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](http://dblp.uni-trier.de/pers/hd/c/Cruzes:Daniela)
daniela.s.cruzes@sintef.no
Martin G. Jaatun

- **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](https://scholar.google.no/citations?user=R8Fw41kAAAAJ&hl=no&oi=ao)

---

**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

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.
The principles of agile methods

<table>
<thead>
<tr>
<th>Principle</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer involvement</td>
<td>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.</td>
</tr>
<tr>
<td>Incremental delivery</td>
<td>The software is developed in increments with the customer specifying the requirements to be included in each increment.</td>
</tr>
<tr>
<td>People not process</td>
<td>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.</td>
</tr>
<tr>
<td>Embrace change</td>
<td>Expect the system requirements to change and so design the system to accommodate these changes.</td>
</tr>
<tr>
<td>Maintain simplicity</td>
<td>Focus on simplicity in both the software being developed and in the development process. Wherever possible, actively work to eliminate complexity from the system.</td>
</tr>
</tbody>
</table>

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

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/
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.


---

Scrum – overview

![Scrum Process Diagram](http://www.agilebuddha.com/)
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

- Individuals and interactions over processes and tools
- Working systems over comprehensive documentation
- Customer 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?

...and I should care, why?
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.
The cost of fixing a bug in production is 100x bigger than during coding/design.

Most vulnerabilities are coding and design issues, but they are mostly found during Pen Tests.

Source: Applied Software Measurement.

Top Obstacles to Implementing DevOps

- Organizational complexity – people, departments, interdependencies: 35%
- Roles and responsibilities across dev & ops not aligned: 28%
- Security or compliance concerns: 25%
- Lack of understanding of phases of dev lifecycle & who is responsible: 24%
- No budget/no clarity over whose budget is responsible for what: 24%
- Lack of understanding of the business impact by management team: 23%
- The difficulty of merging the working processes of different teams: 21%
- Lack of the right skills within development and operations: 21%

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.

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

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.
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.


Threats, Assets and Controls Relationship Model

Security Objectives

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

Overview

Agile/DevOps Development
- 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

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
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).
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

<table>
<thead>
<tr>
<th>Committed Backlog Items</th>
<th>Not Started</th>
<th>In Progress</th>
<th>Completed</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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.
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

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
5 Categories of Challenges

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

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
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

- **Network Threats**
  - Spoofed packets, etc.

- **Host Threats**
  - Buffer overflows, illicit paths, etc.

- **Application Threats**
  - SQL injection, XSS, input tampering, etc.

Threats Against the Application

<table>
<thead>
<tr>
<th>Threat</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQL injection</td>
<td>Including a DROP TABLE command in text typed into an input field</td>
</tr>
<tr>
<td>Cross-site scripting</td>
<td>Using malicious client-side script to steal cookies</td>
</tr>
<tr>
<td>Hidden-field tampering</td>
<td>Maliciously changing the value of a hidden field</td>
</tr>
<tr>
<td>Eavesdropping</td>
<td>Using a packet sniffer to steal passwords and cookies from unencrypted connections</td>
</tr>
<tr>
<td>Session hijacking</td>
<td>Using a stolen session ID cookie to access someone else's session state</td>
</tr>
<tr>
<td>Identity spoofing</td>
<td>Using a stolen forms authentication cookie to pose as another user</td>
</tr>
<tr>
<td>Information disclosure</td>
<td>Allowing a client to see a stack trace when an unhandled exception occurs</td>
</tr>
</tbody>
</table>
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

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.
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

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


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?

- 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)

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.

---

**Protection Poker**

- 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
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

- 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
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

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!
Agile SDLs

SDLC Cornerstones

Risk
- People
  - Roles and Responsibilities
- Process
  - Activities
  - Deliverables
  - Control Gates
- Knowledge
  - Standards & Guidelines
  - Compliance
  - Transfer Methods
- Tools & Components
  - Development and Support
  - Assessment Tools
  - Management Tools

Training
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
- CLASP
- TSP-Secure

SDL-Agile process - Microsoft
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

- 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)

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

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.
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)

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"


Some ways to think like an attacker

Misuse case diagrams
- Requirement centric

Data flow diagrams
- System centric

Attack trees
- Attacker centric

Top 10 OWASP

STRIKE
CIA
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 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/74vtzvj4zhau32/AABjgHJsAp_V6mz_E03_xml-a?dl=0

10 min Discussion/Reading Material
Top 10 Owasp 2017

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

Source: https://www.owasp.org/index.php/Application_Threat_Modeling
But we do not create use case diagrams…

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.


Data Flow Diagrams

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

10 min Discussion/Reading Material
Data Flow Diagram example

External entities that interact with the application via an entry point.
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

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

Data Flow Diagram example

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

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

You can even make it into a game!

Microsoft EoP

OWASP Cornucopia
**Attack trees**

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

![Attack trees diagram](image)

5 min Discussion/Reading Material

---

**When?**

Software Security Touchpoints (McGraw)

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

![Software Security Touchpoints diagram](image)

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
Software Security Library Boxed Set
by
Gary McGraw, John Viega, Greg Hoglund

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

Series of Talks from NCSU – Laurie Williams

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

Online Resources

- Video:
  - https://www.youtube.com/watch?v=zDPwJXaXsKU

- Microsoft SDL
  - http://www.microsoft.com/sdl
  - http://blogs.msdn.com/sdl

- 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:
- The Microsoft EoP game:
- The OWASP Cornucopia game:
  - https://www.owasp.org/index.php/OWASP_Cornucopia

Questions?

Daniela Cruzes
danielac@sintef.no

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