Threat Mitigation and Software Design Along Lifecycle Phases: Discovering the Optimal Secure Software Design Methodology

Sean Carroll

Villanova University

"This paper or presentation is my own work. Any assistance I received in its preparation is acknowledged within the paper or presentation, in accordance with academic practice. If I used data, ideas, words, diagrams, pictures, or other information from any source, I have cited the sources fully and completely in footnotes and bibliography entries. This includes sources that I have quoted or paraphrased. Furthermore, I certify that this paper or presentation was prepared by me specifically for this class and has not been submitted, in whole or in part, to any other class in this University or elsewhere, or used for any purpose other than satisfying the requirements of this class, except that I am allowed to submit the paper or presentation to a professional publication, peer reviewed journal, or professional conference. In adding my name following the word 'Signature', I intend that this certification will have the same authority and authenticity as a document executed with my hand-written signature.
Signature Sean Carroll"

Abstract

This paper explores current secure software design frameworks in an effort to present the most optimal secure software design methodologies. The articles primarily focus on the issues encountered in practice of such methodologies when threat mitigation is of concern. There appears to be common issues discussed within the current methodologies addressed. Within the lifecycle phases of secure software design, threat mitigation and design of phases from a practical design perspective of a programmer, appear to come into conflict in many areas. The goal or ultimate considerations for design of such software necessitates that security of design is of utmost importance, rather than other ideals. It is found among many companies within research that expectations of users and designers stress reliability, ease of use, and other demands in how various interacting parties use software presented. However, in the effort of secure software design, a methodology that unifies security goals in each phase of the lifecycle with other demands needs to be communicated and integrated in a way that does not impede the progress of software design. Embedding communication in each phase of the secure software development lifecycle (SSDLC) also requires that such methodologies in place do not uproot current design frameworks, but attempt to eliminate the appearance of conflicting efforts in design.

Threat Mitigation and Software Design Along Lifecycle Phases: Discovering the Optimal Secure Software Design Methodology

Current Secure Software Design Methodologies that are developed in academia attempt to optimize current flaws in design with provable design approaches. These designs when put into practice by industry appear to have more conflicts than initially designed. When the designers of such software are communicated the design of such software, it often appears that the communication of such design lacks in its ability to present aligning efforts of security of such software and practical demands of one's role. It is of great importance to seek not a growing divergence of threat prevention and security among the other goals of software design in practice, but to design a methodology that optimizes the communication that such goals are inherently the same when all phases of lifecycle design are considered.

Literature Review

As industry increases expectations in the beginning stages of software design, it is of great importance that design methodologies consider that the current expertise in not present. When current introductory programmers are presented with a task, current development goals favor completion over secure and long-term, viable solutions. As Kahn and Kumar assert in their 2016 research "Embedding Security in Software Development Lifecycle." it is clear that there is a struggle of traditional SDLC to meet all of the demands of all parties within a company. There article states in their concluding remarks that "Various type of SDLC models and security testing techniques were available, but little work has been done in the integration of SDLC with the security factors." There solution to this problem is to identify that in the past, it is true that some models have identified aspects of security within the models that have been designed, however, the phases of the lifecycle have not been addressed individually. With this in mind, Kahn and

Kumar believe that integrating the ideals of a risk and security into the current phases of the current SSDLC is the most optimal design approach. While the two have often had different frameworks, it is important that security and risk are taken into consideration at the beginning of design of software along the SSDLC in order to optimize current issues presented (Khan et al.,2016).

Discussion

The integration of risk considerations and threat mitigation into the software design process is often one that does not meet the demands of the large-scope analysis of design methodologies. It is often that academia fails to consider that if practices of security or design are attempted to be implemented, and those practices are too difficult or time consuming for the employee to consider, it is often that the practices or frameworks of design are circumvented in an unwanted fashion. The separation between the hierarchy of the security professionals with knowledge of secure design practices and those actually designing the software. Ahad, Tariq, Niaz, and Inam, consider this issue within their referenced research. The idea of a "traceable requirement," as in a specific security goal along the phases of secure software design, appears to be the most optimal way of approaching this miscommunication barrier that needs to be addressed. It does not appear unrealistic that such groups could integrate both group by assigning a security advisor to each phase of the design lifecycle phases. However, this integration is one that is not often seen when the software design is conducted under limited amount of funds for the development of such software. A startup in the industry with little funds for software development in one company that is under scrutiny in this proposal of secure software development. The goals of design requirements need to be addressed in some form, and although a security advisor is not something that may be achievable by every designer in industry, it still

holds that traceable requirements of security and threat prevention goals can mitigate misalignment with other requirements of design when traceability is implemented.

Conclusion

The progression towards new and secure software design methodologies is one that necessitates consideration of many influencers of design along the phases of the SSDLC. By providing traceability of requirements and improving communication of desired requirements or goals during. especially the beginning phases of design, an optimal methodology can be design and implemented. It is often the natural of industry to consider other requirements, such as cost and usability of software, over security requirements. In the design of an optimal SSDLC, by improving communication and emphasizing the traceability of requirements and goals in secure design, we are able to design an optimal framework that seeks to mitigate the issues of conflicting design requirements.

References

Ahad, A., Tariq, L., Niaz, S., & Inam, M. (2017). Design of Secure and Traceable Requirement Engineering Process for Security-Sensitive Projects. Journal of Software Engineering and Applications, 10(12), 873–883. https://doi.org/10.4236/jsea.2017.1012049

Karim, N. S. A., Saba, T., & Albuolayan, A. (2017). Analysis of software security model in scenario of Software Development Life Cycle (SDLC). Journal of Engineering Technology (ISSN: 0747-9964), 6(2), 304–316.

Khari, M., & Kumar, P. (2016). Embedding security in Software Development Life Cycle (SDLC). In Computing for Sustainable Global Development (INDIACom), 2016 3rd International Conference on (pp. 2182–2186). IEEE.

van den Berghe, A., Yskout, K., Scandariato, R., & Joosen, W. (2017). A Model for provably secure software design. In Formal Methods in Software Engineering (FormaliSE), 2017 IEEE/ACM 5th International FME Workshop on (pp. 3–9). IEEE.