book-chapter-1/



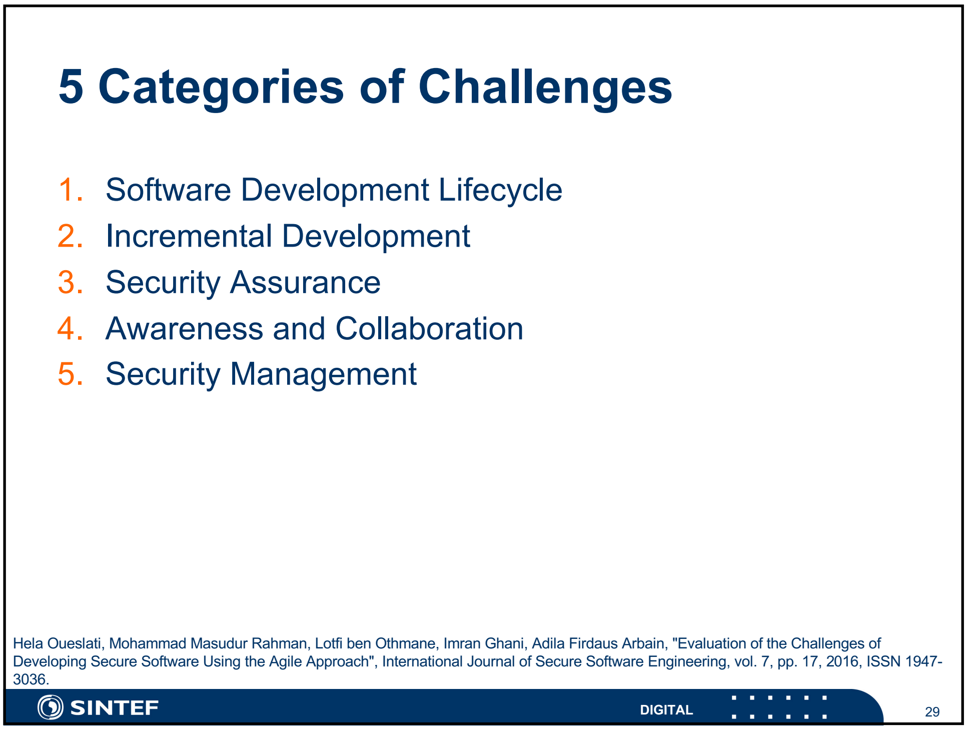
Overview

Agile/DevOps Developmet

- Speed and Flexibility
- Short Cycles
- Limited Documentation
- Functionality-Driven

Security

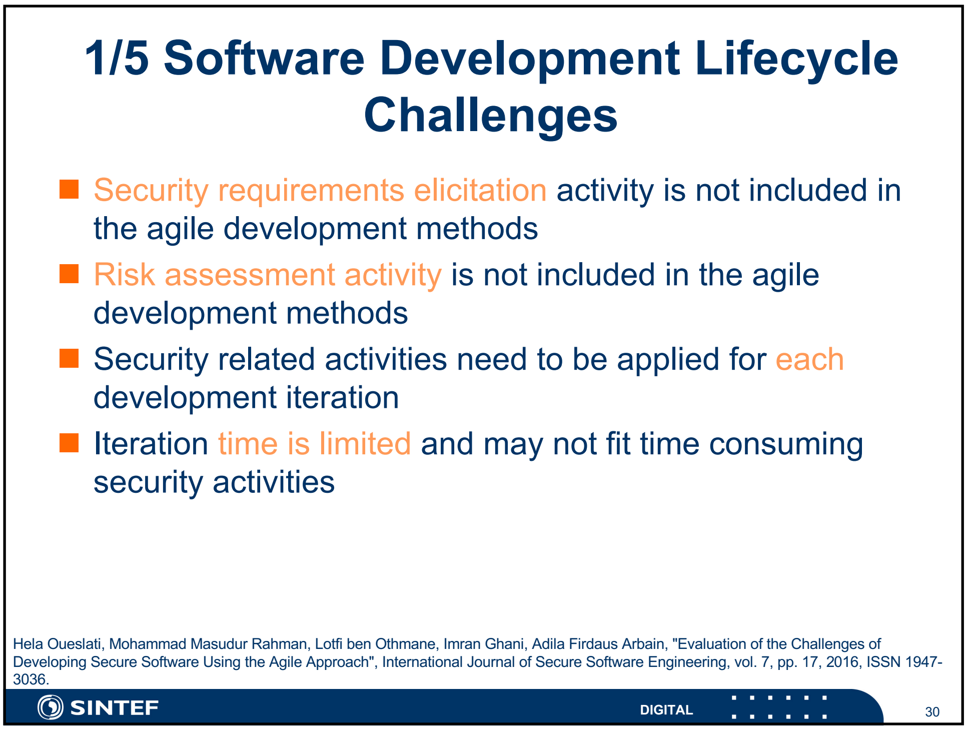
- Stable & Rigorous
- Extra Activities
- Extensive Analysis
- Non-Functional



5 Categories of Challenges

1. Software Development Lifecycle
2. Incremental Development
3. Security Assurance
4. Awareness and Collaboration
5. Security Management

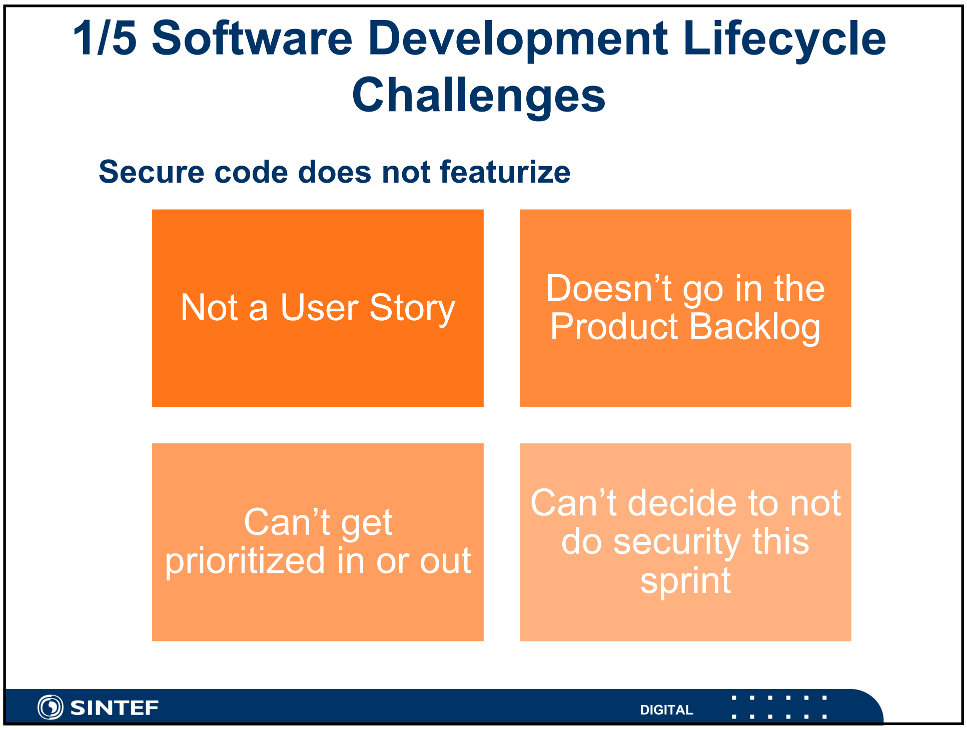
Hela Oueslati, Mohammad Masudur Rahman, Lotfi ben Othmane, Imran Ghani, Adila Firdaus Arbain, "Evaluation of the Challenges of Developing Secure Software Using the Agile Approach", International Journal of Secure Software Engineering, vol. 7, pp. 17, 2016, ISSN 1947-3036.



1/5 Software Development Lifecycle Challenges

- Security requirements elicitation activity is not included in the agile development methods
- Risk assessment activity is not included in the agile development methods
- Security related activities need to be applied for each development iteration
- Iteration time is limited and may not fit time consuming security activities

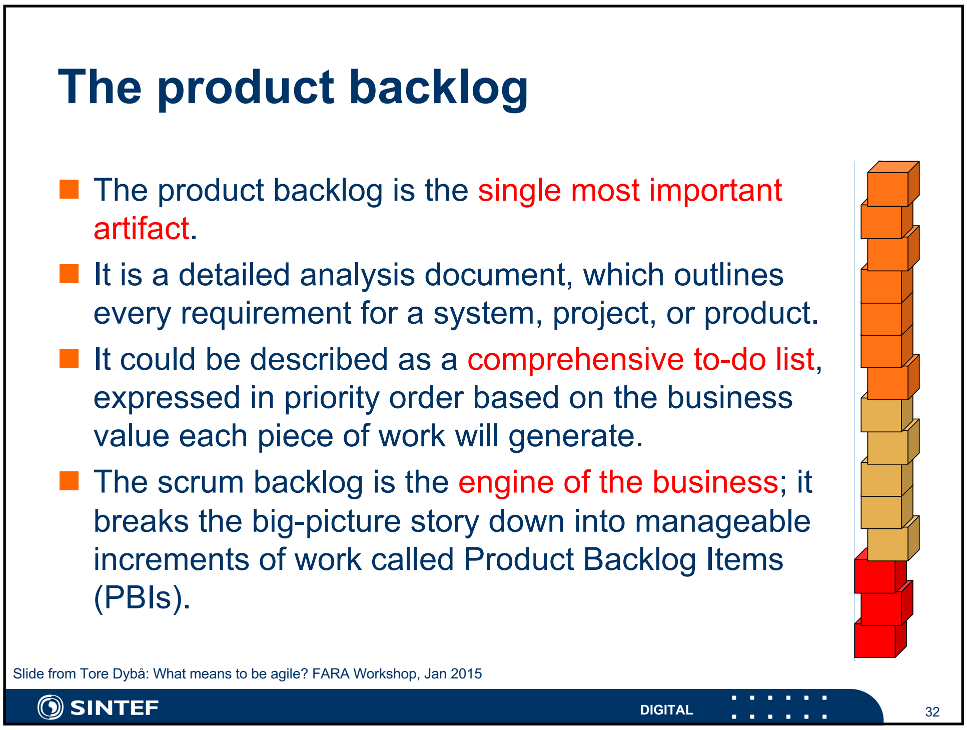
Hela Oueslati, Mohammad Masudur Rahman, Lotfi ben Othmane, Imran Ghani, Adila Firdaus Arbain, "Evaluation of the Challenges of Developing Secure Software Using the Agile Approach", International Journal of Secure Software Engineering, vol. 7, pp. 17, 2016, ISSN 1947-3036.



1/5 Software Development Lifecycle Challenges

Secure code does not featurize

- Not a User Story
- Doesn't go in the Product Backlog
- Can't get prioritized in or out
- Can't decide to not do security this sprint

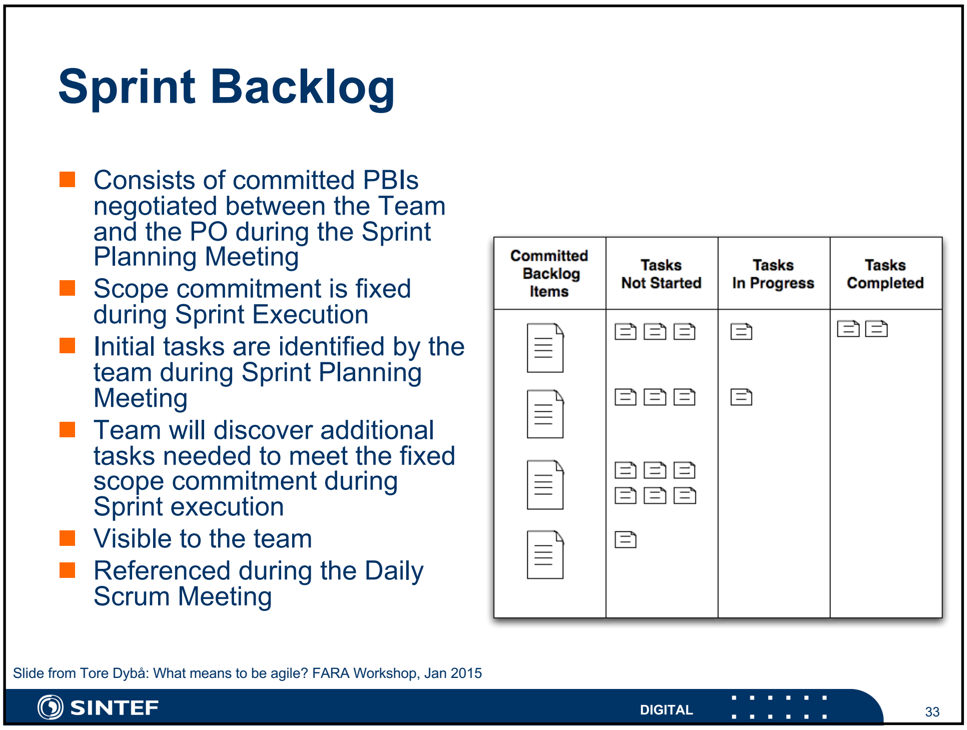


The product backlog

- The product backlog is the single most important artifact.
- It is a detailed analysis document, which outlines every requirement for a system, project, or product.
- It could be described as a comprehensive to-do list, expressed in priority order based on the business value each piece of work will generate.
- The scrum backlog is the engine of the business; it breaks the big-picture story down into manageable increments of work called Product Backlog Items (PBIs).

Slide from Tore Dybå: What means to be agile? FARA Workshop, Jan 2015

32

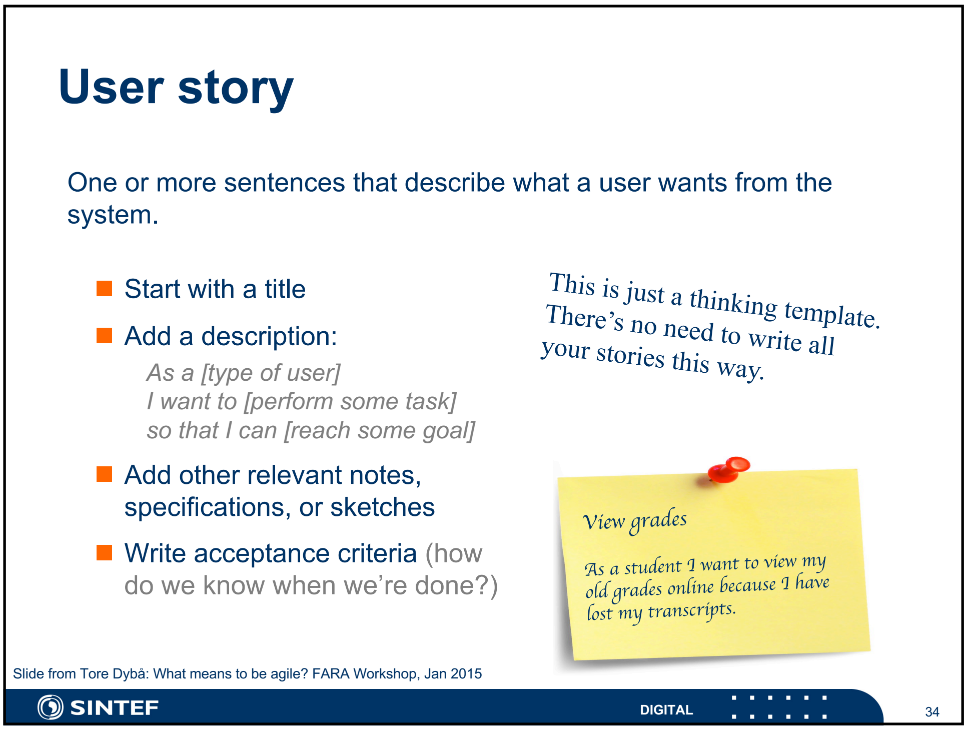


Sprint Backlog

- Consists of committed PBIs negotiated between the Team and the PO during the Sprint Planning Meeting
- Scope commitment is fixed during Sprint Execution
- Initial tasks are identified by the team during Sprint Planning Meeting
- Team will discover additional tasks needed to meet the fixed scope commitment during Sprint execution
- Visible to the team
- Referenced during the Daily Scrum Meeting

Committed Backlog Items	Tasks Not Started	Tasks In Progress	Tasks Completed
▤	▤ ▤ ▤	▤	▤ ▤
▤	▤ ▤ ▤	▤	
▤	▤ ▤ ▤ ▤ ▤ ▤		
▤	▤		

Slide from Tore Dybå: What means to be agile? FARA Workshop, Jan 2015



User story

One or more sentences that describe what a user wants from the system.

