Open Source Security

U.S. Symposium/ITxpo

17—22 October 2004
Walt Disney World
Lake Buena Vista, Florida

John Pescatore

These materials can be reproduced only with Gartner’s written approval. Such approvals must be requested via e-mail — quote.requests@gartner.com.
Open Source Security

Strategic Planning Assumption: If only a half percent of software vulnerabilities were removed prior to production use for purchased and internally developed software, enterprise configuration management costs and incident response costs would be reduced by 75 percent each (0.8 probability).

Open Source Has Security Advantages, but Popularity Breeds Conflict

- Closed source doesn’t mean more secure
  - Neither does open source
  - Competition does mean more secure
- Open source software gets more secure faster
- A trusted supplier is the most-important factor
  - A trustable computing platform is a close second
  - Skilled system administrators not bad either
- The mass attack tipping point is driven by the desktop

Software engineering is still an oxymoron. It lacks the standards, handbooks and formulas that turn a craft into an engineering discipline. Safety and security are prime examples of the problem — software producers have never considered the concept of safety when building software, and only recently have even begun to discuss security. Software developers have hidden behind excuses that software is too complex to ever be perfect, therefore it can’t be secure.

Emphasizing security during the development of software is the most-important predictor of the security of the final product. Open source development of software doesn’t mean more emphasis on security necessarily — a rigorous development process is still needed. However, when developers know source code will be made public, they often take fewer shortcuts in the code and don’t leave "Easter eggs" or other surprise bits of code. The open source review and upgrade process does lead to software becoming more secure, faster — if the open source process is well managed, and clue-full reviewers are encouraged and rewarded.

Random open source programs shouldn’t be trusted any more than random stock market hints would be — know your supplier. Trained system administrators are also required for open source, just as for proprietary software.
Open Source Security

By eliminating or shielding vulnerabilities in software, all forms of attack are avoided — it doesn’t matter whether it is vandalism, hactivism, cybercrime or information warfare.

New Rules/Realities

Every day, new vulnerabilities are reported. The Carnegie Mellon Software Engineering Institute’s (www.cert.org) computer emergency response team reported 3,742 serious vulnerabilities in 2003 — actually less than in 2002. However, 75 percent of these security holes are patched by vendors before any actual attack occurs, and less than 10 percent of listed vulnerabilities are exploited. However, making sure one’s enterprise is not part of that 10 percent requires implementing a process to monitor vulnerability reports and ensuring that all production systems have installed the latest patches.

The largest attack risk to enterprises comes from targeted attacks; insiders will always be the source of more than 60 percent of targeted attacks. It is often impossible to have a firewall between an insider and a vulnerability. The only solution to security is to eliminate the vulnerability.
Most vulnerabilities are known about long before the attack happens. However, it is tremendously expensive and complex for enterprises to patch their software on every desktop and server. Most patches often require patches themselves, and many patches (at least 30 percent) break at least one enterprise applications. To many IT operation organizations, patching is seen as more risky than getting attacked.

The security industry likes to play up the possibility of “day zero” attacks — attacks that exploit vulnerabilities that no one knew about before the attack. Gartner believes that day zero attacks will represent less than 5 percent of successful attacks through 2008, as attackers continue to focus on reverse engineering patches to develop exploit code.

Software developers need to focus first on reducing configuration errors — making default “out of the box” configurations more secure, than on reducing “attack surfaces” — external interfaces that provide traction for attacks.

**Action Item:** The single-most-important action for increasing Internet security is to stop using software that has frequent security vulnerabilities. Barring that, enterprises have to greatly increase the resources they apply to ensuring that all software is safely configured and patched.

---

We Have Met the Enemy, and They Are Us — But 'Day Zero' Is Not

By 2006, the percentage of vulnerabilities that are attacked within one month of patch release will double from 15% in 2003 to 30% (0.7 probability)

Gartner
Strategic Planning Assumption: Enterprises that implement vulnerability management will experience 90 percent fewer successful attacks than those that make only an equal investment in intrusion detection systems (0.8 probability).

