CERT'S PODCASTS: SECURITY FOR BUSINESS LEADERS: SHOW NOTES

Developing Secure Software: Universities as Supply Chain Partners

Key Message: Integrating security into university curricula is one of the key solutions to developing more secure software.

Executive Summary

Software security education is not generally integrated into today's computer science and software engineering university curricula. As a result, many software producers and consumers find themselves in the remedial training business. They are forced to develop courses, processes, practices, and tools to make sure their new software engineers understand how to develop more secure software, given growing marketplace demands. Universities need to start viewing themselves as supply chain partners.

In this podcast, Mary Ann Davidson, Chief Security Officer (CSO) for Oracle, discusses the critical need to educate software engineering students about how to develop more secure software.

PART 1: THE SOFTWARE SECURITY KNOWLEDGE GAP

Many still view secure software as "security" software – products that protect against specific types of attacks. Examples include firewalls, anti-virus, and intrusion detection system products.

Secure software includes:

- security-enabled software where security functionality is embedded in the software product
- more importantly, software that is designed, developed, and delivered with security in mind

Secure software isn't just about the features and functions. It is built using engineering and processes that cause the software to be resilient, aware of security threats, and able to take appropriate mitigating measures when under attack.

Why Secure Software?

Today's IT systems serve as critical infrastructure for conducting business in most organizations. As a result, IT systems and software need to be more secure.

Increased regulations call for software and systems to be more secure.

Software cost of ownership over the life cycle is driving some investment decisions in more secure software. The unpredictable cost and schedule impact of patching systems, for both suppliers and users, is a growing concern.

There Is a Knowledge Gap in Today's Software Engineers

Most engineering disciplines require their students to understand what it takes to make a design or structure resilient and fault tolerant, such that it doesn't fail. It's part of the mindset.

This is not the case for software engineers. Making their software safe, secure, and reliable – and being personally accountable for this – is not part of their thinking and practice.

Someone in a CSO position should not have to explain to a software engineer what a buffer overflow is and how to prevent it.

Software engineers, and the universities from which they graduate, are part of the supply chain for organizations that develop software.

If hiring organizations do not get a good supply chain product – a good security-aware, educated developer – it's hard for the organization to produce secure software.

PART 2: UNIVERSITIES AS SUPPLY CHAIN PARTNERS

Reaching Out to Universities

Mary Ann sent letters to the top 10-12 universities that Oracle recruits from, addressed to the department heads or deans of computer science and engineering. [See also Mary Ann's blog on this topic.]

She described how much Oracle is spending to do remedial training with new software engineers on the fundamentals of developing secure software.

The marketplace, both commercial and government, is demanding that software vendors provide more secure products. For organizations like the U.S. Department of Homeland Security (DHS), this is a national security concern.

Her letter cited DHS efforts in software assurance and the SANS secure coding certification as examples of the required expertise.

She also stated "In the future, we are going to change our recruiting practices to focus on universities who do place an emphasis on educating undergraduate and graduate students in secure software engineering principles."

Unfortunately, only one university responded, seeking funding to add software security to their curriculum.

Identifying Qualified Institutions

Some universities are starting to integrate software security disciplines into their curriculum. [One place to start is the U.S. National Security Agency Centers of Academic Excellence in Information Assurance Education program.]

Mary Ann's intent is that Oracle start recruiting from these institutions.

Software Vendor Action

The good news is that many of the mainstream software vendors are beginning to seriously address software security.

A Broader Solution

One solution is to work with organizations that accredit computer science program [such as CSAB] to make sure that software security principles and practices are part of the accreditation criteria.

PART 3: RE-ENGINEERING THE CURRICULUM; PRACTICES TO DEPLOY IN THE MEAN TIME

Topics to Include in the Curriculum

Include the basics of secure coding in all relevant courses that address software development. This is analogous to requiring civil engineers to always understand statics and the basic physics of building as part of the fundamentals of their field.

Include courses on:

- security architecture, threat modeling, and design
- good secure programming practices
- effective uses of automated tools, for example, to perform code analysis. Manual code reviews do not scale for today's large, complex systems.

Ultimately, all software engineers should adopt a mindset that "my code is going to be attacked. I need to design, build, and deploy defensively. Otherwise, my software will fail."

Practices to Deploy in the Mean Time

Develop templates for functional design and test specifications that include security, and make sure these are used on every project.

Develop secure coding practice standards that describe both what not to do and how to handle a particular problem the correct way. Make sure everyone takes the basic course, including project and release managers.

Develop and institute security checklists at key project milestones such as software release into production.

Use automated tools to support all of the above.

Create an ethical hacking team to not only find bugs but also to transfer their knowledge to the development team.

Create security points of contact with particular areas of expertise that the development team can call upon.

Collect metrics on software defects, not to be used in a punitive way but to help teams that are having problems. Training and tools may need to be improved.

Roles and Responsibilities

The CSO/CISO (chief information security officer) should not be responsible for software security. Like all other software quality attributes (for example, performance and portability), security is the responsibility of the development team.

The best way to succeed is to provide processes, practices, and tools that support secure software development, along with measures and rewards. The idea is to make it easy for people to do the right thing.

The CSO office is responsible for governance, and for ensuring that developers understand that they are personally responsible and accountable for the security worthiness of every line of code that they write.

Ultimately, the organization needs to establish software security as a cultural norm.

Resources

Addison-Wesley Software Security Series

U.S. DHS Software Assurance Program's Build Security In web site

U.S. DHS Software Assurance Program/Forum

Allen, Julia; Barnum, Sean; Ellison, Robert; McGraw, Gary; Mead, Nancy. *Software Security Engineering: A Guide for Project Managers*, Addison-Wesley, 2008.

CERT podcast: Building More Secure Software

CERT podcast: Identifying Software Security Requirements Early, Not After the Fact

CERT podcast: How to Start a Secure Software Development Program