- Start with a title
- Add a description:

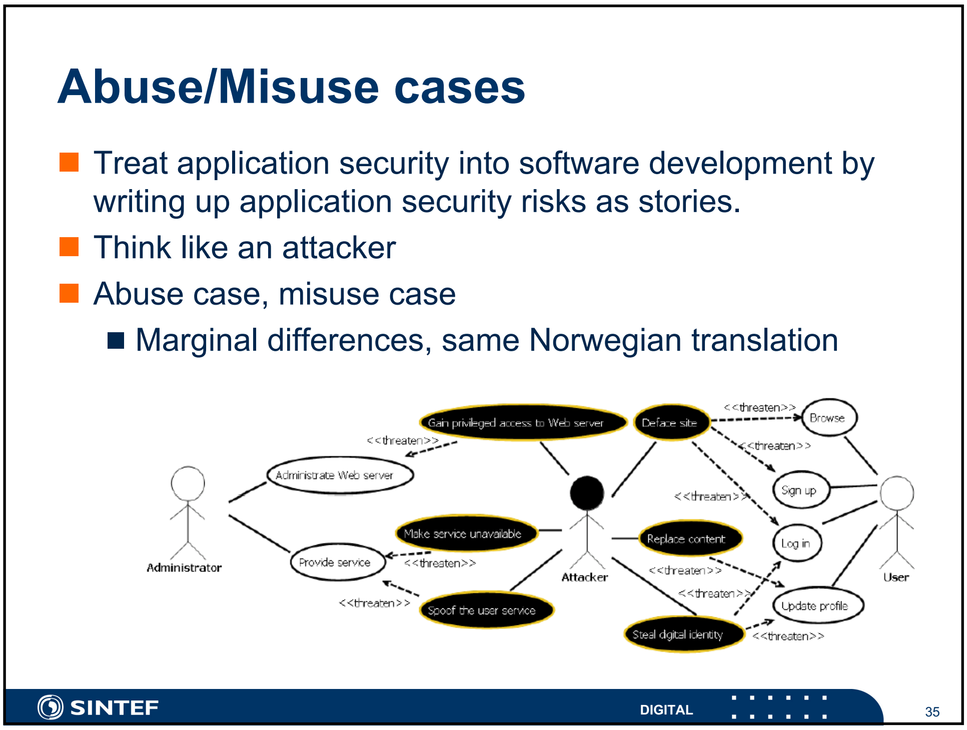
  As a [type of user]
  I want to [perform some task]
  so that I can [reach some goal]
- Add other relevant notes, specifications, or sketches
- Write acceptance criteria (how do we know when we're done?)

This is just a thinking template. There's no need to write all your stories this way.

View grades

As a student I want to view my old grades online because I have lost my transcripts.

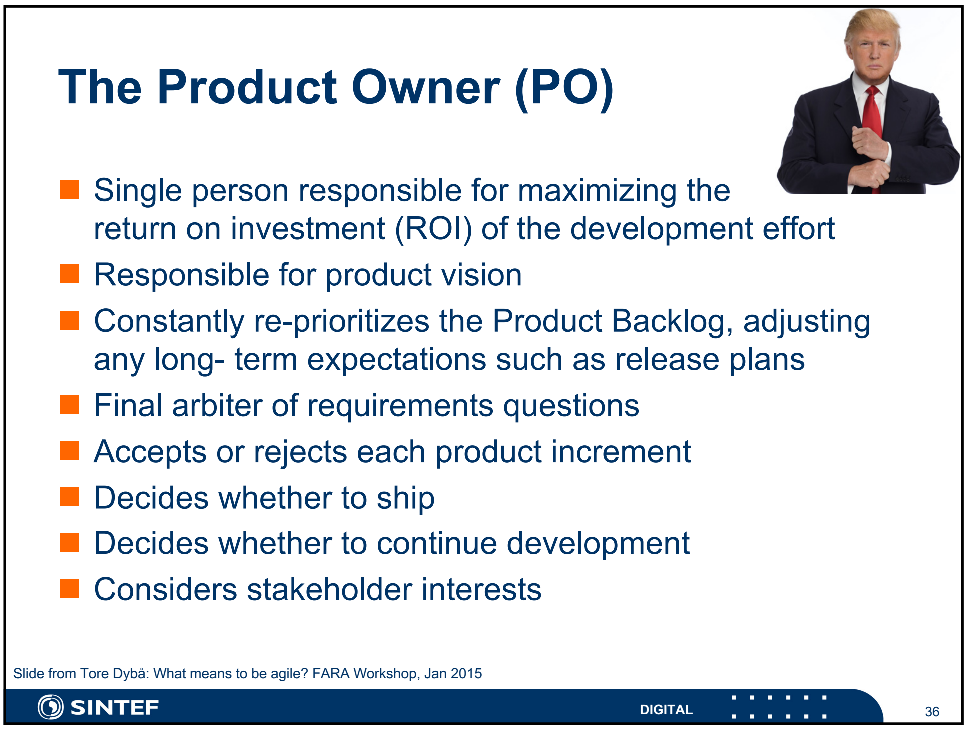
Slide from Tore Dybå: What means to be agile? FARA Workshop, Jan 2015



Abuse/Misuse cases

- Treat application security into software development by writing up application security risks as stories.
- Think like an attacker
- Abuse case, misuse case
  - Marginal differences, same Norwegian translation

SINTEF DIGITAL 35

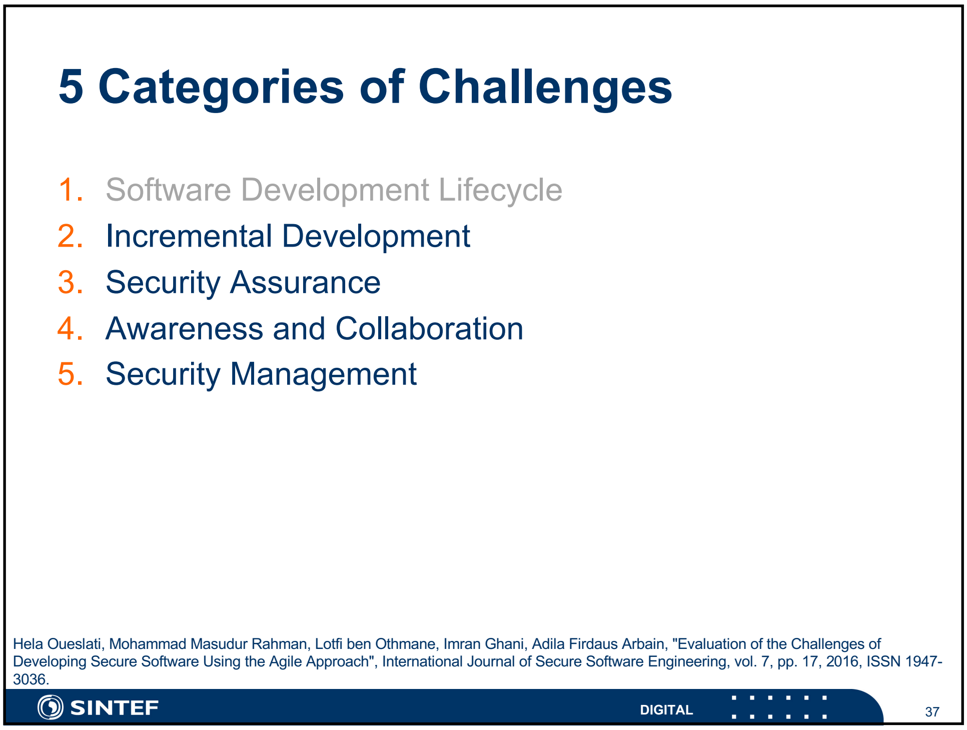


The Product Owner (PO)

- Single person responsible for maximizing the return on investment (ROI) of the development effort
- Responsible for product vision
- Constantly re-prioritizes the Product Backlog, adjusting any long- term expectations such as release plans
- Final arbiter of requirements questions
- Accepts or rejects each product increment
- Decides whether to ship
- Decides whether to continue development
- Considers stakeholder interests

Slide from Tore Dybå: What means to be agile? FARA Workshop, Jan 2015

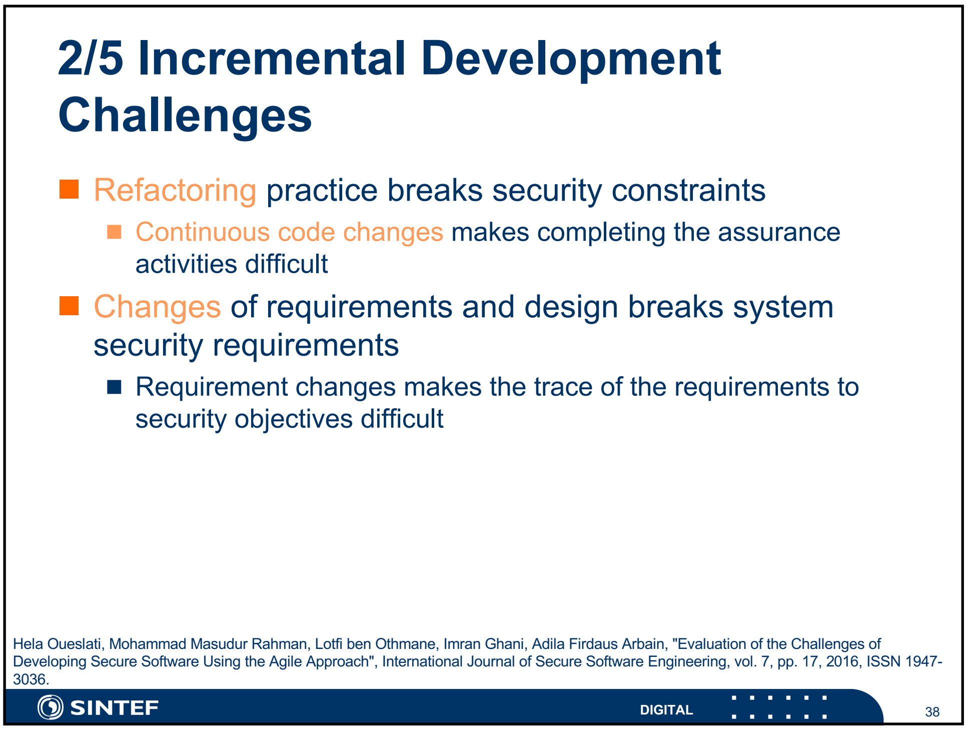
SINTEF DIGITAL 36



5 Categories of Challenges

1. Software Development Lifecycle
2. Incremental Development
3. Security Assurance
4. Awareness and Collaboration
5. Security Management

Hela Oueslati, Mohammad Masudur Rahman, Lotfi ben Othmane, Imran Ghani, Adila Firdaus Arbain, "Evaluation of the Challenges of Developing Secure Software Using the Agile Approach", International Journal of Secure Software Engineering, vol. 7, pp. 17, 2016, ISSN 1947-3036.

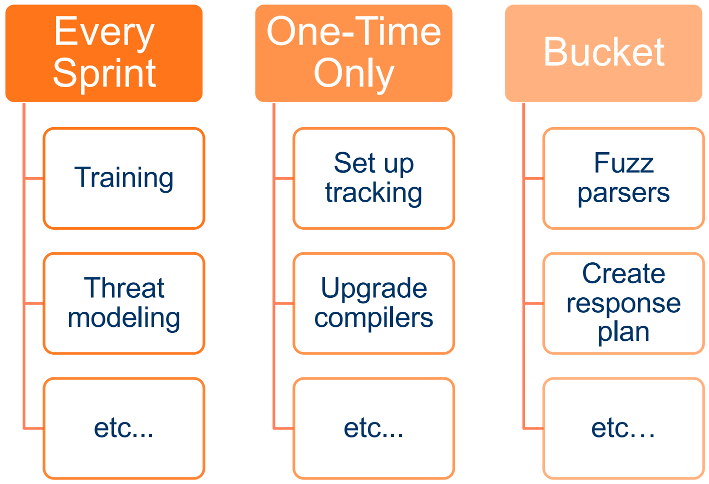


2/5 Incremental Development Challenges

- Refactoring practice breaks security constraints
  - Continuous code changes makes completing the assurance activities difficult
- Changes of requirements and design breaks system security requirements
  - Requirement changes makes the trace of the requirements to security objectives difficult

Hela Oueslati, Mohammad Masudur Rahman, Lotfi ben Othmane, Imran Ghani, Adila Firdaus Arbain, "Evaluation of the Challenges of Developing Secure Software Using the Agile Approach", International Journal of Secure Software Engineering, vol. 7, pp. 17, 2016, ISSN 1947-3036.

Three classes of requirements (Microsoft Agile SDL)



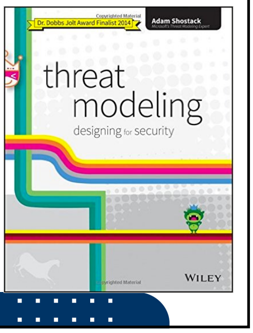
Every Sprint
- Training
- Threat modeling
- etc...

One-Time Only
- Set up tracking
- Upgrade compilers
- etc...

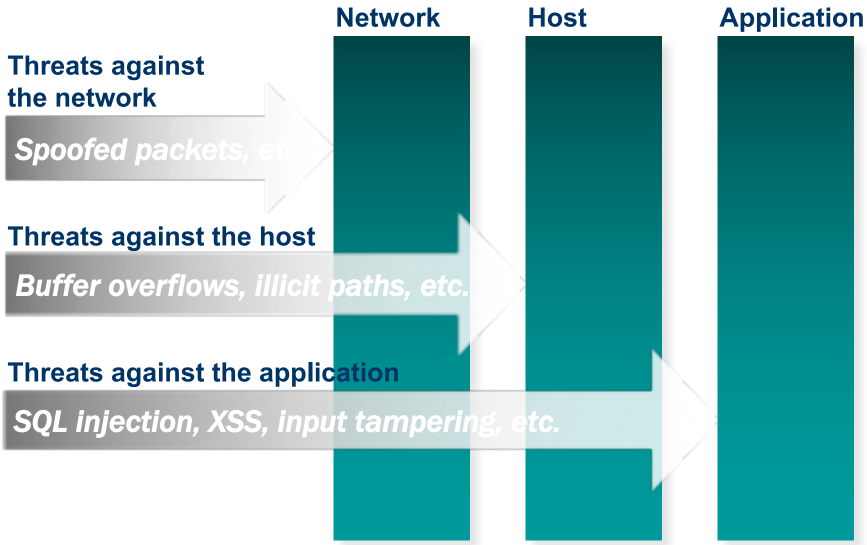
Bucket
- Fuzz parsers
- Create response plan
- etc…

Threat Modeling

- Structured approach to identifying, quantifying, and addressing threats
- Essential part of development process
  - Just like specifying and designing
  - Just like coding and testing
- Many techniques and Tools (e.g., STRIDE/element, OCTAVE)



Types of Threats



Threats against the network

Spoofed packets, etc.

