DESIGNING SOFTWARE SECURITY WITH UML EXTENSIONS*

POST-CONFERENCE WORKSHOP

Susan J. Lincke PhD CISA
University of Wisconsin – Parkside, Kenosha WI
(262) 595-2129
lincke@uwp.edu

ABSTRACT

Security is becoming required knowledge for software engineers. Building security into the product means that security must be considered during the requirements and throughout the software development process. This workshop considers how UML can be enhanced for security, including through misuse cases, business process diagrams, class diagrams, mis-sequence diagrams, and misuse deployment diagrams. The OCTAVE process to analyze risk is also introduced.

OVERVIEW

The OCTAVE process is an excellent process for defining security risks, as part of the Requirements process. A brief overview of OCTAVE includes the following steps [12]:

1. Identify critical assets: Using (for example) a Business Process Diagram, it is possible to determine what information will be used and is critical for protection.
2. Define security goals: Security goals include Confidentiality, Integrity, and Availability. For the information assets defined in step 1, which security goals are important to each?
3. Identify threats: STRIDE is an excellent technique to determine important threats. S=Spoof Identity, T=Tamper with Data, R=Repudiation, I=Information Disclosure, D=Denial of Service, E=Elevation of Privilege (e.g., including: hide system files, access control, segregation of duties). Threats are further evaluated using a Misuse Case Diagram, and Misuse Cases Descriptions (defined in next paragraph).
4. Analyze risks: Prioritize risks according to the metric: Priority = Impact * Likelihood.

---

5. Define security requirements: In this stage, security use cases are added to the misuse cases.

During the Requirements process, normally a Use Case Diagram is developed, showing the users with their associated major functions (or use cases). For each major function, a Use Case Description is written to describe the function's user interface process. Use Case Diagrams can be modified during OCTAVE stage 3 to show potential attacks, and during stage 5 to show planned defenses. The resulting misuse cases describe attacks on various use cases (e.g., Fig. 1) with misuse components shown in black [7, 9-11]. Misuse Case Diagrams have been extended to not only include misusers (e.g., "attacker") and misuse cases (e.g. "Launch DOS"), but also security use cases (e.g., "Validate Registration") which mitigates the misuse case. Each misuse case is then expanded in a misuse case description, which defines the attack and security mitigation processing [10].

During the Analysis and Design Stages, class diagrams provide a blueprint of how the static system is defined, while sequence diagrams show the programmed interaction between classes to implement a use case. To add security, class diagrams can be modified with security classes or patterns to design security aspects (e.g. [5,8]). UMLpac documents security packages with Risk Factor and Security Descriptors, to show the priority and name the defense mechanism [5]. Mis-sequence diagrams are sequence diagrams that additionally show conditional logic for how attacks occur and are handled [11]. State diagrams can ensure integrity of real-time processing, thus avoiding incorrect actions when receiving the right packet at the wrong time [3].

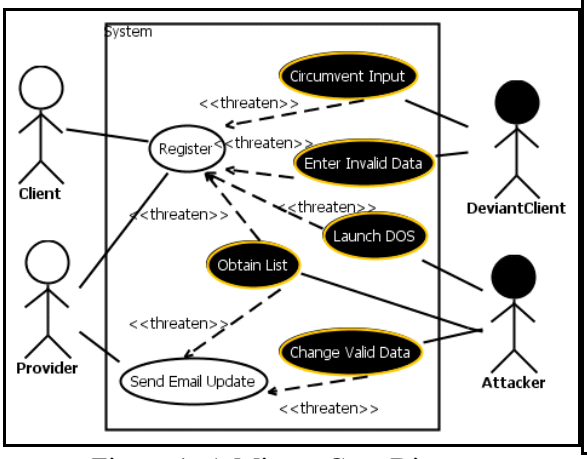

Figure 1: A Misuse Case Diagram

Figure 2: A Misuse Deployment Diagram

Concrete security deployment, including how attacks can be mitigated across systems, is also important. For example, input validation must occur to combat SQL

attacks. But if this validation only occurs on the client side, security is inadequate. Therefore, the location of security code is as important as its existence. A deployment diagram helps to solve this problem, because it shows in a single picture how software is deployed on physical hardware [1].

The Misuse Deployment Diagram (MDD) [4] shows where attacks are handled in a single diagram. A preliminary MDD may be useful in the Requirements Stage, to show where major attacks should be addressed. This first version of the diagram may simply include misusers (or the possible attacks) and where they will be handled (e.g., client/server). During the Analysis and Design stages, the diagram may be further refined, by adding security software components, and additional attack information. See Fig. 2 [4].

Activity diagrams can be used in any development stage, to show the sequence of processing. Business process diagrams are most useful to show organizational flow and processing. These diagrams can be enhanced by including locks to show how security is added to specific business processes [6]. Activity diagrams have also been used during system test for vulnerability testing, using Vulnerability Inspection Diagrams, which flow chart testing [2].

This abstract has discussed a number of proposed techniques to help build security into software. Hopefully you will find some techniques useful in your software development efforts!

ACKNOWLEDGEMENT

These materials, including the development of the Small Business Security Workbook, was funded by the National Science Foundation (NSF) Course, Curriculum and Laboratory Improvement (CCLI) grant 0837574: Information Security: Audit, Case Study, and Service Learning. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF.

REFERENCES

[1] Arlow, J., Neustadt, I., *UML2 and the Unified Process, 2nd Ed.*, Pearson Education, Inc., 481-485, 2005.

[2] Baadshaug, E. T., Erdogan, G., Meland, P. H., "Security modeling and tool support advantages", *2010 International Conf. on Availability, Reliability, and Security*, 537-542, 2010.

[3] Kong, J., Xu, D., "A UML-based Framework for Design and Analysis of Dependable Software", *32nd IEEE International Computer Software and Applications (COMPSAC '08)*, 28-31, 2008.

[4] Lincke, S. J., Knautz, T., Lowery, M., Designing System Security with UML Misuse Deployment Diagrams, *2012 IEEE International Workshop on Information Assurance (IA2012)*, in publication, 2012.

[5] Peterson, M. J., Bowles, J. B., Eastman, C. M., "UMLpac: An approach for Integrating Security into UML Class Design", Proc. *IEEE SoutheastCon*, 267-272, 2006.

[6] Rodriquez, A., Fernandez-Medina, E., Piattini, M., "Security Requirement with a UML 2.0 Profile", *Proc. First International Conf. on Availability, Reliability, and Security*, 1-8, 2006.

[7] SeaMonster Misuse Case Drawing Tool, http://sourceforge.net/projects/seamonster, accessible 2011.

[8] Shiroma, Y., Washizaki, H., Fukazawa, Y., Kubo, A., Yoshioka, N., "Model-Driven Security Patterns Application Based on Dependences among Patterns", *2010 International Conf. on Availability, Reliability, and Security*, 555-559, 2010.

[9] Sindre, G., Opdahl, A. L., "Eliciting Security Requirements by Misuse Cases", *Requirements Engineering*, 10, (1) Jan., Springer-Verlag, 120-131, 2005.

[10] Sindre, G., Opdahl, A. L., "Misuse Cases for Identifying System Dependability Threats", *Journal of Information Privacy & Security*, 4, (2) ABI/INFORM Global, 3-22, 2008.

[11] Whittle, J., Wijesekera, D., Hartong, M., "Executable Misuse Cases for Modeling Security Concerns", *ACM/IEEE 30th International Conf. on Software Engineering*, 121-130, 2008.

[12] Woody, C., Alberts, C., "Considering Operational Security Risk during System Development", *IEEE Security & Privacy*, 5, (1) Jan., 30-43, 2007.