Collaborating for Secure Software

I have worked with Joe Jarzombek and the Department of Homeland Security (DHS) over the past years and have seen the result of their commitment to software security in a growing number of resources for assistance. While most people have a basic definition of security, I'll share the DHS definition: *Protection against intentional subversion or forced failure. Preservation of the three subordinate properties that make up security – availability, integrity, and confidentiality.*

With all the current efforts to secure software, our nation still encounters attacks. It is estimated that 90 percent of reported security incidents result from exploits against defects in the software design or code. This is especially troubling because many of the vulnerabilities that enable these attacks can be prevented with the use of solid software engineering principles. Given that, why do we still have difficulty ensuring the integrity of software that is so key to protecting the infrastructure from threats and vulnerabilities, reducing overall risk to cyber attack?

In a continuing effort to develop defenses to software attacks and educate software personnel on implementing these defenses, DHS is involved with multiple initiatives. Their primary goals while deploying software security practices include the following:

- Exploitable faults and other weaknesses are avoided by well-intentioned developers.
- The likelihood is greatly reduced or eliminated that malicious developers can intentionally implant exploitable faults and weaknesses or malicious logic into the software.
- The software is attack-resistant, attack-tolerant, and attack-resilient.

To this end, a great deal of attention has been focused on their BuildSecurityIn initiative. By accessing <http://BuildSecurityIn.us-cert.gov>, developers, acquirers, and users can find a broad range of information about best practices, tools, guidelines, rules, principles, and other knowledge to help organizations build secure and reliable software. Through DHS' sponsorship of conferences and workshops, a common body of knowledge and a repository of practical guidance for software developers and architects are being produced to improve the quality, reliability, and dependability of software. In collaboration with industry, academia, and government partners, DHS' approach to addressing software assurance encompasses the following components:

- People – Education and training for developers and users.
- Processes – Practical guidelines and best practices for the development of secure software.
- Technology – Tools for evaluating software vulnerabilities and quality.
- Acquisition – Specifications and guidelines for acquisition and outsourcing.

I am pleased that CROSSTALK continues to be a forum for educating the software community on software security. Some of this month's authors have personally worked with DHS on the topics discussed. We begin with Robert A. Martin's discussion of the Common Weaknesses dictionary in *Being Explicit About Security Weaknesses*. Standards provide consolidated resources for the software community, especially developers, to glean specific techniques for developing secure software. We build on this with *Secure Coding Standards* by James W. Moore and Robert C. Seacord and *How a Variety of Information Assurance Methods Delivers Software Security in the United Kingdom* by Kevin Sloan and Mike Ormerod. Going beyond standards, we begin our discussion of actual practices with Idongesit Mkpong-Ruffin and Dr. David A. Umphress' article, *High-Leverage Techniques for Software Security*. Next, *Baking in Security During the Systems Development Life Cycle* by Kwok H. Cheng emphasizes the DHS philosophy that security must be part of the entire software development process. We conclude with Mel Crocker's insights on certifications and technological advances to enable information sharing across platforms in *Cross-Domain Information Sharing in a Tactical Environment.*

I echo Mr. Jarzombek's challenges in past issues to our readers to take the time to understand and apply these principles and techniques. Through this effort, we can continue to expand the software security community of practice.

Elizabeth Starrett
Publisher