Threats against the host

Buffer overflows, illicit paths, etc.

Threats against the application

SQL injection, XSS, input tampering, etc.

Threats Against the Application

SCFM
Secure Coding Field Manual
A Programmer's Guide to OWASP Top 10 and CWE/SANS Top 25
Now Includes HTML5
SECURE CODE
Written by Sunny Wear, CISSP, CRISC, CEH, GIAC GSSP-JAVA, CSSLP

Threat	Examples
SQL injectio	TABLE command in text typed
Cross-site scr	t-side script to steal cookies
Hidden-field tampering	he value of a hidden field
Eavesdropping	steal passwords and encrypted connections
Session hijacking	ookie to access e
Identity spoofing	cation cookie to pose
Information disclosure	A see a stack trace when an u exception occurs

Static Analysis Tools

- A white box testing approach
  - Passive scanning of application code without executing it
- Analyzing software “at rest” –
- Source code
  – Bytecode
  – Binary
  - Results can be in a report form
  - May be integrated into IDEs
  - May be integrated into CI/CD environments

Code Review

- Manual
  - Time consuming
  - Dependent on expertise
- Automated
  - Many static analysis tools on the market
  - Can look for known bugs
  - Remember: Can prove that there are bugs in the code, but can never prove that it is bug-free

Illustration by Fraser Speirs, CC BY 2.0

5 Categories of Challenges

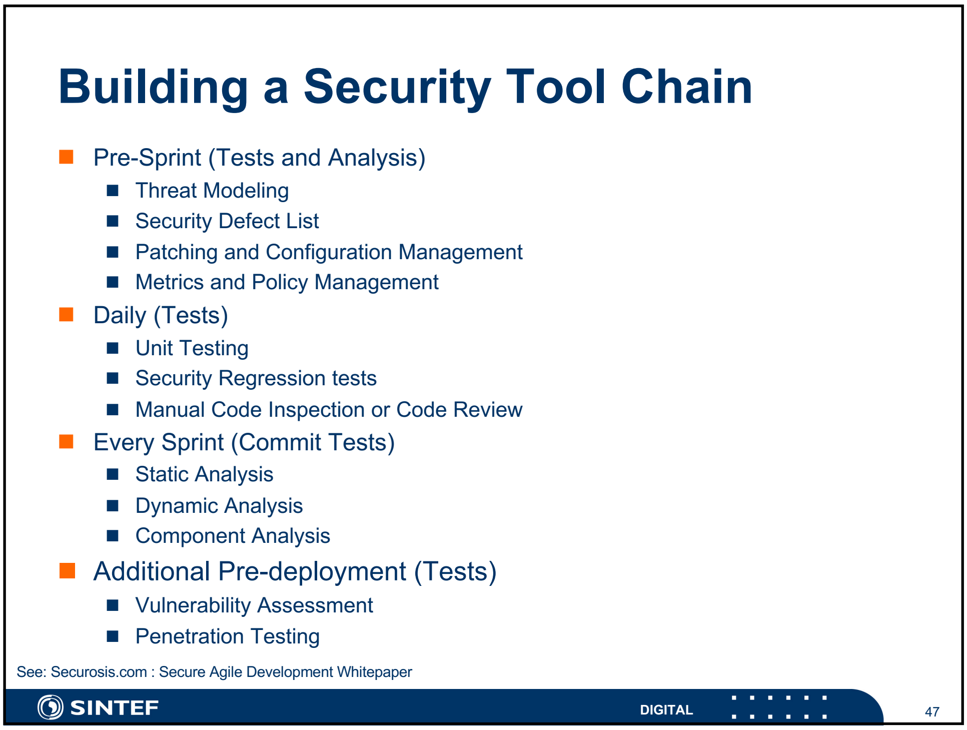
1. Software Development Lifecycle
2. Incremental Development
3. Security Assurance
4. Awareness and Collaboration
5. Security Management

Hela Oueslati, Mohammad Masudur Rahman, Lotfi ben Othmane, Imran Ghani, Adila Firdaus Arbain, "Evaluation of the Challenges of Developing Secure Software Using the Agile Approach", International Journal of Secure Software Engineering, vol. 7, pp. 17, 2016, ISSN 1947-

3/5 Security Assurance Challenges

- Agile Means Constant Transition and Every transition is a risk!
- Tests are, in general, insufficient to ensure the implementation of security requirements
- Tests do not cover in general, all vulnerability cases
- Security tests are in general difficult to automate
- Continuous changing of the development processes (to support lessons learned) conflicts with audit needs of uniform stable processes.

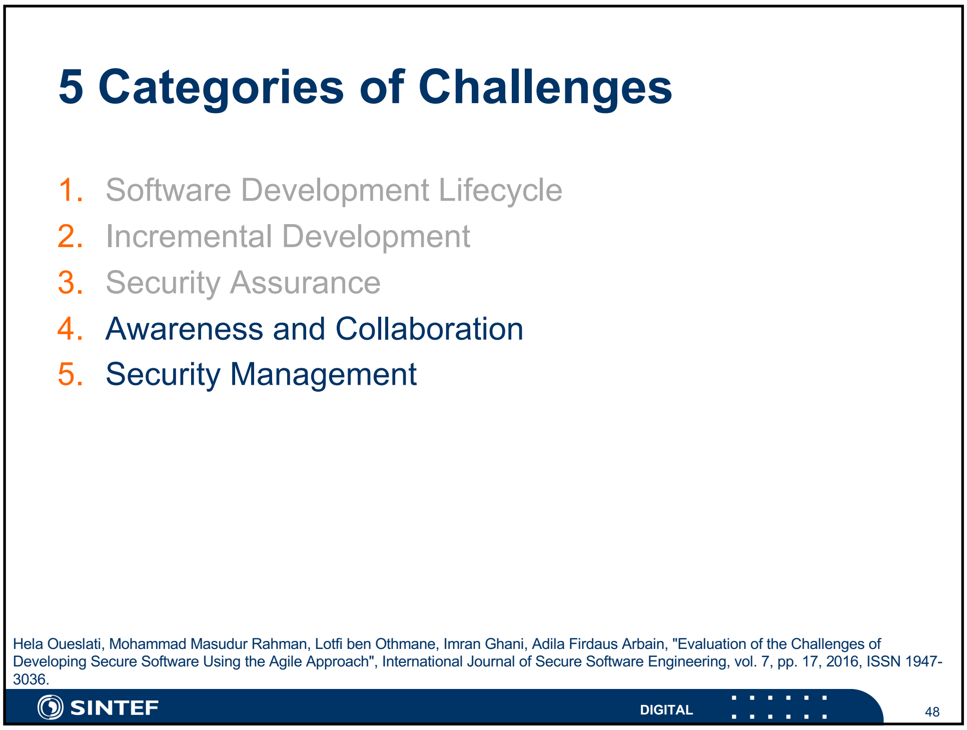
Hela Oueslati, Mohammad Masudur Rahman, Lotfi ben Othmane, Imran Ghani, Adila Firdaus Arbain, "Evaluation of the Challenges of Developing Secure Software Using the Agile Approach", International Journal of Secure Software Engineering, vol. 7, pp. 17, 2016, ISSN 1947-3036.



Building a Security Tool Chain

- Pre-Sprint (Tests and Analysis)
  - Threat Modeling
  - Security Defect List
  - Patching and Configuration Management
  - Metrics and Policy Management
- Daily (Tests)
  - Unit Testing
  - Security Regression tests
  - Manual Code Inspection or Code Review
- Every Sprint (Commit Tests)
  - Static Analysis
  - Dynamic Analysis
  - Component Analysis
- Additional Pre-deployment (Tests)
  - Vulnerability Assessment
  - Penetration Testing

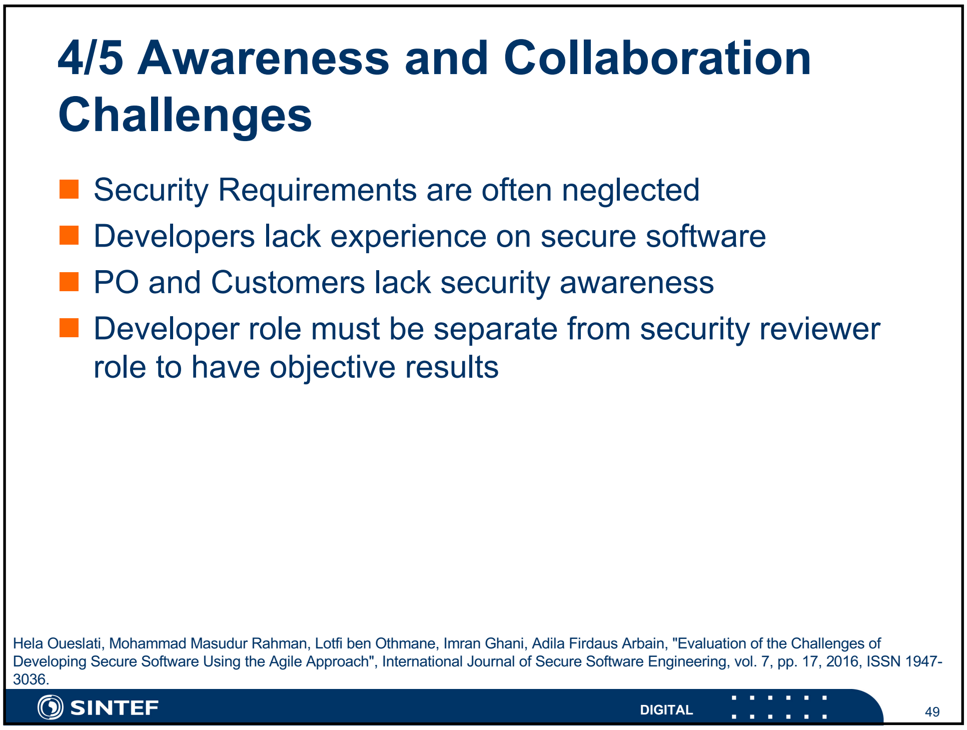
See: Securosis.com : Secure Agile Development Whitepaper



5 Categories of Challenges

1. Software Development Lifecycle
2. Incremental Development
3. Security Assurance
4. Awareness and Collaboration
5. Security Management

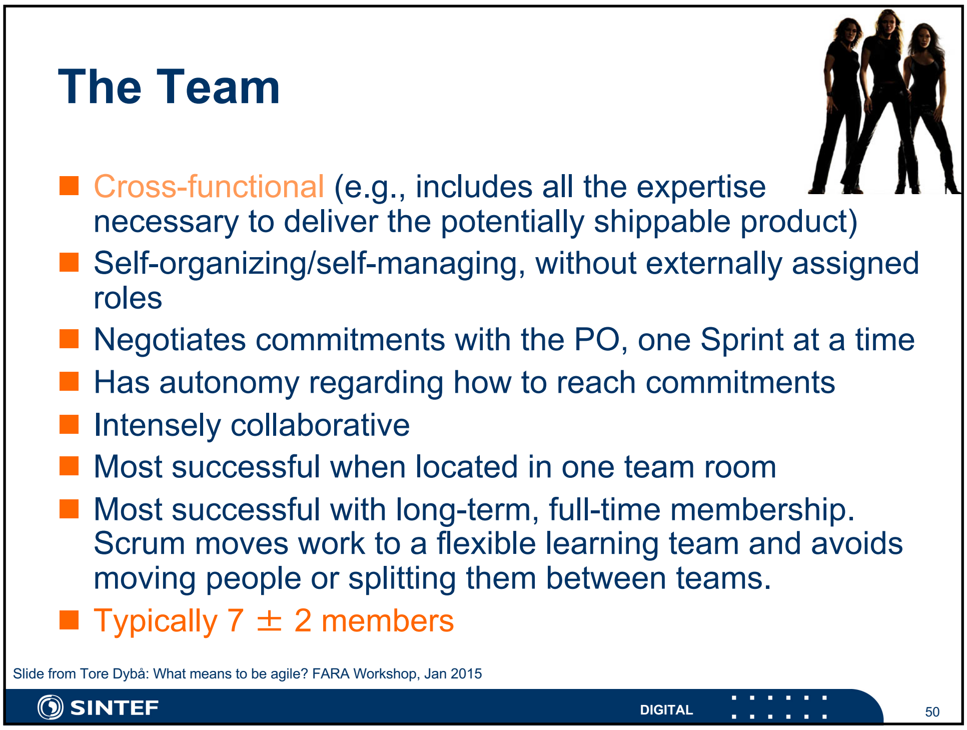
Hela Oueslati, Mohammad Masudur Rahman, Lotfi ben Othmane, Imran Ghani, Adila Firdaus Arbain, "Evaluation of the Challenges of Developing Secure Software Using the Agile Approach", International Journal of Secure Software Engineering, vol. 7, pp. 17, 2016, ISSN 1947-3036.



4/5 Awareness and Collaboration Challenges

- Security Requirements are often neglected
- Developers lack experience on secure software
- PO and Customers lack security awareness
- Developer role must be separate from security reviewer role to have objective results

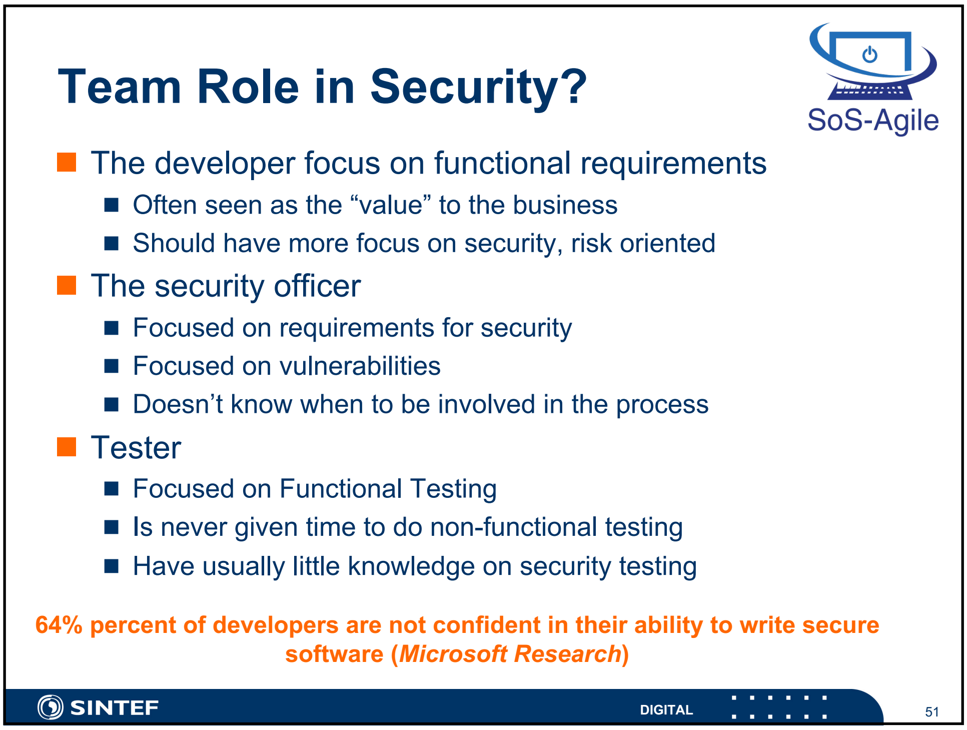
Hela Oueslati, Mohammad Masudur Rahman, Lotfi ben Othmane, Imran Ghani, Adila Firdaus Arbain, "Evaluation of the Challenges of Developing Secure Software Using the Agile Approach", International Journal of Secure Software Engineering, vol. 7, pp. 17, 2016, ISSN 1947-3036.



