A methodology to build secure systems using patterns

Eduardo B. Fernandez and Maria M Larrondo-Petrie
Dept. of Computer Science and Eng., Florida Atlantic University, Boca Raton, FL 33431

A main idea in the proposed methodology is that security principles should be applied at every stage of the software lifecycle and that each stage can be tested for compliance with security principles [Fer04, Fer06a]. Another basic idea is the use of patterns to guide security at each stage [Sch06]. Patterns are applied in the different architectural levels of the system to realize security mechanisms. Figure 1shows a secure software lifecycle, indicating where security can be applied (white arrows) and where we can audit for compliance with security principles and policies (dark arrows). This project proposes guidelines for incorporating security from the requirements stage through analysis, design, implementation, testing, and deployment. Our approach considers the following development stages:

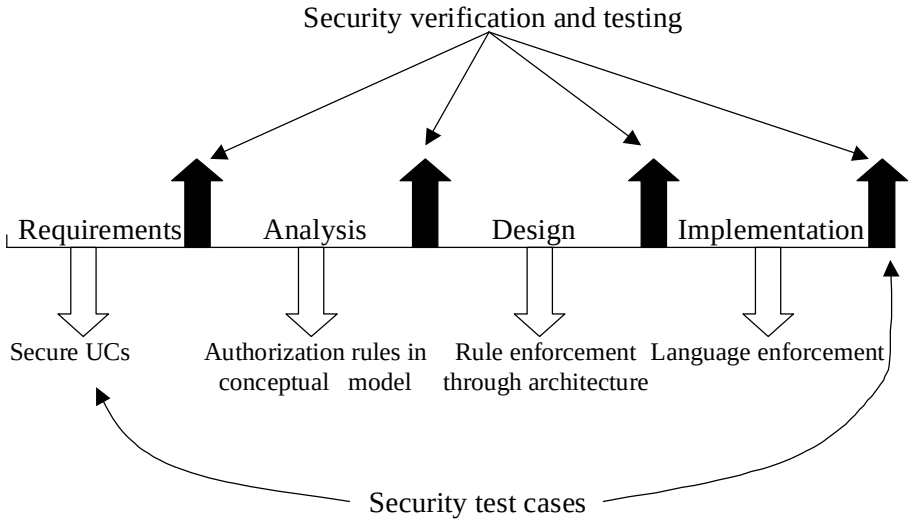

Fig. 1 Secure software lifecycle

Domain analysis stage: A generic conceptual model is defined. Legacy systems are identified and their security implications analyzed. Domain and regulatory constraints are identified. Analysis patterns lead to a domain model. Security policies must be defined up front, in this phase. The suitability of the development team is assessed, possibly leading to added training. Security issues of the developers, themselves, and their environment may also be considered in some cases. This phase may be performed only once for each new domain or team.

Requirements stage: Use cases define the required interactions with the system. Applying the principle that security must start from the highest levels, it makes sense to relate attacks to use cases. We study each action within a use case and see which threats are possible [Fer06b]. We then determine which policies would stop these attacks. From the use cases we can also determine the needed rights for each actor and thus apply a need-to-know policy [Fer97]. Note that the set of all use cases defines all the uses of the system and from all the use cases we can determine all the rights for each actor. The security test cases for the complete system are also defined at this stage.

Analysis stage: Analysis patterns can be used to build the conceptual model in a more reliable and efficient way. We build a conceptual model where repeated applications of a security model pattern [Sch06] realize the rights determined from use cases. In fact, analysis patterns can be built with predefined authorizations according to the roles in their use cases. Then we only need to additionally specify the rights

for those parts not covered by patterns. We can start defining conceptual mechanisms (countermeasures) to prevent attacks.

Design stage: Design mechanisms are selected to stop the attacks identified earlier and realize the required policies [Fer05]. User interfaces should correspond to use cases and may be used to enforce the authorizations defined in the analysis stage. Secure interfaces enforce authorizations when users interact with the system. Components can be secured by using authorization rules for Java or .NET components. Distribution provides another dimension where security restrictions can be applied. Deployment diagrams can define secure configurations to be used by security administrators. A multilayer architecture is needed to enforce the security constraints defined at the application level. In each level we use patterns to represent appropriate security mechanisms. Security constraints must be mapped between levels.

Implementation stage: This stage requires reflecting in the code the security rules defined in the design stage. Because these rules are expressed as classes, associations, and constraints, they can be implemented as classes in object-oriented languages. In this stage we can also select specific security packages or COTS, e.g., a firewall product, a cryptographic package. Some of the patterns identified earlier in the cycle can be replaced by COTS (these can be tested to see if they include a similar pattern).

Current and future work

In addition to the papers mentioned above, we have written several security patterns which appear in [Sch06] and in recent PLoP conferences, e.g. [Del05]. We are preparing more, including patterns for WS-Security, IDS, VPNs, and firewall configurations.

We are incorporating our work in an MDA (Model Driven Architecture) framework [Bro04]. In particular, we are looking at model mappings between architectural levels through the use of patterns.

References

[Bro04] A.W.Brown, “Model driven architecture: Principles and practice”, *Softw. Syst. Model*, vol. 3, 2004, 314-327.

[Del05] N. Delessy, E. B.Fernandez, and T. Sorgente, "Patterns for the eXtensible Access Control Markup Language", *Procs. of the Pattern Languages of Programs Conference (PLoP 2005*), Monticello, Illinois, USA, 7-10 September 2005.

[Fer97] E.B. Fernandez and J.C. Hawkins, “Determining Role Rights from Use Cases”, *Proceedings of the 2nd ACM Workshop on Role-Based Access Control*, November 1997, 121-125, ACM Press, New York, NY, USA. http://www.cse.fau.edu/~ed/RBAC.pdf

[Fer04] E. B. Fernandez, “Two patterns for web services security”, *Proceedings of the International Symposium on Web Services and Applications*, Las Vegas, Nevada, June 2004.

[Fer05] E.B.Fernandez, T. Sorgente, and M.M.Larrondo-Petrie, "A UML-based methodology for secure systems: The design stage", *Procs. of the Third International Workshop on Security in Information Systems (WOSIS-2005)*, Miami, May 24-25, 2005, 207-216.

[Fer06a] E.B. Fernandez, M.M. Larrondo-Petrie, T. Sorgente, and M. VanHilst, "A methodology to develop secure systems using patterns", Chapter 5 in "*Integrating security and software engineering: Advances and future vision"*, H. Mouratidis and P. Giorgini (Eds.), IDEA Press, 2006, 107-126.

[Fer06b] E.B.Fernandez, M. VanHilst, M.M.Larrondo-Petrie, and S. Huang, "Defining security requirements through misuse actions" , *Procs. of the International Workshop on Advanced Software Engineering (IWASE 2006).* Santiago, Chile, August 2006 (part of WCC)

[Sch06] M. Schumacher, E.B.Fernandez, D. Hybertson, F. Buschmann, and P. Sommerlad, *Security Patterns: Integrating security and systems engineering*, West Sussex, England: John Wiley & Sons 2006.