

Preventing Security-Related Defects

Lauren Heinz

Shawn Hernan, a team leader with the CERT Coordination Center[®] at the SEI, spends a good part of his day tracking and analyzing the vulnerability reports he receives from software vendors who have discovered security-related problems in their products. Since January, Hernan and his team have logged more than 1,600 new software vulnerabilities—a staggering number for any system or network administrator to evaluate and possibly act on. The current system of issuing copious warnings and expecting organizations to keep up, Hernan says, is quickly becoming unrealistic.

“It is nearly impossible, and extremely time consuming, for anyone to stay current on these reports and patches. Just reading through them and seeing which ones affect you can take weeks,” Hernan says. “We need to look more at preventing them in the first place.”

In an effort to help organizations reduce vulnerabilities in software, Hernan has teamed up with members of the Team Software ProcessSM (TSPSM) Initiative at the SEI to consider how

improved software quality methods such as the TSP might help developers detect and prevent security defects earlier in the software-development life cycle. Hernan has also developed a list of the five common causes of software defects, which he hopes will help organizations begin to improve their security practices.

TSP and Quality Software

According to CERT Coordination Center statistics, more than 90% of software security incidents are caused by attackers exploiting known software defect types.

The TSP provides a framework, set of processes, and collection of disciplined methods for producing quality software. With TSP, software teams can produce near defect-free software with typically fewer than 0.1 defects per thousand source lines of code (KSLOC). Given that a vulnerability is a flaw or defect in a piece of software that allows an intruder to read, modify, or disable the way that software functions in a system, using TSP could yield significant results. “We know that the TSP reduces overall software defects. We want to see if it can reduce security defects in software,” says Noopur Davis, a member of the TSP team who is working with Hernan. “The question is, how do you design, implement, review, and test software for security?”

The answer lies, in part, in extending TSP for secure systems. Davis said this extension of TSP would support secure software-development practices, help predict the likelihood of security defects by creating specific, security-related measures, and be dynamically tailored to respond to new threats. In the coming months, Davis and her team will explore how applying the TSP can

help developers identify and remove vulnerabilities during the early stages of software design and development.

Lack of Understanding?

Why are so many software vulnerabilities being reported? The problem, Hernan and Davis agree, is not a lack of available data, knowledge, training, or support regarding security practices. Incident response centers (such as the SEI's CERT Coordination Center), guidelines, checklists, best practices, and other useful materials are widely available. "I have come to believe that a vulnerability is not so much a property of software as it is a property of an organization," Hernan says.

Meanwhile, Davis is curious to see how simply applying TSP might help organizations eliminate more defects, including potential vulnerabilities. For example, most operating systems and commercial software applications are known to have more than 2 defects per KSLOC, or 2,000+ defects per million SLOC. Even if only 5% of these defects are potential security concerns, there are 100 vulnerabilities per million SLOC. "That is still a huge number of defects," she says. "There is significant potential here for TSP to reduce those numbers."

Five Common Causes

Although there are thousands of known software vulnerabilities, Hernan has been able to categorize them into five main causes (listed below). In addition to realizing potential

improvements through applying disciplined processes, organizations can have a better chance of creating secure applications and systems by programming software with these causes in mind:

1. **Buffer overflows.** These occur when a programmer does not properly check whether there is adequate space to hold data entered by a user. An intruder can exploit this vulnerability by intentionally entering excess data, which interrupts the program's normal operations and allows the intruder to take control.
2. **Timing windows.** Problems with timing windows occur when a temporary file is exploited by an intruder to gain access to the file, overwrite important data, and use the file as a gateway for advancing further into the system.
3. **Insecure default configurations.** These occur when vendors use known default passwords to make it as easy as possible for consumers to set up new systems. Unfortunately, most intruders know these passwords and can access systems effortlessly.
4. **Bad protocols.** Some protocols, or the standards by which information is exchanged over the Internet, lack any security at all. For example, unsolicited commercial email, commonly referred to as "spam," is the irritating result of poor protocol programming.

5. Trusting untrustworthy information. This is usually a problem that affects routers, or those computers that connect one network to another. When routers are not programmed to verify that they are receiving information from a unique host, bogus routers can gain access to systems and do damage.

“If you can develop software that prevents these top five categories, you are going to prevent thousands of potential vulnerabilities,” Hernan says.

Next Steps

Davis and Hernan are currently seeking organizations to take part in pilot projects to help develop specific technical solutions—new design and implementation practices—for security, review methods and checklists, and tools. If your organization is interested in participating, please contact the SEI.

For more, contact—

Customer Relations

Phone

412-268-5800

Email

customer-relations@sei.cmu.edu

World Wide Web

<http://www.sei.cmu.edu/tsp?si>

<http://www.cert.org>

® CERT Coordination Center is registered in the U.S. Patent and Trademark Office.

SM Team Software Process and TSP are service marks of Carnegie Mellon University.