The Team

- Cross-functional (e.g., includes all the expertise necessary to deliver the potentially shippable product)
- Self-organizing/self-managing, without externally assigned roles
- Negotiates commitments with the PO, one Sprint at a time
- Has autonomy regarding how to reach commitments
- Intensely collaborative
- Most successful when located in one team room
- Most successful with long-term, full-time membership. Scrum moves work to a flexible learning team and avoids moving people or splitting them between teams.
- Typically 7 ± 2 members

Slide from Tore Dybå: What means to be agile? FARA Workshop, Jan 2015

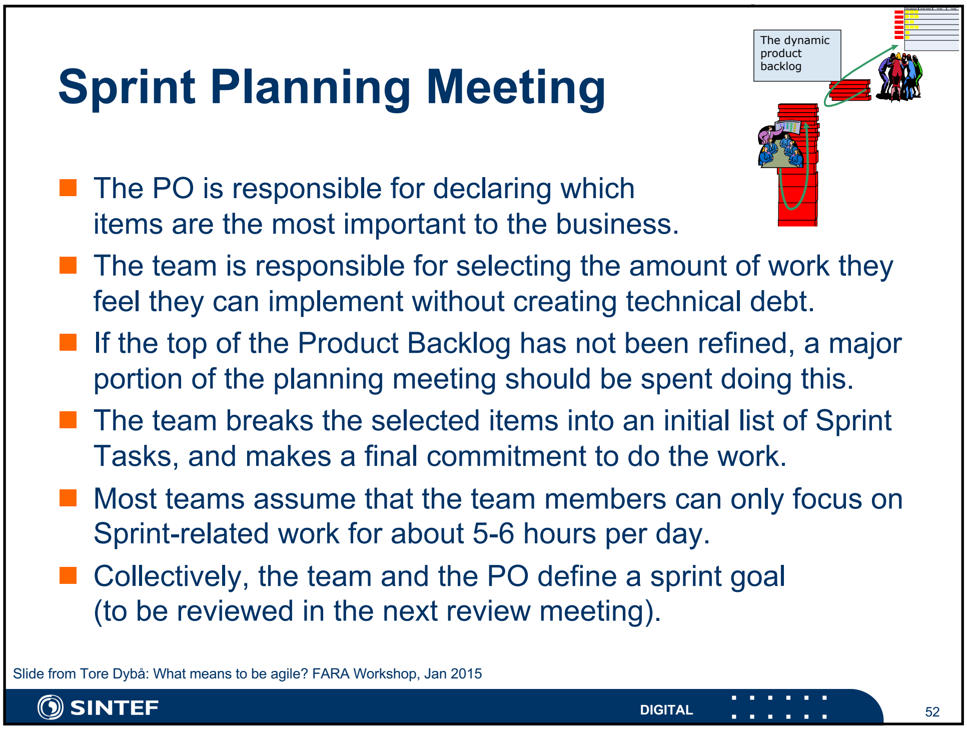


Team Role in Security?

SoS-Agile

- The developer focus on functional requirements
  - Often seen as the “value” to the business
  - Should have more focus on security, risk oriented
- The security officer
  - Focused on requirements for security
  - Focused on vulnerabilities
  - Doesn’t know when to be involved in the process
- Tester
  - Focused on Functional Testing
  - Is never given time to do non-functional testing
  - Have usually little knowledge on security testing

64% percent of developers are not confident in their ability to write secure software (*Microsoft Research*)

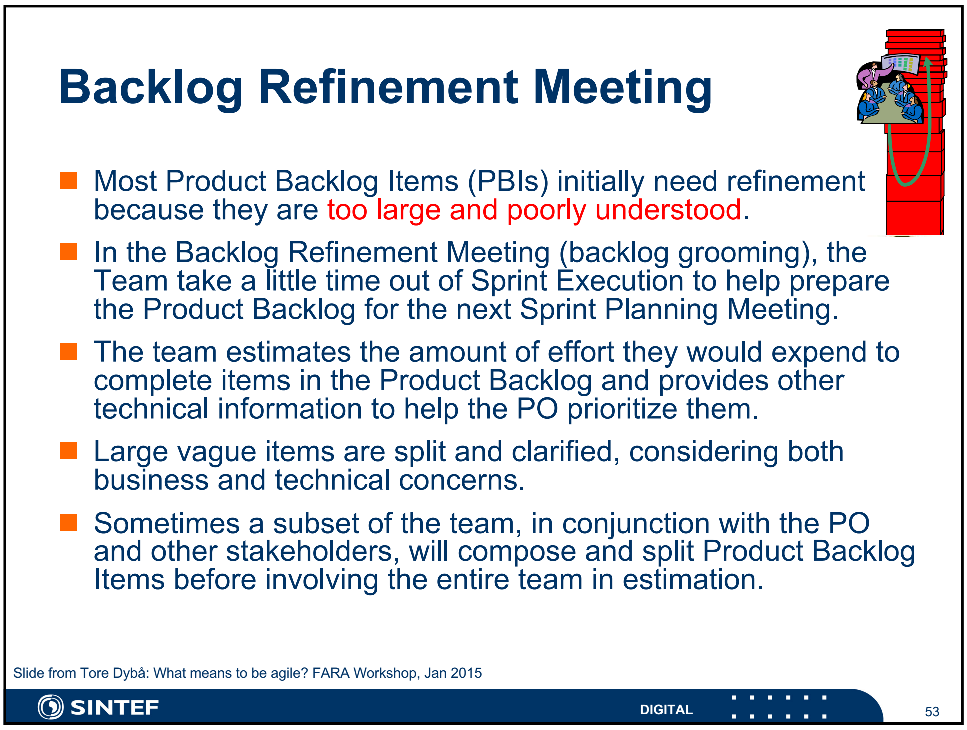


The dynamic product backlog

Sprint Planning Meeting

- The PO is responsible for declaring which items are the most important to the business.
- The team is responsible for selecting the amount of work they feel they can implement without creating technical debt.
- If the top of the Product Backlog has not been refined, a major portion of the planning meeting should be spent doing this.
- The team breaks the selected items into an initial list of Sprint Tasks, and makes a final commitment to do the work.
- Most teams assume that the team members can only focus on Sprint-related work for about 5-6 hours per day.
- Collectively, the team and the PO define a sprint goal (to be reviewed in the next review meeting).

Slide from Tore Dybå: What means to be agile? FARA Workshop, Jan 2015

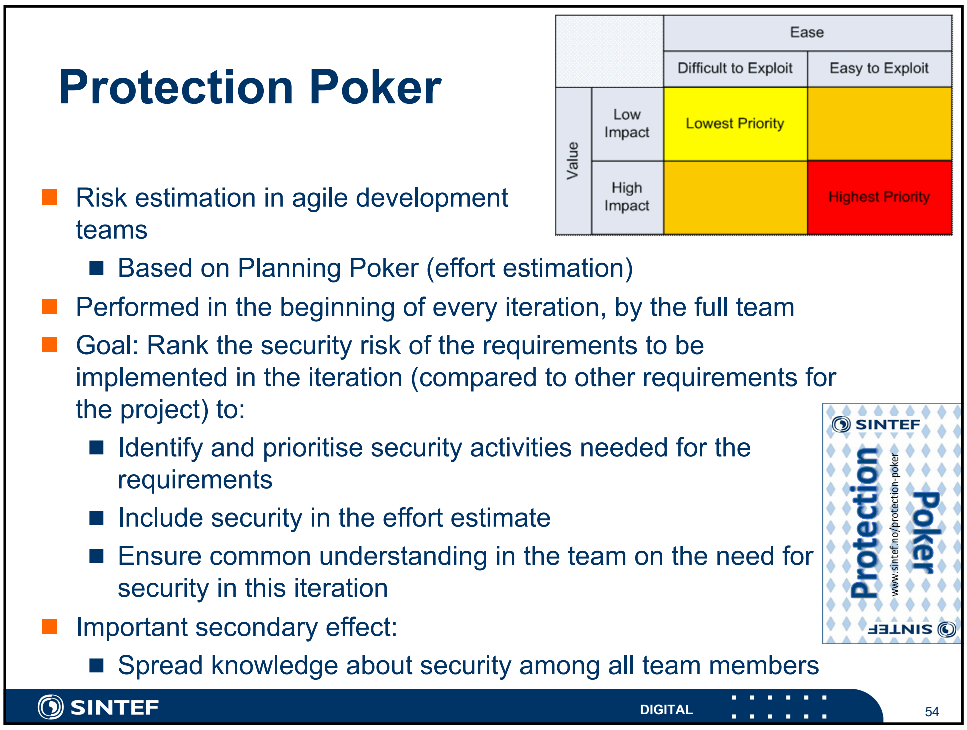


Backlog Refinement Meeting

- Most Product Backlog Items (PBIs) initially need refinement because they are too large and poorly understood.
- In the Backlog Refinement Meeting (backlog grooming), the Team take a little time out of Sprint Execution to help prepare the Product Backlog for the next Sprint Planning Meeting.
- The team estimates the amount of effort they would expend to complete items in the Product Backlog and provides other technical information to help the PO prioritize them.
- Large vague items are split and clarified, considering both business and technical concerns.
- Sometimes a subset of the team, in conjunction with the PO and other stakeholders, will compose and split Product Backlog Items before involving the entire team in estimation.

Slide from Tore Dybå: What means to be agile? FARA Workshop, Jan 2015

SINTEF DIGITAL 53



Protection Poker

		Ease	
		Difficult to Exploit	Easy to Exploit
Value	Low Impact	Lowest Priority	🟧
	High Impact	🟧	Highest Priority

- Risk estimation in agile development teams
  - Based on Planning Poker (effort estimation)
- Performed in the beginning of every iteration, by the full team
- Goal: Rank the security risk of the requirements to be implemented in the iteration (compared to other requirements for the project) to:
  - Identify and prioritise security activities needed for the requirements
  - Include security in the effort estimate
  - Ensure common understanding in the team on the need for security in this iteration
- Important secondary effect:
  - Spread knowledge about security among all team members

SINTEF Protection Poker www.sintef.no/protection-poker SINTEF

SINTEF DIGITAL 54

Daily Scrum and Sprint Execution

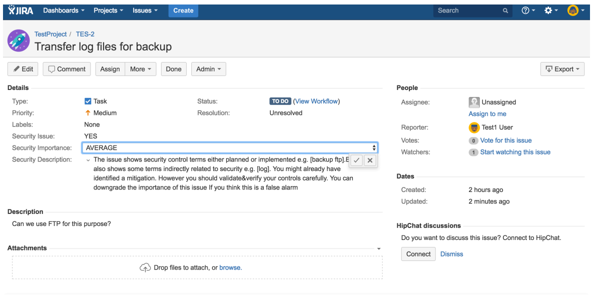
- Every day at the same time and place, the Team members spend about 15 minutes reporting to each other
  - What did you do yesterday? What will you do today? Are there any impediments in your way?
- Standing up at the Daily Scrum will help keep it short. Topics that require additional attention may be discussed by whomever is interested after every team member has reported.
- The team may find it useful to maintain a current Sprint Task List, a Sprint Burndown Chart, and an Impediments List.
- Any impediments that are raised in the scrum meeting become the Scrum Master's responsibility to resolve as quickly as possible.

Slide from Tore Dybå: What means to be agile? FARA Workshop, Jan 2015

JiraSecPlugin

SoS-Agile

- A Plugin to Classify and Rank Security Related Issues Classify Issue as security related or not
  - Report the importance of the classification
  - Provide a feedback (Message)
  - Support for continuous deployment
  - Create Awareness



TestProject / TES-2

Transfer log files for backup

Details

Type: ☑ Task
Status: TO DO (View Workflow)
Priority: ↑ Medium
Resolution: Unresolved
Labels: None
Security Issue: YES
Security Importance: AVERAGE
Security Description: The issue shows security control terms either planned or implemented e.g. [backup ftp].E also shows some terms indirectly related to security e.g. [log]. You might already have identified a mitigation. However you should validate&verify your controls carefully. You can downgrade the importance of this issue If you think this is a false alarm

Description

Can we use FTP for this purpose?

Attachments

Drop files to attach, or browse.

People

Assignee: Unassigned
Assign to me
Reporter: Test1 User
Votes: 0 Vote for this issue
Watchers: 1 Start watching this issue

Dates

Created: 2 hours ago
Updated: 2 minutes ago

HipChat discussions

Do you want to discuss this issue? Connect to HipChat.

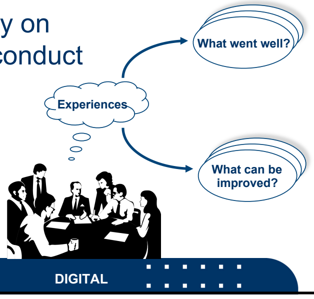
Connect Dismiss

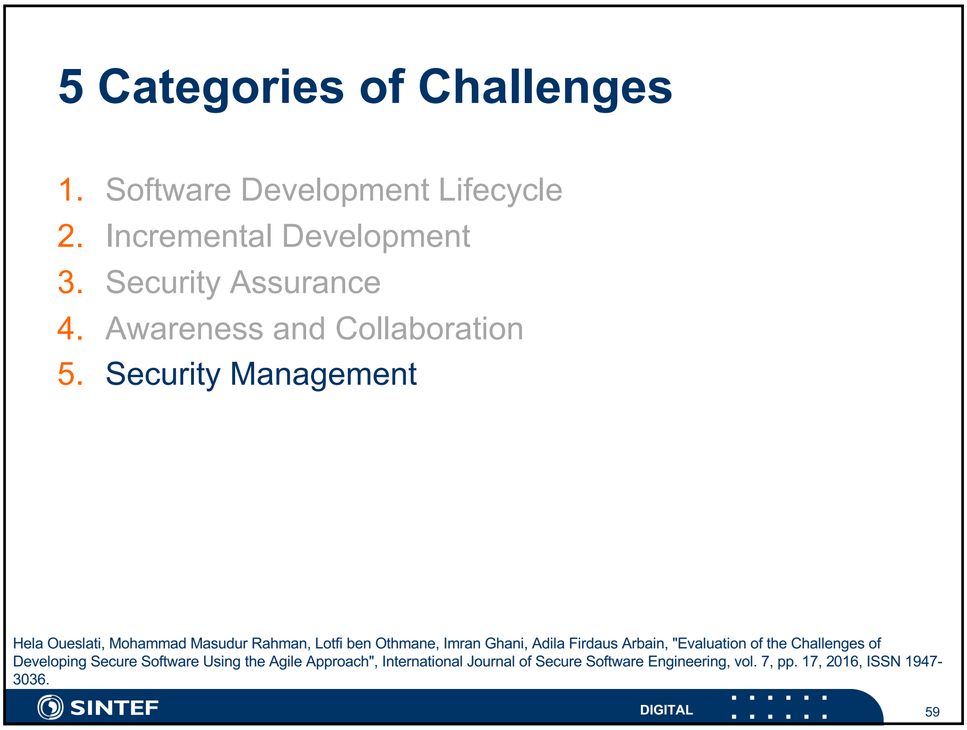
Sprint Review Meeting

- The team holds a Sprint Review Meeting to demonstrate a working product increment to the PO and everyone else who is interested.
- The meeting should feature a live demonstration, not a report.
- After the demonstration, the PO reviews the commitments made at the Sprint Planning Meeting and declares which items she/he now considers *done*.
- Incomplete items are returned to the Product Backlog and ranked according to the PO's revised priorities as candidates for future Sprints.
- It is the opportunity to inspect and adapt the product as it emerges, and iteratively refine everyone's understanding of the requirements.
- New products, particularly software products, are hard to visualize in a vacuum. Many customers need to be able to react to a piece of functioning software to discover what they will actually want.