A vulnerability is a weakness in process, administration or technology that can be exploited to compromise IT security. Vulnerability management is a set of processes and technologies that are used to establish and maintain a security configuration baseline; discover, prioritize and mitigate exposures; establish security controls; and eliminate root cause. Vulnerabilities can be present in any layer of the application stack, and can be caused by weaknesses in just about every IT administration, process or design function. Vulnerabilities can be introduced by a wide variety of IT professionals, network and security administrators can introduce flawed firewall filters that expose vulnerable services that can be exploited, system administrators can misconfigure systems that leave critical resources unprotected or activate vulnerable services, flaws in user administration can result in inappropriate levels of system access and application developers can introduce flaws that can be exploited to gain control of systems. A layered approach to IT security that uses strong perimeter protection in combination with general system hardening is required to secure a modern IT environment from external and internal threats.
No commercial software is secure out of the box — although some are more secure than others. Most operating systems are general-purpose software that support many functions and services that should be disabled or closely monitored when used outside the firewall or in sensitive situations.

The first level of security is using security configuration checklists (see http://www.cisecurity.org) to turn off dangerous services. The next level up is using tools for hardening an operating system so that in addition to disabling risky functions, additional security measures and mechanisms are deployed to monitor security and enforce policy. Products such as Argus PitBull provide this level of security.

The top of the pyramid is a trusted operating system. In this case, the software is developed to implement a well-defined policy (based on a verifiable security model) with design constraints that implement security first and functionality second, and it has been evaluated by a third party using an approved structured testing methodology. There is no trusted version of Linux yet, and trusted versions of Unix are always several versions behind the commercial version and come with a greatly increased cost of ownership.
One critical element in maintaining a secure network infrastructure is minimizing operating system diversity and developing standard operating system configurations. A common question Gartner receives is: “Which operating system is the most secure?” The answer is “Out of the box, none of them.” Operating system hardening, through the use of tools such as Argus PitBull or through simple minimization of available services, is required for any server connected to any network.

The mainstream Unix operating systems (Solaris, HP-UX and AIX) had represented the lowest risk choices, because they have had the most security testing and most of the major vulnerabilities have been addressed. Windows 2003 has had sufficient operational testing to be suitable for security-critical applications. Linux is a mixed bag from a security perspective: Linux vendors that invest in managing the open-source process and attract sufficiently “clue-full” reviewers will produce products that get more secure faster than proprietary operating systems. However, merely repackaging source code and splashing “Open Source” on shrink-wrapped boxes does not increase security.

**Action Item:** Every server connected to any network should be periodically audited to ensure compliance with operating system configuration guidelines.
Strategic Planning Assumption: By 1Q04, Windows-based servers that are exposed to the Internet will require approximately equal administration time per server to maintain acceptable levels of security as Unix-based servers (0.7 probability).

Through year-end 2001, a Microsoft Web server was almost three times as likely to suffer defacement as a Unix-based Web server. The recent Slammer worm (in January 2003) hit Microsoft database servers much harder than any previous attack against vulnerabilities in Oracle’s competing database software. The common thread between the attacks, however, was the rate of patches released to mitigate software defects. System administrators simply can’t keep up when there is an endless series of serious patches to install.

Another common thread or factor is that in the Windows server world, more boxes are required to provide the same performance and reliability of an equivalent Unix-based configuration. This acts as a multiplier — not only are there more patches, but also there are more servers to patch.

The final factor is that many end-user PC experts have the skill to put up a Windows-based server, but they can’t install and deploy Unix-based servers. The IT organization has a much harder time keeping “rogue” Windows servers off their networks than they do Unix servers.

