Software Security Engineering
Lecture 9

Nancy R. Mead, SEI
nrm@sei.cmu.edu

Topics

1. Attack surface
2. Measurement
3. Inspecting for security

Attack Surface Video and Discussion

Security Measurement Research

Outline

1. Definitions and Questions
2. Example Drivers and Considerations
3. Revised Considerations for Security Requirements
4. Conclusions and Future Work
5. Questions

Definitions and Questions

Definitions

Software assurance (SwA) is the level of confidence that software is free from vulnerabilities, either intentionally designed into the software or accidentally inserted at any time during its life cycle, and that the software functions in the intended manner. **(CNSS)**

Software security assurance is justified confidence that software-reliant systems are adequately planned, acquired, built, and fielded with sufficient security to meet operational needs, even in the presence of attacks, failures, accidents, and unexpected events. **(SSMA Project)**

Questions to be Answered by SSMA

1. How do we establish, specify, and measure justified confidence that interactively complex, software-reliant systems are sufficiently secure to meet operational needs?

2. At each phase of the development or acquisition lifecycle, how do we measure that the required/desired level of security has been achieved?

Example Drivers and Considerations

Driver 4: Security Process

Driver 4: Security Process

Driver Question

Does the process being used to develop and deploy the system sufficiently incorporate security?

Considerations:

- Security-related tasks and activities in the program workflow
- Conformance to security process models
- Measurements and controls for security-related tasks and activities
- Process efficiency and effectiveness
- Software security development life cycle
- Security-related training
- Compliance with security policies, laws, and regulations
- Security of all product-related information

Response

- ❑ Yes
- ❑ Likely Yes
- ❑ Equally Likely
- ☒ Likely No
- ❑ No
- ❑ Don't Know

Rationale

+ Program management recognizes that security should be addressed. The program's process documentation states the importance of addressing security when engineering software and systems.
- The program does not have a formal process for developing secure software and systems.
- The program does not have a means of measuring and controlling software and system security.
- Security training for developers and testers has been scheduled but keeps getting postponed due to scheduling conflicts.
- Program management and staff lack sufficient awareness of applicable security-related laws and regulations.

Draft Considerations

Driver 10: Security Requirements

Driver Question

Do requirements sufficiently address security?

Considerations:

1. Process for developing and coordinating security requirements
2. Customer, user, and stakeholder security requirements and needs
3. Tradeoffs between security, performance, and other quality attributes
4. Operational security requirements
5. Information security requirements
6. Maturity of technology used and implications for security requirements
7. Relevant policies, standards, guidelines, and regulations
8. Results of risk analysis of security requirements
9. Analysis of security threats as they affect security requirements (using methods such as misuse/abuse cases, threat models, and attack patterns)

Revised Considerations for Security Requirements

SQUARE Process

Revised Considerations

Considerations:

1. Process for developing and coordinating security requirements *(existence of a process such as SQUARE)*
2. Agree on definitions and identify assets and security goals *(SQUARE steps 1 and 2)*
3. Relevant policies, standards, guidelines, and regulations *(SQUARE step 3)*
4. Results of risk analysis of security requirements *(SQUARE step 4)*
5. Maturity of technology used and implications for security requirements *(SQUARE step 4)*
6. Analysis of security threats as they affect security requirements (using methods such as misuse/abuse cases, threat models, and attack patterns) *(SQUARE step 3 or 4)*
7. Customer, user, and stakeholder security requirements and needs, developed in conjunction with requirements engineers *(SQUARE steps 5 and 6)*
8. Information security requirements *(SQUARE step 6)*
9. Security requirements categorization and prioritization *(SQUARE steps 7 and 8)*
10. Operational security requirements *(should appear in SQUARE step 6)*

Application of Original Considerations *(1)*

Driver 10: Security Requirements

Driver Question

Do requirements sufficiently address security?