Sprint Retrospective Meeting

- Each Sprint ends with a retrospective. At this meeting, the Team reflects on its own process. They inspect their behavior and take action to adapt it for future Sprints.
- An in-depth retrospective requires an environment of psychological safety not found in most organizations.
- Without safety, the retrospective discussion will either avoid the uncomfortable issues or deteriorate into blaming and hostility.
- A common impediment to full transparency on the Team is the presence of people who conduct performance appraisals.



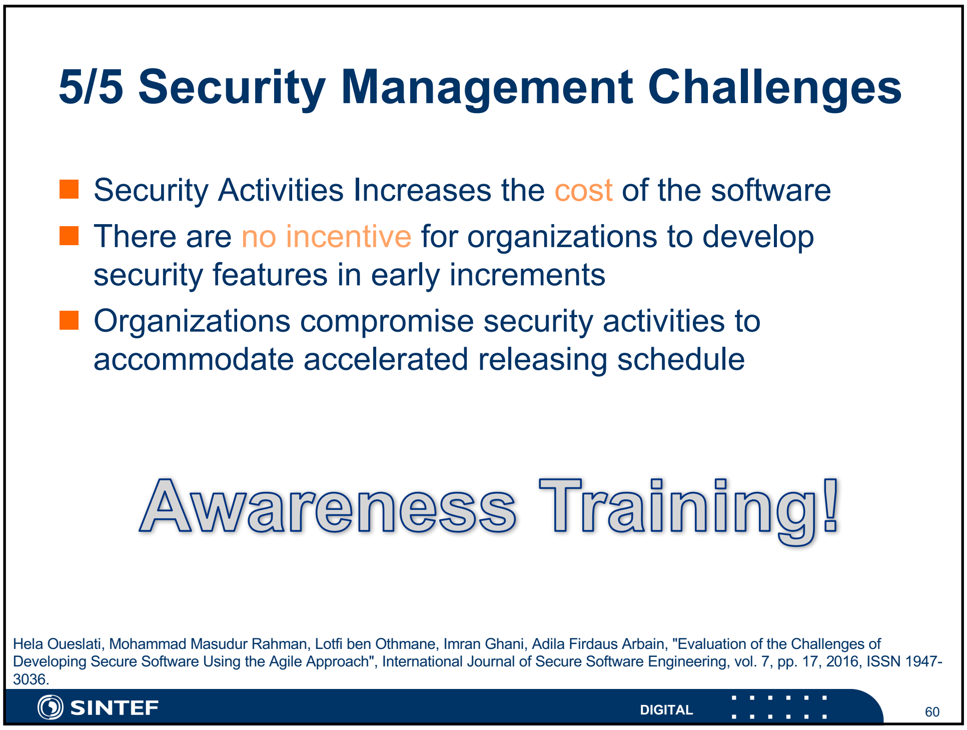


5 Categories of Challenges

1. Software Development Lifecycle
2. Incremental Development
3. Security Assurance
4. Awareness and Collaboration
5. Security Management

Hela Oueslati, Mohammad Masudur Rahman, Lotfi ben Othmane, Imran Ghani, Adila Firdaus Arbain, "Evaluation of the Challenges of Developing Secure Software Using the Agile Approach", International Journal of Secure Software Engineering, vol. 7, pp. 17, 2016, ISSN 1947-3036.

SINTEF DIGITAL 59



5/5 Security Management Challenges

- Security Activities Increases the cost of the software
- There are no incentive for organizations to develop security features in early increments
- Organizations compromise security activities to accommodate accelerated releasing schedule

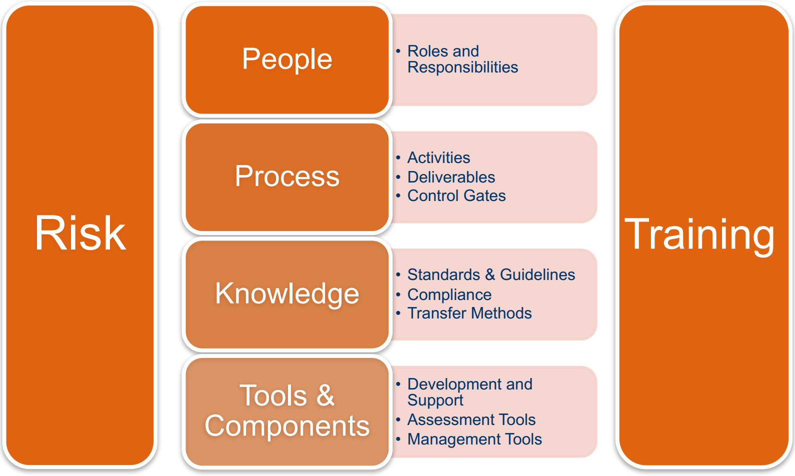
Awareness Training!

Hela Oueslati, Mohammad Masudur Rahman, Lotfi ben Othmane, Imran Ghani, Adila Firdaus Arbain, "Evaluation of the Challenges of Developing Secure Software Using the Agile Approach", International Journal of Secure Software Engineering, vol. 7, pp. 17, 2016, ISSN 1947-3036.

SINTEF DIGITAL 60

Agile SDLs

SDLC Cornerstones



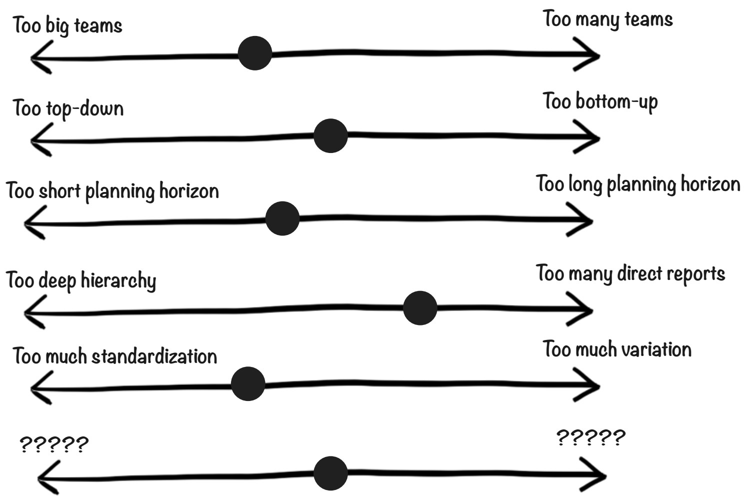
- Roles and Responsibilities

- Activities
- Deliverables
- Control Gates

- Standards & Guidelines
- Compliance
- Transfer Methods

- Development and Support
- Assessment Tools
- Management Tools

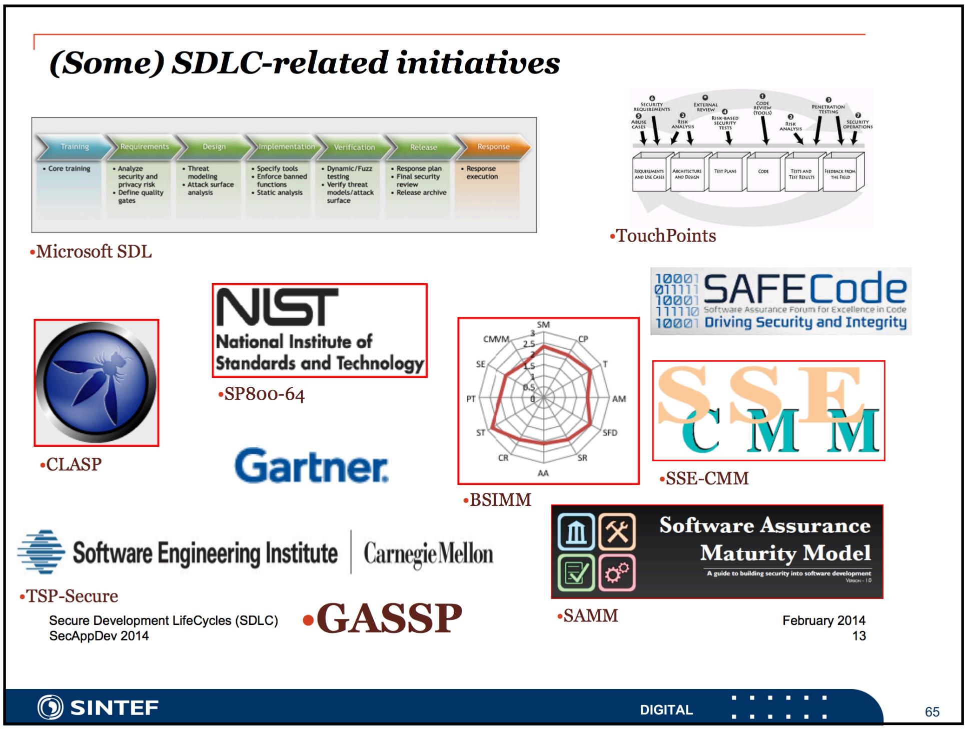
There's no right or wrong. It's all tradeoffs!



Henrik Kniberg, What is Scrum?

Agile thinking

Pick the right solutions based on your needs!



(Some) SDLC-related initiatives

•Microsoft SDL

•TouchPoints

•CLASP

•SP800-64

•BSIMM

•SSE-CMM

•TSP-Secure

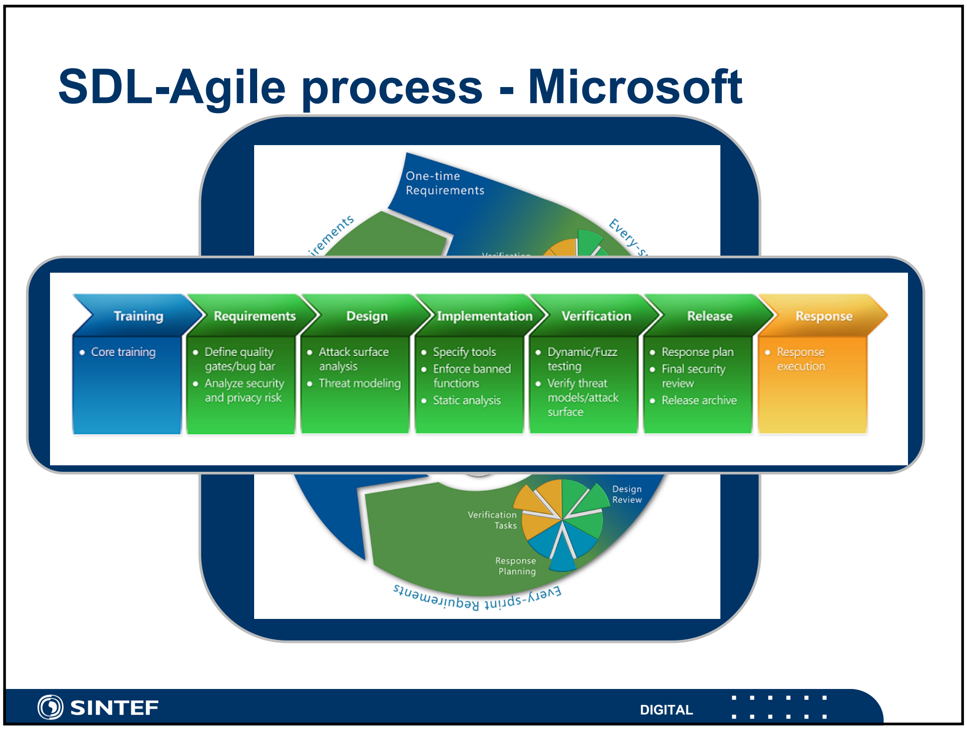
•GASSP

•SAMM

Secure Development LifeCycles (SDLC)
SecAppDev 2014

February 2014
13

SINTEF DIGITAL 65



SDL-Agile process - Microsoft

Training
- Core training

Requirements
- Define quality gates/bug bar
- Analyze security and privacy risk

Design
- Attack surface analysis
- Threat modeling

Implementation
- Specify tools
- Enforce banned functions
- Static analysis

Verification
- Dynamic/Fuzz testing
- Verify threat models/attack surface

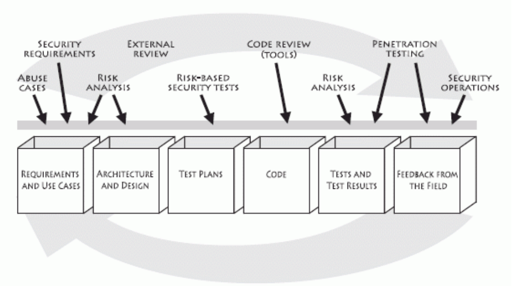
Release
- Response plan
- Final security review
- Release archive

Response
- Response execution

SINTEF DIGITAL

The Touchpoints

- Code review
- Architectural risk analysis
- Penetration testing
- Risk-based security tests
- Abuse cases
- Security requirements
- Security operations

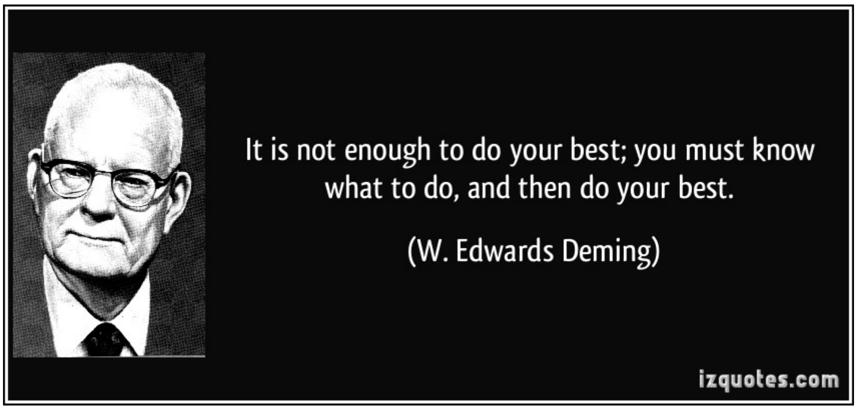


http://www.swsec.com/resources/touchpoints/

Automation : Tools and People

Illustration by Scott Lewis
https://www.flickr.com/photos/99781513@N04/15803129554/ CC BY 2.0

- **Automate what you can**
- **Knowledge on Security is needed before Tools can be useful!**



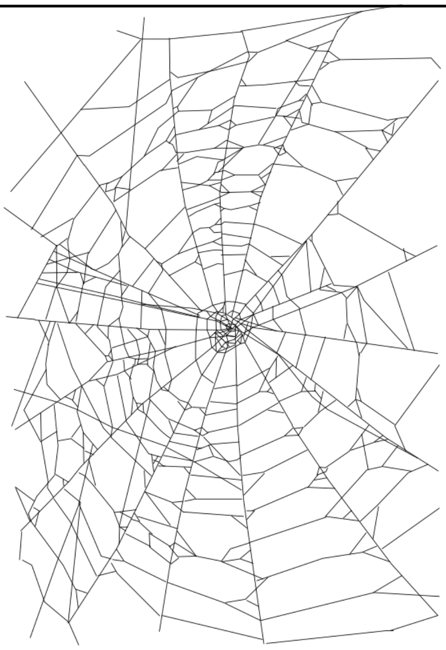
It is not enough to do your best; you must know what to do, and then do your best.