Action Item: When you deploy Internet-facing servers, ensure that all software configurations have the most-current patches installed and have been configured to meet security policies.
In 1999, spam was a security problem. By 2003, spam became more than 50 percent of e-mail traffic and the problem changed into an operational e-mail problem. Messaging managers were facing the need to buy more e-mail servers and more storage to deal with messages that their users didn’t even want. Once spam passed 50 percent of e-mail volume, the problem changed from “find the spam and block it” to “find the real messages and let them through.”

Network traffic is nearing a similar tipping point. For many enterprises 30 percent (often 50 percent to 80 percent in university environments) of network traffic are worms, file sharing or peer-to-peer traffic that isn’t officially allowed or other malicious traffic. Network managers will soon be forced to add capacity for no good business reason, just to make up for the capacity lost to this traffic.

Similarly, as Linux gains market share it will hit a tipping point, where attackers find it to be a more-fruitful target for attacks. On desktops, a 30 percent market share will represent that tipping point, but that will not occur for at least five years. When Linux reaches 50 percent server market share, there will be a similar tipping point. However, the impact of attacks against Linux will always be less than that of similar attacks against Windows through 2008, because Windows includes many applications that have been tightly integrated into the basic OS code.
Training for system administrators is even more important for open source applications.

---

**Linux Server Security Recommendations**

- Train those system administrators
- Have Linux configuration standards ([www.cisecurity.org](http://www.cisecurity.org)) and a vulnerability management process
- Prioritize vulnerability management and log consolidation over host-based IPS for Linux through 2006
  - HIPS for hi-value servers now
- Security as key tradeoff criteria in RFPs for all applications to run on Linux

---

Most security recommendations for Linux-based servers are identical as those for Unix or Windows servers. Standard Linux security configuration templates should be developed and system administrators should be trained in best security practices. Linux servers should be included in all vulnerability management and configuration management processes.

Attacks against Linux servers are less common and of lesser impact than those against Windows servers. The biggest risk for Linux servers is misconfiguration and invalid user accounts. Through 1Q06, host-based security investments for Linux servers should focus on identity and vulnerability management, and frequent auditing vs. intrusion prevention.

**Action Item:** For all procurement of Linux or any open source applications, security should be a key criteria in all RFPs and RFIs, and used in the acceptance criteria for any funded development.
Linux desktops should be moved as close as possible to full lockdown.

Linux Desktop Security Recommendations

- Personal firewall on every laptop
- HIPS by 2007
- Don’t repeat the mistakes made in the PC generation:
  - Think lockdown, dumb terminal, SSL, VPN, server-based computing where ever possible
  - Don’t fall for the OS + integrated applications trap again
  - Scan and block, scan and block

If you are moving to Linux desktops, from a security perspective you have a rare opportunity to start over and do things right this time. Start with making sure every laptop has a personal firewall installed. Most Linux distributions have capabilities for simple personal firewall capabilities, such as the Redhat Secure Level Configuration Tool or the GNOME Lokkit. As Linux desktop use accelerates, the major personal firewall vendors will have the leading offerings. Host-based intrusion prevention for desktops will not be mature until YE05 and not required on Linux desktops until 1H07.

After getting personal firewall capabilities integrated, aim to keep your Linux desktops as close to locked down as possible — minimize all local applications and look to thin-client solutions or server-based applications as the dominant form of computing.

All Linux PCs should be integrated into scan-and-block approaches — on connection scan the connecting PC to determine it meets security policies. If not, deny connection or isolate the PC onto a safe VLAN segment.
Operating systems are the only types of open source software that will be used in the enterprise. Database management systems, office productivity applications and even security applications (such as Nessus and Snort) are in widespread use already and will continue to grow. Open source software should generally be treated like any other application software:

- Only use software from trusted suppliers — random applications downloaded from random sources will likely have random malicious capabilities mixed in.
- Integrate all open source applications into vulnerability management, configuration management and software distribution approaches to minimize unpatched exposure.
- Host-based intrusion prevention should be used on all Internet-facing servers, whether proprietary or open source.
- All RFPs or RFIs procuring any software should require evidence of security testing. Any open source software downloaded should be tested as well.
## Software Vulnerability Testing Approaches

