Managing Software Security Risks

Gary McGraw, Cigital

Most organizations manage computer security risk reactively by investing in technologies designed to protect against known system vulnerabilities and monitor intrusions as they occur. However, firewalls, cryptography, and antivirus protection address the symptoms, not the root cause, of most security problems. Buying and maintaining a firewall, for example, is ineffective if external users can access remotely exploitable Internet-enabled applications through it.

Because hackers attack software, improving computer security depends on proactively managing risks associated with software and software development. The current "penetrate and patch" approach of fixing broken software *only after it has been compromised* is insufficient to control the problem.

System designers and developers must take a more proactive role in building secure software.

A GROWING PROBLEM

About 20 new security holes are reported each week in postings to the BugTraq mailing list (http://online.securityfocus.com/archive/1), and this number is growing. Even "tried and true" software is not as safe as once thought—people continue to expose vulnerabilities that have existed for months, years, and even decades.

Why are modern computing systems susceptible to software security problems? Software is not necessarily any worse than it has ever been, but three major trends have changed the risk environment in which it exists.

Connectivity

Growing Internet connectivity has increased the number of attack vectors as well as the ease of exploiting software. As people, businesses, and governments become more dependent on the network-enabled communication that information systems provide, they become vulnerable to attacks from distant sources. Malicious hackers no longer need physical access to a system to exploit its software.

Extensibility

Sun Microsystems' popular Java platform and Microsoft's new .NET Framework are designed to easily accept mobile code updates and extensions that let system functionality evolve incrementally. Operating systems support extensibility through dynamically loadable device drivers and modules; applications such as word processors, e-mail clients, spreadsheets, and Web browsers do so through scripting, controls, components, and applets. Preventing software vulnerabilities from slipping in as unwanted extensions is a major challenge. Understanding how a system may be extended in the future is essential to getting a handle on system risks.

Complexity

Preventing vulnerabilities from compromising an operating system depends on the proper functioning of both the kernel and end-user applications. The more complex systems and applications become, the harder it is to avoid bugs. Use of unsafe programming languages such as C and C++ that do not protect against buffer overflows and other simple kinds of attacks exacerbates the problem.

The complexity of operating systems has increased dramatically during the past decade. For example, Microsoft's Windows XP has 40 million lines of code compared to the 3 million in Windows 3.1. Formally analyzing today's simplest desktop systems to prove that they are secure is impossible, not to mention the enterprise-wide systems that businesses and governments use.

SOFTWARE RISK MANAGEMENT

Many software vendors incorrectly equate security software—often a set of add-on features such as cryptography—with software security. However, software security is an emergent property of a complete system, not a feature. After a product has been publicly compromised, vendors scramble to rush out a patch that ironically serves as an attack map for exploiting unpatched