(W. Edwards Deming)

izquotes.com

Threat modeling

Inger Anne Tøndel
Research Scientist SINTEF ICT
PhD candidate NTNU

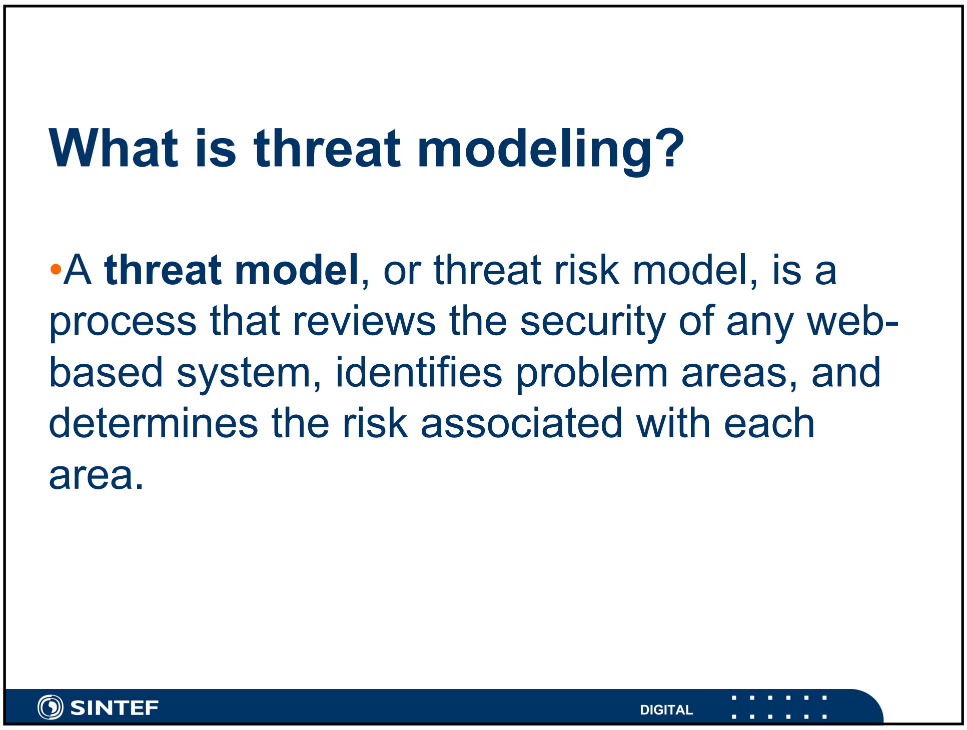


If we had our hands tied behind our backs … and could do only one thing to improve software security … we would do threat modeling

Michael Howard Steve Lipner

"The Security Development Lifecycle", Microsoft Press, 2006

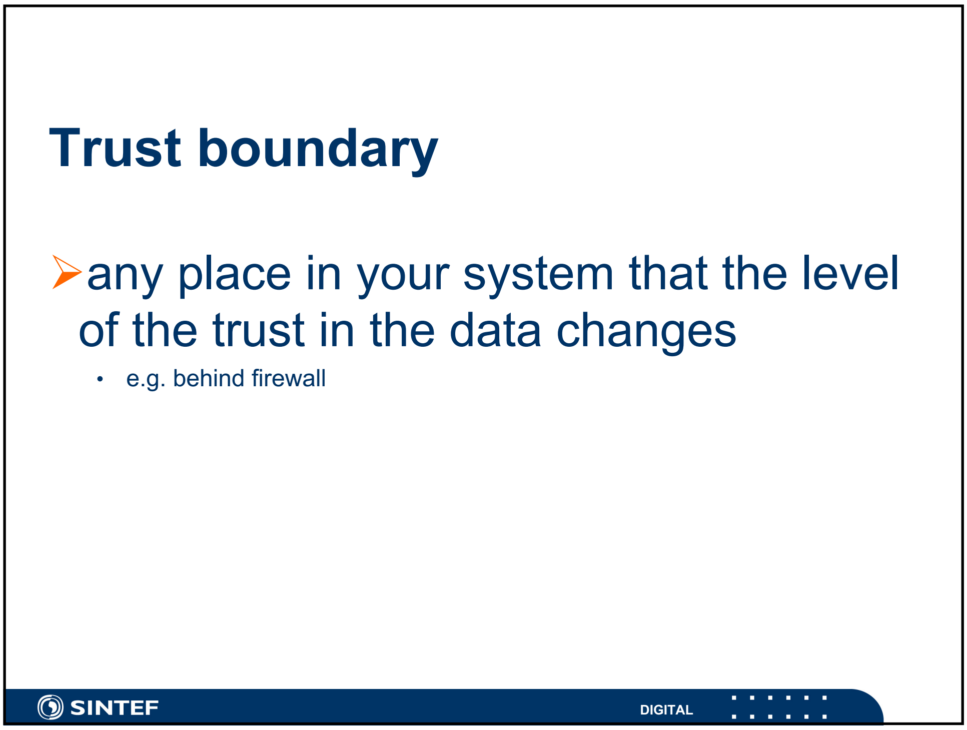
SINTEF DIGITAL



What is threat modeling?

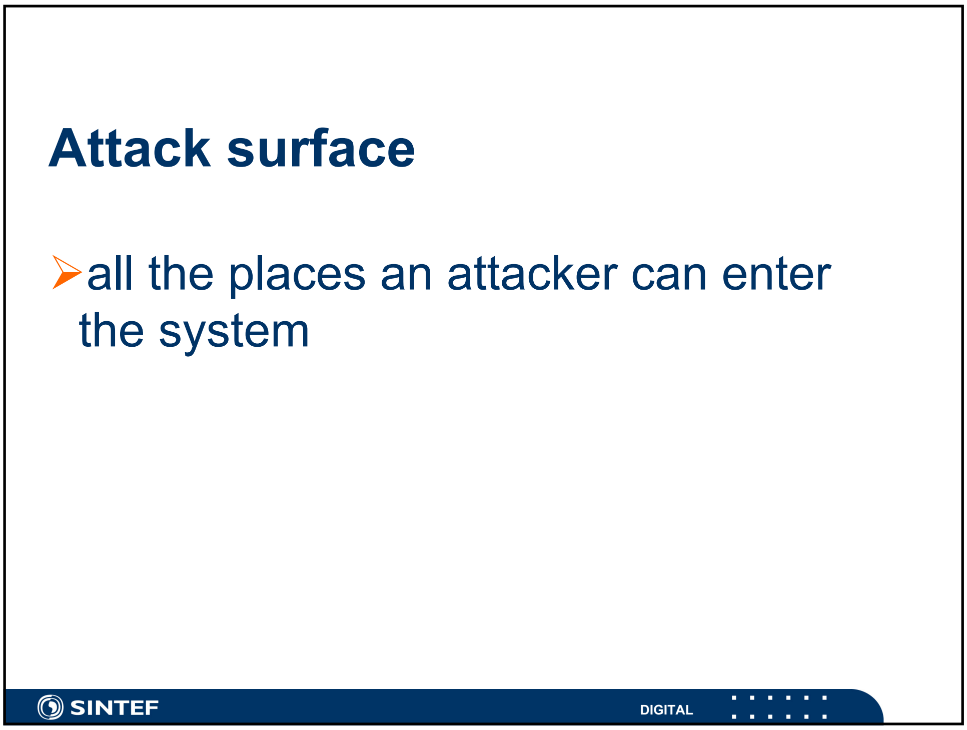
- A **threat model**, or threat risk model, is a process that reviews the security of any web-based system, identifies problem areas, and determines the risk associated with each area.

SINTEF DIGITAL



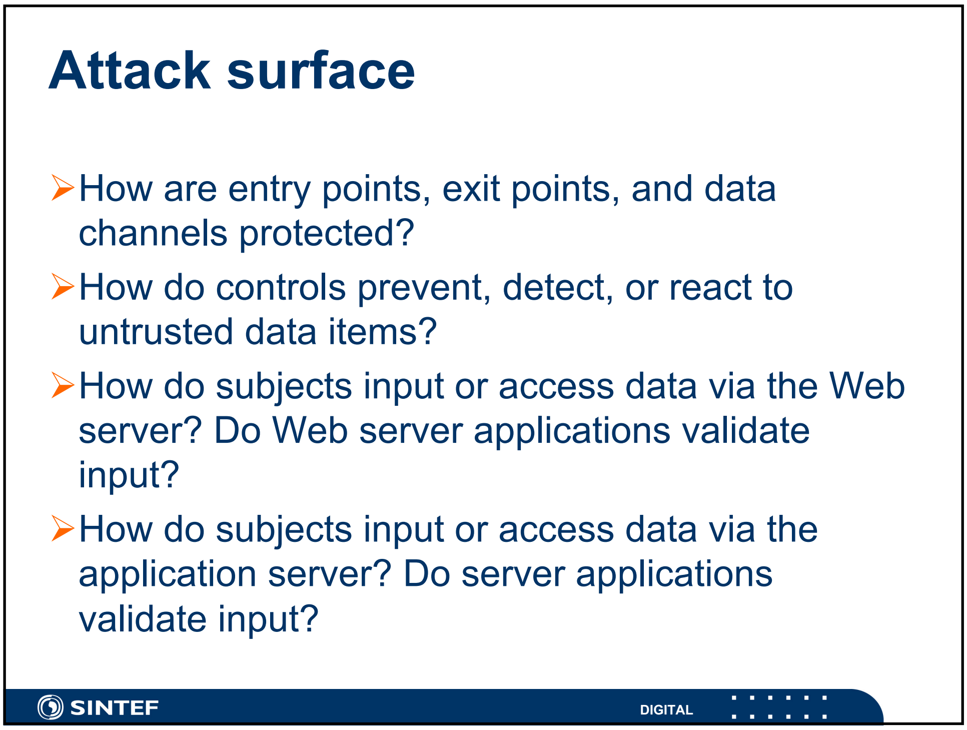
Trust boundary

- any place in your system that the level of the trust in the data changes
  - e.g. behind firewall



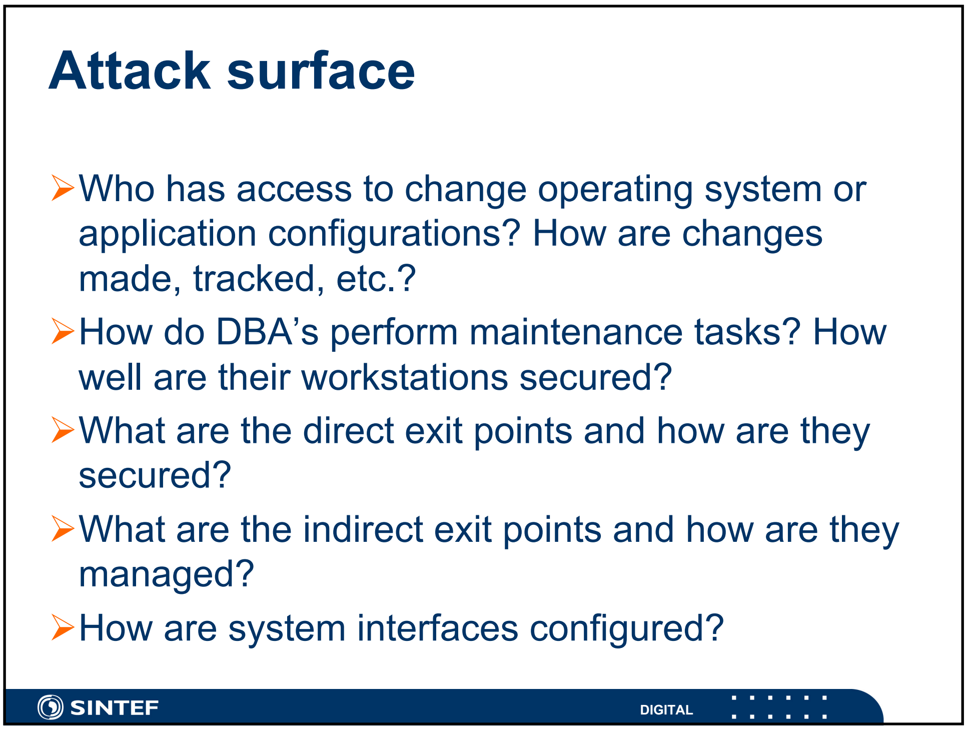
Attack surface

- all the places an attacker can enter the system



Attack surface

- How are entry points, exit points, and data channels protected?
- How do controls prevent, detect, or react to untrusted data items?
- How do subjects input or access data via the Web server? Do Web server applications validate input?
- How do subjects input or access data via the application server? Do server applications validate input?



Attack surface

- Who has access to change operating system or application configurations? How are changes made, tracked, etc.?
- How do DBA’s perform maintenance tasks? How well are their workstations secured?
- What are the direct exit points and how are they secured?
- What are the indirect exit points and how are they managed?
- How are system interfaces configured?

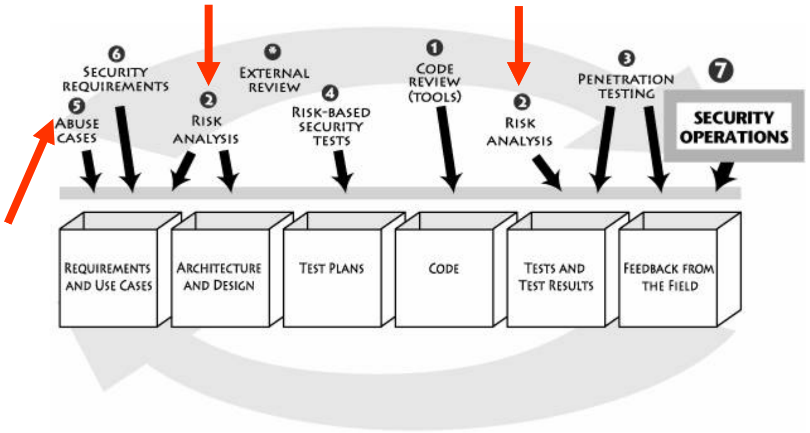
A Threat Model is….

- **A visual representation of four main elements:**
  - The assets within a system;
  - The system's attack surface;
  - A description of how the components and assets interact;
  - Threat actors who could attack the system and how the attack could occur;

- Think like an attacker -

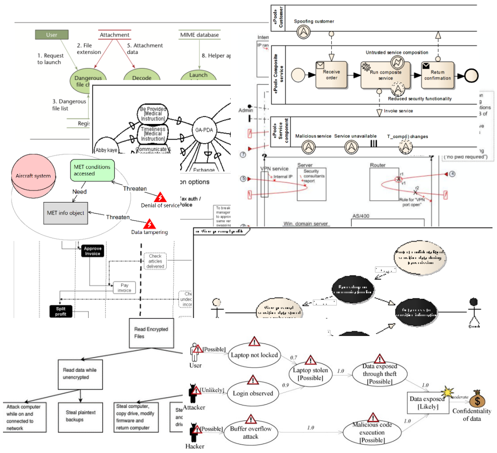
What is threat modeling input to?