- **Vulnerability Scanning**: Cenzic, eEye, Fortify, Kavado, Sanctum (Watchfire)
- **Code Testing**: @Stake, Coverity, Fortify, HBGary, Ounce Labs
- **Development Environment Integration**: Sanctum (Watchfire), SPI Dynamics
- **Consulting**: Cigital, Security Innovation, Secure Software, Aspect Security, @Stake
- **Service**: Reasoning

Recently, vulnerability scanner vendors such as Sanctum Technologies and SPI Dynamics have released application-level vulnerability test software that can be easily integrated into software development environments. By integrating software vulnerability test functions into the development environment shell, developers don't have to become experts in security to perform useful IT security tests. Conversely, by embedding coding knowledge into security-testing tools, IT security personnel don't have to become experts in software development.

These software test tools are still first-generation technology. As part of its Secure Windows Initiative, which began in 2000, Microsoft started to use security test tools in its development of Windows Server operating system software. Buffer overflow vulnerability announcements still are a common Microsoft event in older code, but by moving security testing further up the development cycle, we believe that Microsoft Windows Server 2003 will reach security stability more quickly than did Windows 2000 or Windows NT.

For enterprises that don’t have the capability to procure and integrate security-testing tools, external services can be used.
Trustable Computing Platforms

- Changes to the Windows Kernel (NGSCB) and the Intel CPU hardware (LaGrand) will provide a Trustable Computing Platform by 2008, some capability in 2006
- NX support helps fight buffer overflow attacks in 2005
- Linux needs these capabilities on PCs or migration to Linux would be a net security negative

The recent push by Microsoft, Intel and other vendors for trusted computing platforms (see www.trustedcomputinggroup.org) is largely driven by the software and content industries' need to fight piracy and online file swapping. The architectural features required to support digital rights management are also needed to implement a trusted execution environment. Stopping a password sniffer or key-stealing program isn’t much different from stopping a program that copies a streaming audio or video file.

From a PC user’s perspective, there are many aspects of trust. Having software do what users expect it to do is important. It’s also critical for PCs to keep users safe from some of their own dangerous actions. However, to the PC hardware and software industries, trustable computing platforms primarily are PCs that prevent unauthorized software from taking unauthorized actions, with which authorization is mainly defined by licensing agreements for software or other digital content. The key to providing this level of assurance is to have the standard PC operating system and processor platform support a trusted execution environment, which is a critical requirement for a trustable computing platform.

Gartner uses the phrase "trustable computing," rather than "trusted computing," because a trusted execution environment on Windows or Linux -based PCs and servers does not equal a fully trusted computer platform. Predictability, reliability and safety are other key requirements that must be provided to enable trustable computing.
Look outside the office door of most CIOs and you see a wall full of service-level performance charts. Security is glaring by its absence. Everyone agrees that you can’t improve what you can’t measure, yet year after year security groups avoid metrics and benchmarks and other hard measurements.

Gartner has defined six key metrics that provide a baseline for measuring the status and trend of any enterprise’s information security program. These all represent values that are relatively easy to obtain, even though some are difficult to improve. Each enterprise will need to define an acceptable service level for each value, and track performance against that service level. Based on business demands, enterprises can also weight each of the six metrics to create a “blended” index to have a simpler, yet less-meaningful security score.

There are many other specific security metrics that can be applied as driven by particular business drivers. However, the six listed here should be the starting point, because the ability to answer those questions will provide the data needed to create more-detailed metrics.
Open Source Security

Recommendations

• Grab software vendors by their security features and their hearts will follow.
• Move security testing up the food chain.
• Start at the application server tier and design software security architectures outward.
• Don’t trust PCs just because you don’t have code running on them.
• Eliminate the root cause — why are you still using software that is killing you?
This is the end of this presentation. Click anywhere to continue.