Considerations:

1. Process for developing and coordinating security requirements
2. Customer, user, and stakeholder security requirements and needs
3. Tradeoffs between security, performance, and other quality attributes
4. Operational security requirements
5. Information security requirements
6. Maturity of technology used and implications for security requirements
7. Relevant policies, standards, guidelines, and regulations
8. Results of risk analysis of security requirements
9. Analysis of security threats as they affect security requirements (using methods such as misuse/abuse cases, threat models, and attack patterns)

Application of Original Considerations *(2)*

Driver 10: Security Requirements *(cont.)*

Rationale:

+ The project has a process for developing and coordinating security requirements that involves all stakeholder groups.
+ The project develops information security requirements, considering risk and threat analysis.
+ The project considers the technology used and relevant artifacts such as policies and standards.
− The process does not incorporate tradeoffs with other quality attributes, but it does prioritize security requirements.
− The process does not consider operational security requirements except peripherally.

Response: Likely Yes

Application of Revised Considerations to an Actual Project *(1)*

Driver 10: Security Requirements

Driver Question

Do requirements sufficiently address security?

Considerations:

1. Process for developing and coordinating security requirements *(existence of a process such as SQUARE)*
2. Agree on definitions and identify assets and security goals *(SQUARE steps 1 and 2)*
3. Relevant policies, standards, guidelines, and regulations *(SQUARE step 3)*
4. Results of risk analysis of security requirements *(SQUARE step 4)*
5. Maturity of technology used and implications for security requirements *(SQUARE step 4)*
6. Analysis of security threats as they affect security requirements (using methods such as misuse/abuse cases, threat models, and attack patterns) *(SQUARE step 3 or 4)*
7. Customer, user, and stakeholder security requirements and needs, developed in conjunction with requirements engineers *(SQUARE steps 5 and 6)*
8. Information security requirements *(SQUARE step 6)*
9. Security requirements categorization and prioritization *(SQUARE steps 7 and 8)*
10. Operational security requirements *(should appear in SQUARE step 6)*

Application of Revised Considerations to an Actual Project *(2)*

Driver 10: Security Requirements *(cont.)*

Rationale:

+ The project has a process for developing and coordinating security requirements that involves all stakeholder groups and requirements engineers.
+ The project considers definitions, assets, and security goals.
+ The project develops information security requirements, considering risk and threat analysis.
+ The project considers the technology used and relevant artifacts such as policies and standards.
− The process does not consider operational security requirements except peripherally.

Response: Yes or Likely Yes, depending on whether operational security requirements are considered

Application of Revised Considerations to an Actual Project without Security Goals *(1)*

Driver 10: Security Requirements

Driver Question

Do requirements sufficiently address security?

Considerations:

1. Process for developing and coordinating security requirements *(existence of a process such as SQUARE)*
2. Agree on definitions and identify assets and security goals *(SQUARE steps 1 and 2)*
3. Relevant policies, standards, guidelines, and regulations *(SQUARE step 3)*
4. Results of risk analysis of security requirements *(SQUARE step 4)*
5. Maturity of technology used and implications for security requirements *(SQUARE step 4)*
6. Analysis of security threats as they affect security requirements (using methods such as misuse/abuse cases, threat models, and attack patterns) *(SQUARE step 3 or 4)*
7. Customer, user, and stakeholder security requirements and needs, developed in conjunction with requirements engineers *(SQUARE steps 5 and 6)*
8. Information security requirements *(SQUARE step 6)*
9. Security requirements categorization and prioritization *(SQUARE steps 7 and 8)*
10. Operational security requirements *(should appear in SQUARE step 6)*

Application of Revised Considerations to an Actual Project without Security Goals *(2)*

Driver 10: Security Requirements *(cont.)*

Rationale:

- − The project does not have a process for developing and coordinating security requirements.
- − The project did not consider definitions, assets, and security goals.
- − The project did not do risk analysis for security requirements.
- − The project develops information security requirements but does not do risk or threat analysis.
- − The project did not consider the technology used or relevant artifacts such as policies and standards.
- \+ The project considers operational security requirements.
- − Security requirements are not categorized or prioritized.

Response: No or Likely No, depending on credit given for attempting to include security.

Conclusions and Future Work

Conclusions

1. Merging the SSMA results with SQUARE resulted in a better process assessment.

2. The results were consistent with our own view of the two actual projects.

Needed Improvements and Future Work

1. Try this with other security requirements engineering processes.
2. Revise the measurement approach to make it more objective.
3. Benchmark other work in the field.
4. Look at product and cost–benefit, not just process.