Software Security Touchpoints (McGraw)



Source: http://www.cigital.com/justiceleague/category/software-security-touchpoints/

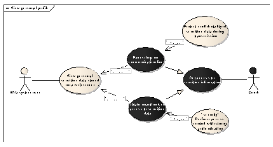
The many dialects



"there is no single best or correct way of performing threat modeling, it is a question of trade-offs and what we want to achieve by doing it"

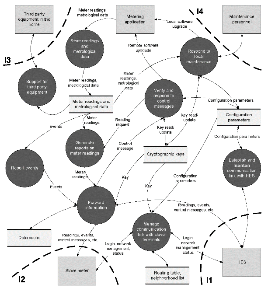
Adam Shostack,
"Experiences Threat Modeling at Microsoft," in Modeling Security Workshop, in Association with MODELS'08, 2008.

Some ways to think like an attacker



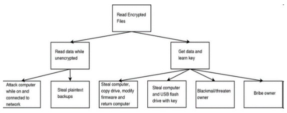
Misuse case diagrams

- Requirement centric



Data flow diagrams

- System centric



Attack trees

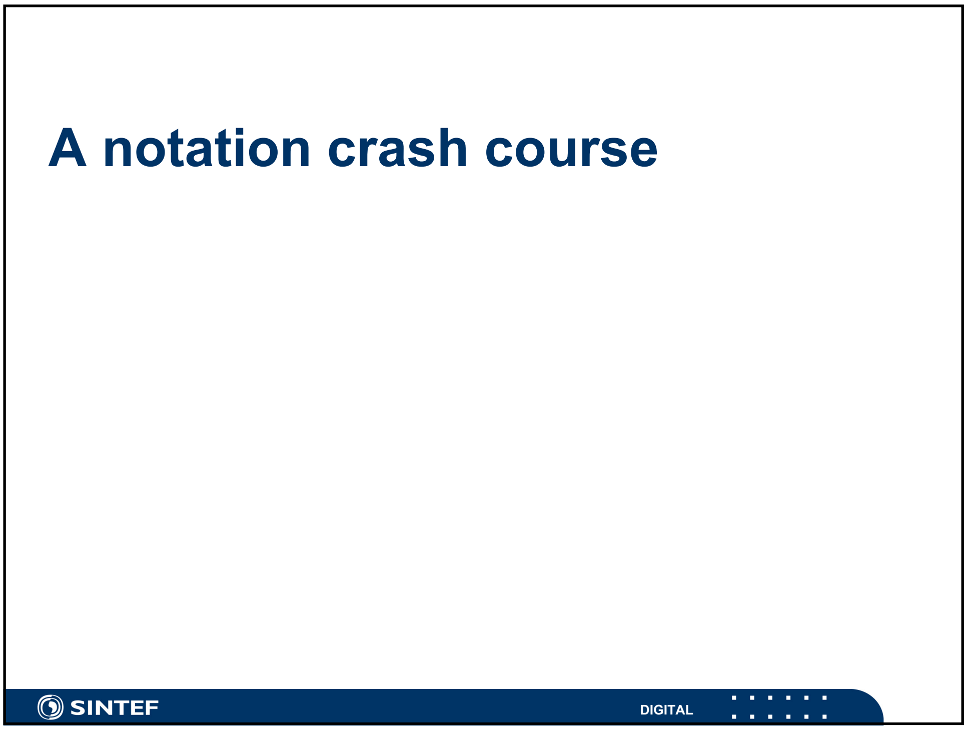
- Attacker centric

STRIDE

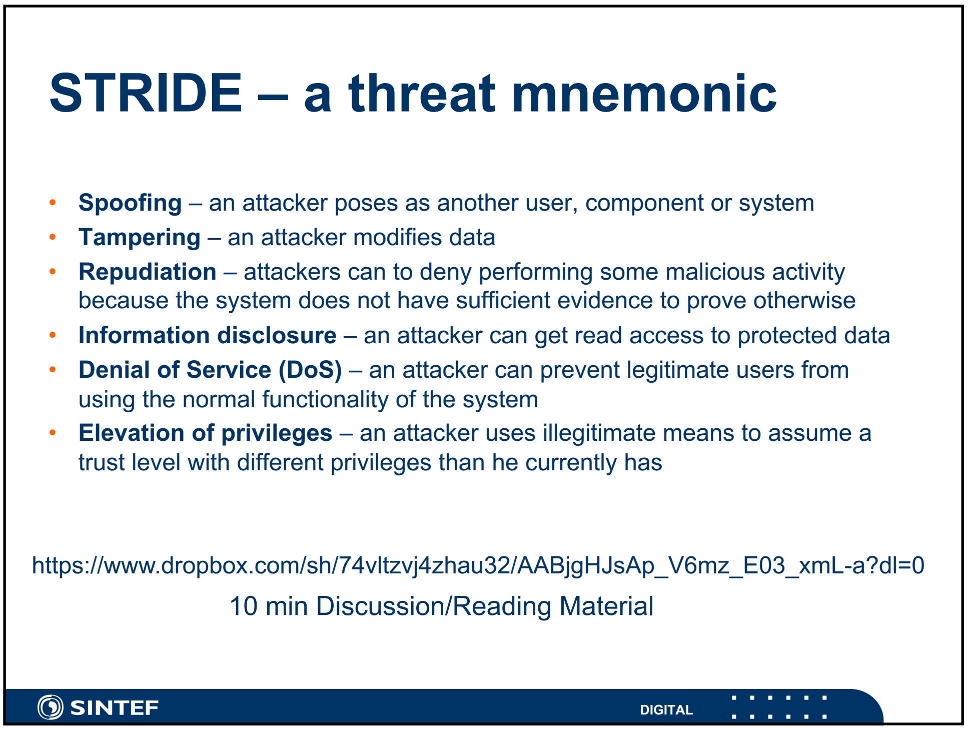
CIA

Top 10

OWASP



A notation crash course



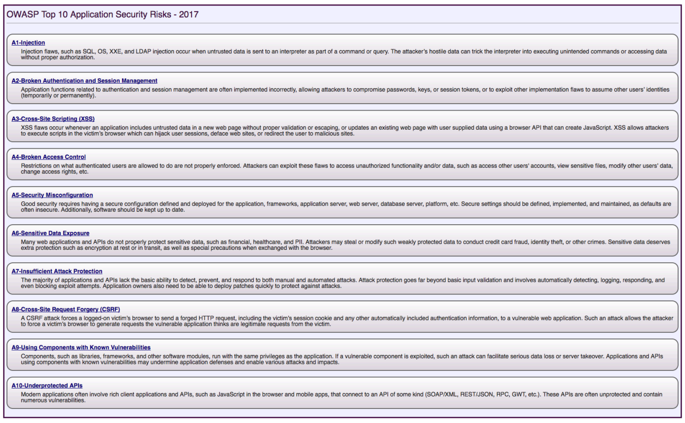
STRIDE – a threat mnemonic

- **Spoofing** – an attacker poses as another user, component or system
- **Tampering** – an attacker modifies data
- **Repudiation** – attackers can to deny performing some malicious activity because the system does not have sufficient evidence to prove otherwise
- **Information disclosure** – an attacker can get read access to protected data
- **Denial of Service (DoS)** – an attacker can prevent legitimate users from using the normal functionality of the system
- **Elevation of privileges** – an attacker uses illegitimate means to assume a trust level with different privileges than he currently has

https://www.dropbox.com/sh/74vltzvj4zhau32/AABjgHJsAp_V6mz_E03_xmL-a?dl=0

10 min Discussion/Reading Material

Top 10 Owasp 2017



OWASP Top 10 Application Security Risks - 2017

A1-Injection
Injection flaws, such as SQL, OS, XXE, and LDAP injection occur when untrusted data is sent to an interpreter as part of a command or query. The attacker's hostile data can trick the interpreter into executing unintended commands or accessing data without proper authorization.

A2-Broken Authentication and Session Management
Application functions related to authentication and session management are often implemented incorrectly, allowing attackers to compromise passwords, keys, or session tokens, or to exploit other implementation flaws to assume other users' identities (temporarily or permanently).

A3-Cross-Site Scripting (XSS)
XSS flaws occur whenever an application includes untrusted data in a new web page without proper validation or escaping, or updates an existing web page with user supplied data using a browser API that can create JavaScript. XSS allows attackers to execute scripts in the victim's browser which can hijack user sessions, deface web sites, or redirect the user to malicious sites.

A4-Broken Access Control
Restrictions on what authenticated users are allowed to do are not properly enforced. Attackers can exploit these flaws to access unauthorized functionality and/or data, such as access other users' accounts, view sensitive files, modify other users' data, change access rights, etc.

A5-Security Misconfiguration
Good security requires having a secure configuration defined and deployed for the application, frameworks, application server, web server, database server, platform, etc. Secure settings should be defined, implemented, and maintained, as defaults are often insecure. Additionally, software should be kept up to date.

A6-Sensitive Data Exposure
Many web applications and APIs do not properly protect sensitive data, such as financial, healthcare, and PII. Attackers may steal or modify such weakly protected data to conduct credit card fraud, identity theft, or other crimes. Sensitive data deserves extra protection such as encryption at rest or in transit, as well as special precautions when exchanged with the browser.

A7-Insufficient Attack Protection
The majority of applications and APIs lack the basic ability to detect, prevent, and respond to both manual and automated attacks. Attack protection goes far beyond basic input validation and involves automatically detecting, logging, responding, and even blocking exploit attempts. Application owners also need to be able to deploy patches quickly to protect against attacks.

A8-Cross-Site Request Forgery (CSRF)
A CSRF attack forces a logged-on victim's browser to send a forged HTTP request, including the victim's session cookie and any other automatically included authentication information, to a vulnerable web application. Such an attack allows the attacker to force a victim's browser to generate requests the vulnerable application thinks are legitimate requests from the victim.

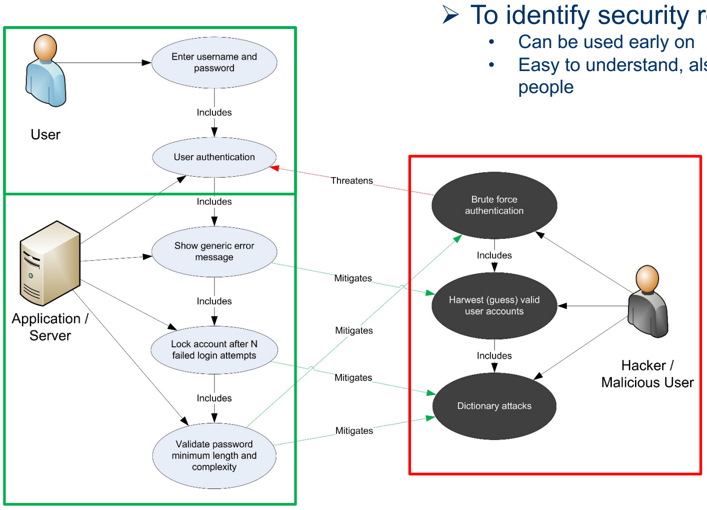
A9-Using Components with Known Vulnerabilities
Components, such as libraries, frameworks, and other software modules, run with the same privileges as the application. If a vulnerable component is exploited, such an attack can facilitate serious data loss or server takeover. Applications and APIs using components with known vulnerabilities may undermine application defenses and enable various attacks and impacts.

A10-Underprotected APIs
Modern applications often involve rich client applications and APIs, such as JavaScript in the browser and mobile apps, that connect to an API of some kind (SOAP/XML, REST/JSON, RPC, GWT, etc.). These APIs are often unprotected and contain numerous vulnerabilities.

10 min Discussion/Reading Material

Misuse Case Diagrams

- To identify security requirements
  - Can be used early on
  - Easy to understand, also for non-technical people


Enter username and password
User
Includes
User authentication
Threatens
Includes
Application / Server
Show generic error message
Includes
Lock account after N failed login attempts
Includes
Validate password minimum length and complexity
Mitigates
Mitigates
Mitigates
Mitigates
Brute force authentication
Includes
Harwest (guess) valid user accounts
Includes
Dictionary attacks
Hacker / Malicious User

Source: https://www.owasp.org/index.php/Application_Threat_Modeling

But we do not create use case diagrams…

Abuser stories

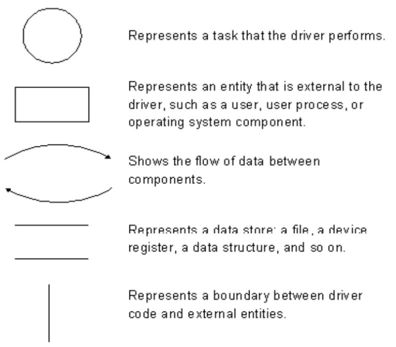
As an Authenticated Customer, I paste HTML that includes JavaScript into every field possible to see what happens.

As a Malicious Hacker, I want to gain access to this web application's Cloud Hosting account so that I can lock out the legitimate owners and delete the servers and their backups, to destroy their entire business.

Examples taken from:
https://rietta.com/blog/2015/10/11/what-is-an-abuser-story-software/

Data Flow Diagrams

- To understand the system's attack surface
  - Get an overview
  - Trust boundaries
  - How data flows in the system



Represents a task that the driver performs.

Represents an entity that is external to the driver, such as a user, user process, or operating system component.

Shows the flow of data between components.

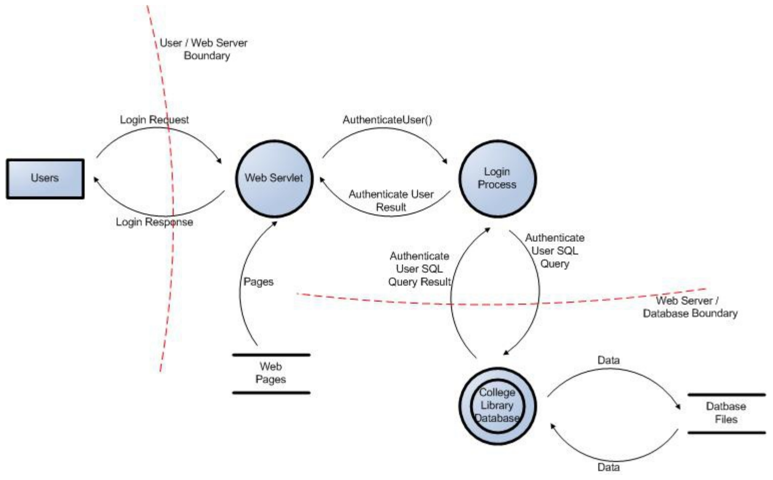
Represents a data store: a file, a device register, a data structure, and so on.

Represents a boundary between driver code and external entities.

