Editorial

Risk in the Age of Software Security

Martin Gilje Jaatun

SINTEF Digital, Trondheim / University of Stavanger

Editor-in-Chief

For general applications, it is way too costly to aim for 100 % secure software; for complex systems it may even be impossible. To achieve effective software security at reasonable cost, it is thus necessary to identify which parts of the software are more critical regarding security, and determine which activities will be most efficient and effective in securing the software product. All major secure software development lifecycles (e.g., the Software Security Touchpoints and Microsoft's Security Development Lifecycle (SDL)) and frameworks (e.g., BSIMM, OpenSAMM) contain activities directly related to assessment of risk and threats.

Unfortunately, there is no getting away from the fact that risk assessments also take time. In predominantly agile software development organizations, there is a constant struggle between various worthy causes; any activities that are perceived as being too onerous, run the risk of being deprecated if they do not contribute directly to production of new features. There is therefore a need for further research on how to perform risk management more smoothly in agile development; both on the risk assessment itself, and on activities that can mitigate the identified software risks.

Special thanks to IJSSE associate editor Prof. Riccardo Scandariato, who took responsibility for managing the peer review of the paper submitted by my colleague Inger Anne Tøndel, of which I was a co-author. This ensured that any conflict of interest was avoided, and allowed me to support IJSSE with this submission. I would encourage all editorial board members to do likewise, and submit your software security research to IJSSE!

This issue contains three articles. First, Tøndel et al. present an empirical study of risk-related software security practices in a number of public bodies in Norway. They conclude that the majority of activities are actually not risk-driven at all, but rather compliance-driven. The practices observed seem to be applied in a haphazard way, and there is potential for improvement in (among other things) stakeholder cooperation and risk perception.

Then, in "LDAP Vulnerability Detection in Web Applications" Shariar et al. explain how vulnerabilities in the Lightweight Directory Access Protocol can be detected, thus preventing LDAP injection attacks in web applications.

Finally, Misra et al. present a comparison of different complexity measures for software programs in "Analysis of Existing Software Cognitive Complexity Measures". They argue that in order to reduce complexity, it must first be identified, and to do this, an appropriate complexity measure is required. Complex code is very difficult to make secure, and thus reduced complexity is likely to result in increased security.