Inspecting for security

Inspecting Requirements

Goal is to find defects in the requirements

- Ambiguities
- Inconsistencies
- Incorrect assumptions

Varying levels of formality

- Fagan Inspection
- Peer reviews

Inspecting Requirements

Peer review log format

SNo	DATE	ORIGIN	DEFECT TYPE	DESCRIPTION	SEVERITY	OWNER	REVIEWER	STATUS

Assigns each team member inspection responsibility

Ranks problems according to severity

Inspecting Requirements

- Requirements engineering team responsibilities
  - Facilitate the inspection process
    - —Provide orientation to the structured inspection
    - —Informal inspection guides

Inspecting Requirements

Stakeholder responsibilities

- Come to a consensus on the validity of each security requirement
- Verify that each requirement is verifiable and feasible to implement
- Remove requirements, if needed, from the working set (Last chance!)

Inspecting Requirements

Joint responsibilities

- Verify that each requirement is directly applicable to one or more of the security goals of the project or supports a higher level requirement

Inspecting Requirements

Exit criteria

- All requirements have been verified both by the requirements engineering team and the stakeholders

Validating Threat Models

- Validate the whole threat model
  - Does diagram match final code?
  - Are threats enumerated?
  - Minimum: STRIDE per element that touches a trust boundary
  - Has Test / QA reviewed the model?
    - Tester approach often finds issues with threat model or details
  - Is each threat mitigated?
  - Are mitigations done right?
- Did you check these before Final Security Review?
  - Shipping will be more predictable

Validate Quality of Threats and Mitigations

- Threats: Do they:
  - Describe the attack
  - Describe the context
  - Describe the impact
- Mitigations
  - Associate with a threat
  - Describe the mitigations
  - File a bug
    - ✖ Fuzzing is a test tactic, not a mitigation

Validate Information Captured

- Dependencies
  - What other code are you using?
  - What security functions are in that other code?
  - Are you sure?
- Assumptions
  - Things you note as you build the threat model
    - ✗ "HTTP.sys will protect us against SQL Injection"
    - ✗ "LPC will protect us from malformed messages"
    - ✓ GenRandom will give us crypto-strong randomness

Exercise (10-15 Minutes)

Task: Review and document review comments

Exit Criteria:

- All requirements should be carefully reviewed and the review comments should be documented

If there is any drastic change in the requirements, the categorization and prioritization might change.

Both teams should read and follow the instructions given in your respective documents for more detailed information.

Resources

Attack Surface Tool Download:http://blogs.msdn.com/b/sdl/archive/2012/08/02/attack-surface-analyzer-1-0-released.aspx

All Microsoft Tool Downloads: http://www.microsoft.com/security/sdl/adopt/tools.aspx

Microsoft Webcasts (including Attack Surface) http://msdn.microsoft.com/en-us/security/aa570424.aspx

Questions?

Reading Assignment

Software Security Engineering Chapter 6

Paper: Measuring The Software Security Requirements Engineering Process

Paper: Attack Surface Measurement

http://www.cs.cmu.edu/~pratyus/tse10.pdf from website

http://www.cs.cmu.edu/~pratyus/as.html

Homework Assignment # 4

Download the Microsoft Attack Surface tool and follow the download instructions to model the attack surface on your computer. If you are unable to do the download and execution, do as much as possible by hand. Either way, include the results. (40%)

- What did you learn as a result of the attack surface calculation? (20%)
- Will you make changes to your configuration OR to a development project on the basis of the attack surface video and your analysis? What are they? (40%)

Assignment is due on Tuesday August 7 prior to class.

Looking Ahead: Lecture #10

Secure Coding with David Svoboda

NO WARRANTY

THIS MATERIAL OF CARNEGIE MELLON UNIVERSITY AND ITS SOFTWARE ENGINEERING INSTITUTE IS FURNISHED ON AN “AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

Use of any trademarks in this presentation is not intended in any way to infringe on the rights of the trademark holder.

This work was created in the performance of Federal Government Contract Number FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center. The Government of the United States has a royalty-free government-purpose license to use, duplicate, or disclose the work, in whole or in part and in any manner, and to have or permit others to do so, for government purposes pursuant to the copyright license under the clause at 252.227-7013.