10 min Discussion/Reading Material

Case and data flow diagram taken from https://www.owasp.org/index.php/Application_Threat_Modeling

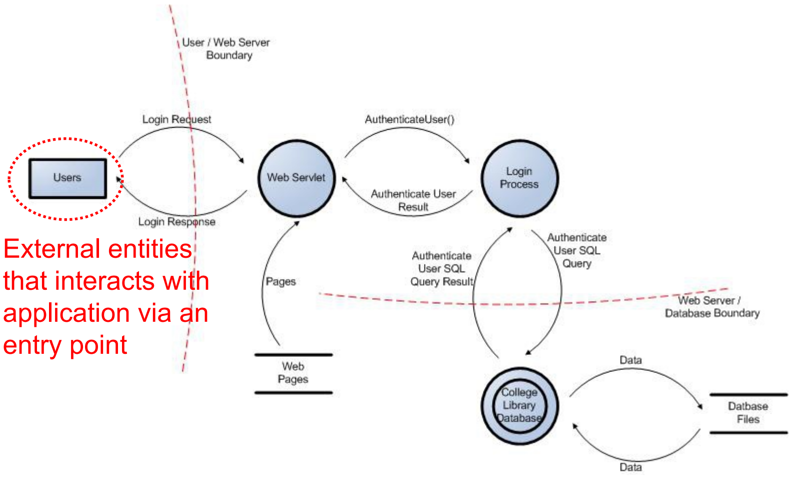
Data Flow Diagram example



SINTEF DIGITAL

Case and data flow diagram taken from https://www.owasp.org/index.php/Application_Threat_Modeling

Data Flow Diagram example

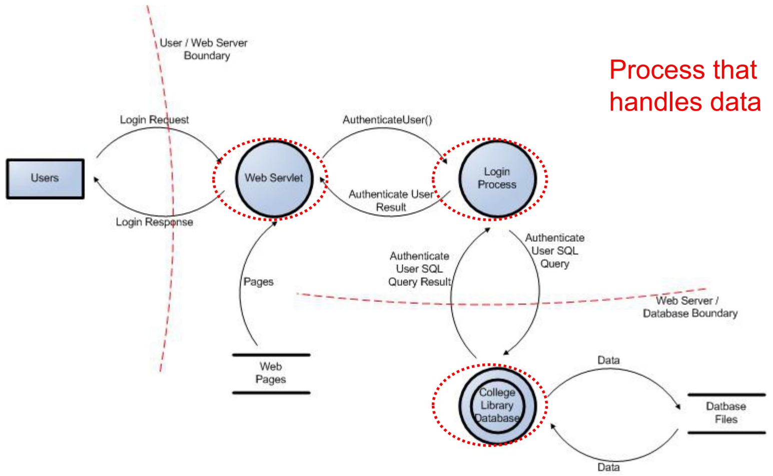


External entities that interacts with application via an entry point

SINTEF DIGITAL

Case and data flow diagram taken from
https://www.owasp.org/index.php/Application_Threat_Modeling

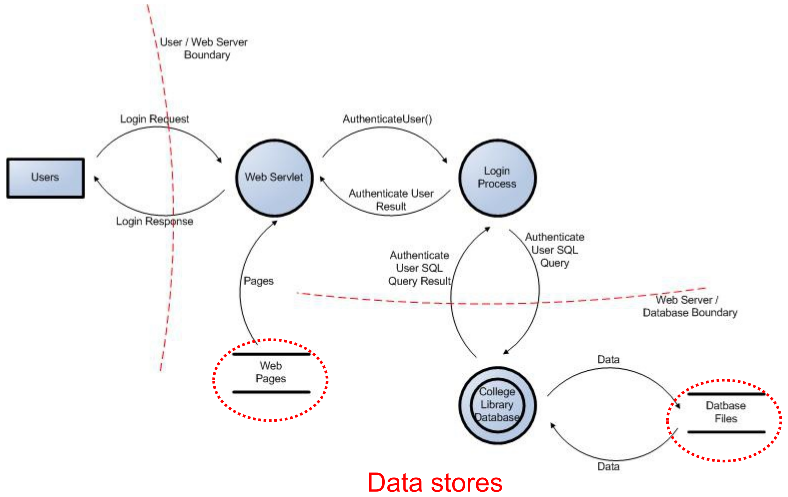
Data Flow Diagram example



Process that handles data

Case and data flow diagram taken from
https://www.owasp.org/index.php/Application_Threat_Modeling

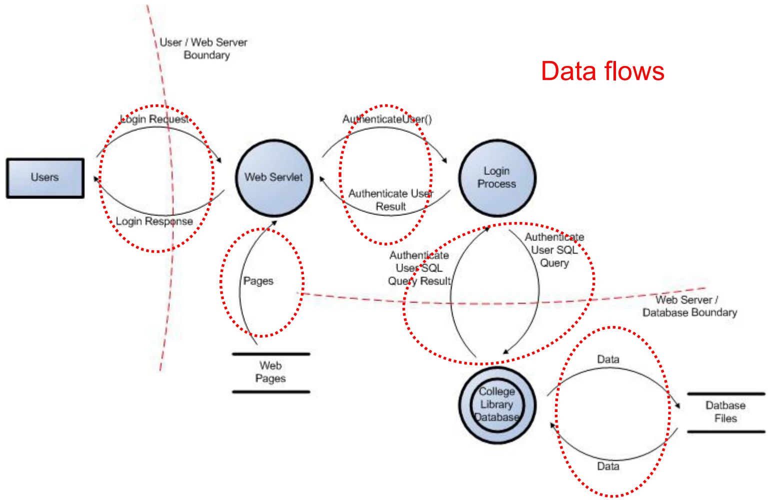
Data Flow Diagram example



Data stores

Case and data flow diagram taken from
https://www.owasp.org/index.php/Application_Threat_Modeling

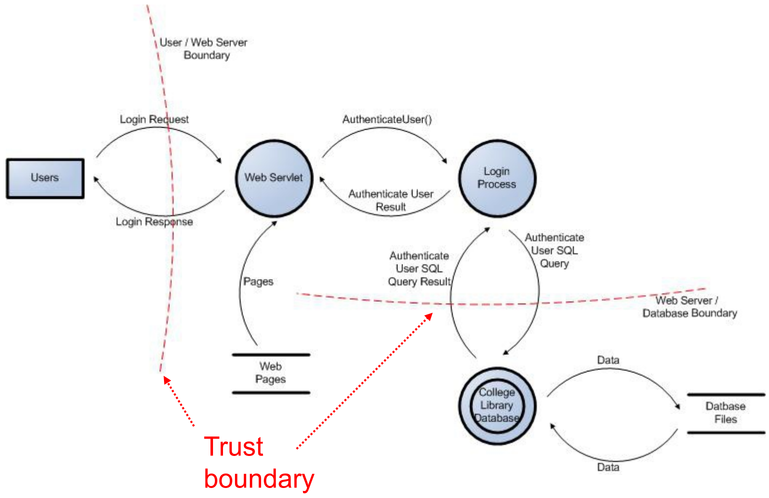
Data Flow Diagram example



Data flows

Case and data flow diagram taken from
https://www.owasp.org/index.php/Application_Threat_Modeling

Data Flow Diagram example

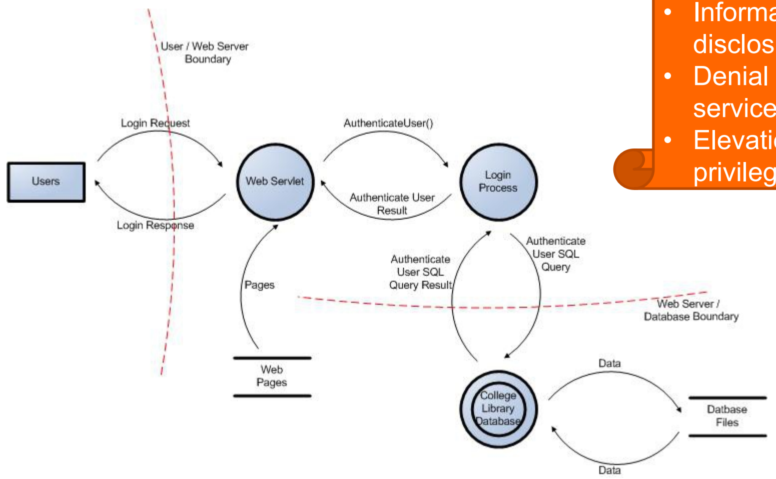


Trust
boundary

Case and data flow diagram taken from https://www.owasp.org/index.php/Application_Threat_Modeling

Data Flow Diagram examp

- Spoofing
- Tampering
- Repudiation
- Information disclosure
- Denial of service
- Elevation of privilege



You can even make it into a game!

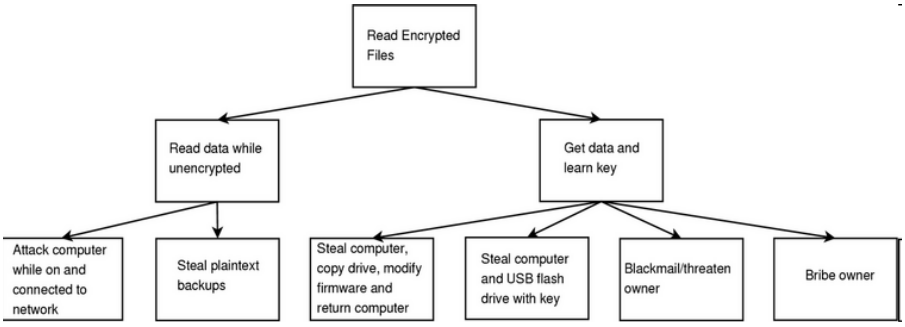
5 Tampering
An attacker can replay data without detection because your code doesn't provide timestamps or sequence numbers
Q Spoofing
An attacker could go after the way credentials are updated or recovered (account recovery doesn't require disclosing the old password)
3 Spoofing
An attacker could try one credential after another and there's nothing to slow them down (online or offline)
J Spoofing
An attacker could steal credentials stored on the client and reuse them
SESSION MANAGEMENT
J
Jeff can resend an identical repeat interaction (e.g. HTTP request, signal, button press) and it is accepted, not rejected
OWASP SCP
-
OWASP ASVS
15.2
OWASP AppSensor
IE5
CAPEC
60
SAFECODE
12, 14

Microsoft EoP

OWASP Cornucopia

Attack trees

- To explore attacker goals and strategies
  - Go deeper on specific threats
  - Basis for planning security tests

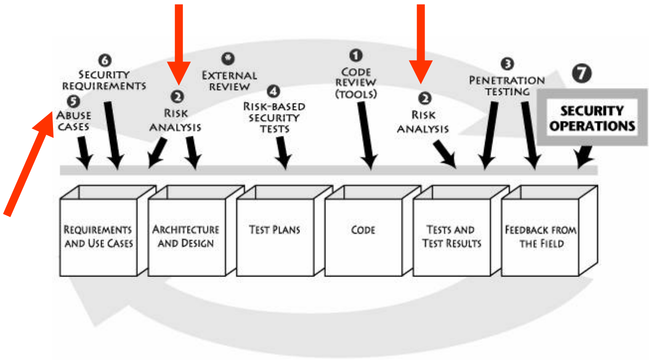

Read Encrypted Files
Read data while unencrypted
Get data and learn key
Attack computer while on and connected to network
Steal plaintext backups
Steal computer, copy drive, modify firmware and return computer
Steal computer and USB flash drive with key
Blackmail/threaten owner
Bribe owner

5 min Discussion/Reading Material

When?

Where would you put
- Misuse cases?
- DFDs?
- Attack trees?

Software Security Touchpoints (McGraw)

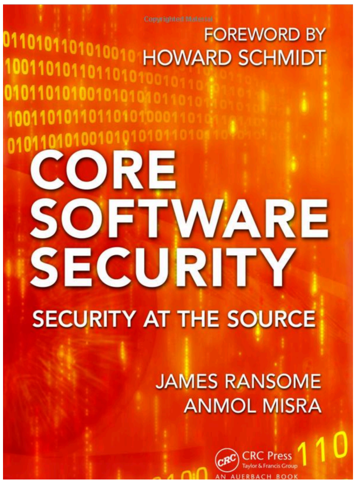


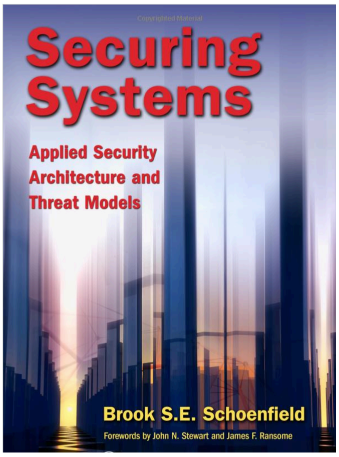
Source: http://www.cigital.com/justiceleague/category/software-security-touchpoints/

Sum up

- Threat modeling is a way of considering possible attacks to your system, users, organisation and environment
- Potential benefits:
  - Address problems early
  - Better security assessments and security testing
  - Limited resources for security – allows you to focus your effort on the key issues
  - Increase awareness about security – make people excited about security
- No single correct way to do it
  - misuse case diagrams, DFDs + STRIDE, attack trees – and many more…..
- Agile is no excuse
- Use the threat models to make good decisions on what security activities and mechanisms are needed for the system

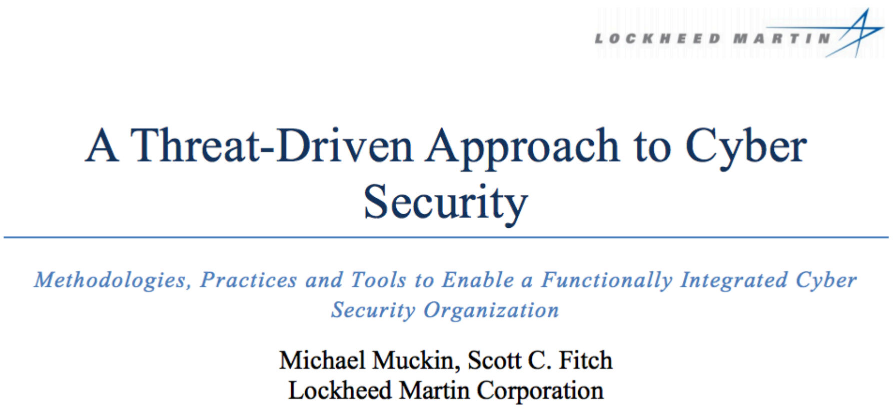
Resources




Securing Systems
Applied Security Architecture and Threat Models
Brook S.E. Schoenfield
Forewords by John N. Stewart and James F. Ransome

Software Security Library Boxed Set

by
Gary McGraw,
John Viega,
Greg Hoglund



LOCKHEED MARTIN

A Threat-Driven Approach to Cyber Security

Methodologies, Practices and Tools to Enable a Functionally Integrated Cyber Security Organization

Michael Muckin, Scott C. Fitch
Lockheed Martin Corporation

http://lockheedmartin.com/content/dam/lockheed/data/isgs/documents/Threat-Driven%20Approach%20whitepaper.pdf

Series of Talks from NCSU – Laurie Williams

- https://www.youtube.com/watch?v=9CnpHT5Nn8c

SQLi Hall of Shame

http://codecurmudgeon.com/wp/sql-injection-hall-of-shame/

Online Resources

- Video:
  - https://www.youtube.com/watch?v=zDPwJXaXsKU
- Microsoft SDL
  - http://www.microsoft.com/sdl
  - http://blogs.msdn.com/sdl
  - https://www.microsoft.com/en-us/sdl/adopt/eop.aspx
  - http://social.technet.microsoft.com/wiki/contents/articles/285.elevation-of-privilege-the-game.aspx?PageIndex=2
- More on SEI DevOps Blog
  - https://insights.sei.cmu.edu/devops

Online Resources

- OWASP Application Threat Modeling guide:
  - https://www.owasp.org/index.php/Application_Threat_Modeling
- CAPEC: Common Attack Pattern Enumeration and Classification:
  - https://capec.mitre.org/index.html
- OWASP Top 10:
  - https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project
- The Microsoft EoP game:
  - https://www.microsoft.com/en-us/sdl/adopt/eop.aspx
- The OWASP Cornucopia game:
  - https://www.owasp.org/index.php/OWASP_Cornucopia
- https://msdn.microsoft.com/en-us/library/ff648641.aspx
- https://books.nowsecure.com/secure-mobile-development/en/sensitive-data/implement-secure-data-storage.html

Questions?

Daniela Cruzes
danielac@sintef.no

SoS-Agile

The SoS-Agile project is funded by the Research Council of Norway IKTPLUSS program.