Chapter V

Software Standards, Reliability, Safety, and Risk

Introduction

Software, more than anything else, is at the heart of the information communication infrastructure. It is in fact one of the three main components of the infrastructure, together with hardware and humanware, as we discussed before. Being at the core of this infrastructure we all depend on implies the importance we must put on the software component. In this chapter, we are going to focus on this role and how we can keep software safe, dependable, and secure, as we struggle to make the information communication infrastructure secure. For the remainder of this chapter, we are going to focus on the quality of the service of software products, causes of software failures, developer and buyer protection, and techniques for improving software quality.
The Role of Software in the Security of Computing Systems

Because of our growing dependence on software systems and the increasing complexity of software systems, many criminals, terrorists, nation-state adversaries, and malicious and recreational attackers increasingly are targeting software systems to discover and exploit inherent software vulnerabilities. To ensure the safety, reliability, and integrity of software-based systems, we need to include security provisions and mechanisms within the software as it is being developed. This points to a formal approach to security enhancement in the software life cycle.

According to Jarzombek and Mercedes (2006), we need to develop processes, methods, and techniques that software developers can deploy early in software development to specify, design, implement, configure, update, and sustain software that is able to accomplish the following:

- Resist or withstand many anticipated attacks and
- Recover rapidly and mitigate damage from attacks that cannot be resisted or withstood.

To achieve this, Jarzombek and Mercedes (2006) suggest a shift in emphasis and scope that will affect the life cycle in the following ways:

- **Requirements:** During the review of the software’s functional requirements also emphasize security vulnerabilities and risk.
- **Design and implementation:** Pay special attention and scrutinize design assumptions and choices that determine how the software will operate and how different modules/components will interact. Where possible, adjust such designs in order to minimize the exposure of those functions and interfaces to attackers. Select only those languages or language constructs, libraries, tools, and reusable components that are determined to be free of vulnerabilities or for which effective vulnerability mitigation can be realistically implemented.
- **Reviewing, evaluating, and testing:** Design and make the security criteria part of every specification, design, and code review; evaluation and selection of acquired and reused components; and all of the software’s
Related Content

HIPAA: Privacy and Security in Health Care Networks
[www.igi-global.com/chapter/hipaa-privacy-security-health-care/23255?camid=4v1a](www.igi-global.com/chapter/hipaa-privacy-security-health-care/23255?camid=4v1a)

Impact of Bank Operational Efficiency Using a Three-Stage DEA Model
[www.igi-global.com/article/impact-of-bank-operational-efficiency-using-a-three-stage-dea-model/120556?camid=4v1a](www.igi-global.com/article/impact-of-bank-operational-efficiency-using-a-three-stage-dea-model/120556?camid=4v1a)

Developing Trust for Electronic Commerce
[www.igi-global.com/chapter/developing-trust-electronic-commerce/24597?camid=4v1a](www.igi-global.com/chapter/developing-trust-electronic-commerce/24597?camid=4v1a)

Breaching Security of Full Round Tiny Encryption Algorithm
[www.igi-global.com/article/breaching-security-of-full-round-tiny-encryption-algorithm/190859?camid=4v1a](www.igi-global.com/article/breaching-security-of-full-round-tiny-encryption-algorithm/190859?camid=4v1a)