More Bang For the Bug
An Account of 2003’s Attack Trends

We can find considerable information security debris in the wake of 2003’s attack trends and new security flaws. New and serious vulnerabilities were discovered, disclosed, and subsequently exploited in many ways—from simple, straightforward methods to more advanced and innovative exploitation techniques.

In this issue’s Attack Trends, I will examine a handful of the more than 3,000 unique vulnerabilities and 115,000 security incidents reported in 2003 (according to CERT Coordination Center’s report for quarters one through three; www.cert.org/stats/cert_stats.html) and do my best to predict information security woes for 2004. My analysis will focus on the distinguishing characteristics of the 2003 attacks trends rather than on specific vulnerabilities or a precisely defined taxonomy of security bugs.

The year of the worms
Worms affected our daily routines more than once during 2003, more than once our online and offline activities were influenced by the adverse consequences of wild exploitation of vulnerabilities on the information technology infrastructure we rely on to work, conduct business, and enjoy our leisure time. The 2003 worms increased the rate and scope of such problems at a scale we have not experienced for decades.

Slammer
The Slammer/Sapphire worm’s outbreak in January 2003—and its ability to spread across network and geographic boundaries at the fastest rate ever seen in the wild—brought immediate attention to the six-month-old vulnerability that it exploited: a buffer overflow vulnerability in an SQL database engine component shipped with Microsoft’s database server product. Interestingly enough, the engine component is required in some of the company’s workstation software packages as well.

The amount of damage Slammer caused prompted many information security practitioners to take a closer look at the worm’s methods of vulnerability exploitation, target selection, spread efficiency, payload functionality, and its side effects. (See Nicholas Weaver et al’s “Inside the Slammer Worm” article in the July/August 2003 issue, pp. 33–39, for a detailed analysis of the worm; www.computer.org/security/v1n4/j4wea.htm.)

SoBig
The first six months of 2003 saw several variations of SoBig spread over the Internet with different effectiveness rates; SoBig.F, the worm’s most complex and effective variation, began propagating in August. The worm took direct aim at unsuspecting users’ email and the software required to read it—the mail user agent, the most common software on workstations. It spread through mass mailing and used infected systems as part of an army of denial-of-service (DoS) attacks or email spamming activities. SoBig also included the ability to update or replace itself with software components suitable for illicit or questionable activities such as laundering network connections, facilitating unsolicited email campaigns by relaying spam email, and generating network traffic aimed at producing network outages of hypothetical victims of DoS attacks. This worm pointed clearly at the close relationship between software bug exploitation and the use of computer systems for illicit or questionable “business” practices. My department coeditor Elias Levy analyzed all known variants of the worm at that time in “The Making of a Spam Zombie Army: Dissecting the SoBig Worms” (July/August 2003, pp. 58–59).

Blaster
Following on the heels of SoBig, the Blaster worm wreaked havoc in August on systems running the Microsoft Windows operating sys-
tem. It exploited a pervasive vulnerability in the Remote Procedure Call (RPC) interface to the Distributed Computing Object Model (DCOM) component discovered less than a month earlier (http://lsd-pl.net/special.html), which speaks volumes about an attacker’s ability to develop exploit programs for known vulnerabilities at a faster pace and about the modularity of attack tools that make it possible to easily use new attack vectors. Blaster introduced yet another interesting behavior—it used compromised systems to launch a DoS attack against Microsoft’s Windows Updates site, which hosted the software patches that would have prevented exploitation of the RPC DCOM vulnerability Microsoft had recently announced. Blaster’s author designed the worm to not only directly compromise systems but also use those newly acquired assets to prevent other systems from protecting themselves. This signals a marked interest in ensuring the worm’s survival by considering both the technical aspects of the vulnerability exploited and the organizational response to the attack.

**Welchia/Nachi**
The supposedly benign Welchia/Nachi worm emerged shortly after Blaster, its evil counterpart. This worm exploited several vulnerabilities and attempted to clean Blaster from compromised systems by downloading and installing the patch to prevent further infection. However, its net effect was rather adverse, provoking network and system outages due to traffic consumption and patch installation failure.

Since then, several similar vulnerabilities and attack vectors that can be used in highly effective worms have been disclosed in the same or closely related software components to the ones Blaster exploited, which leads me to believe that the worm saga is far from its last episode.

**What did the worms teach us?**
The lessons learned from 2003’s various worms are numerous:

- Vulnerabilities left unpatched can and will be used against you. Diligent patch deployment is necessary but insufficient—you must take preemptive measures to prevent exposure of unsuspected systems and detect and contain attacks that exploit vulnerabilities that are not yet publicly known vulnerabilities. Some of the workstations Slammer compromised had not been considered vulnerable because they were not running MS SQL server.
- You cannot see the big picture by merely understanding a vulnerability’s technicalities. You must learn the methods and techniques attackers use to exploit vulnerabilities, leverage access to compromised systems, and escalate and amplify the attacks across several systems and networks to better assess an organization’s risk. You must bring the attackers’ perspective into your analysis.
- Attackers and their tools are getting increasingly sophisticated; the most damaging and fast-spreading worms of 2003 had a modular design, a very specific attack logic, and deployed multi-purpose “agents” that they used to update themselves automatically, launch DoS attacks, send spam at a massive scale, and several other functions. This increased sophistication in worm code was pointed out in “An Analysis of the Slapper Worm” (January/February 2003, pp. 82–87).
- Even in the most benign cases, worms caused significant damage due to the network traffic they generated and logic flaws in their attack selection and infection algorithms. While some vulnerabilities and their exploitation techniques are more suitable for worms than others, all had adverse side effects.
- Active exploitation of vulnerabilities in the wild affects an organization’s ability to conduct normal operations even if its own infrastructure is immune to that specific vulnerability. A large-scale attack’s effects on other systems will inevitably produce collateral damage. In “Crossover: Online Pests Plaguing the Offline World” (November/December 2003, pp. 71–73), Elias Levy provided an extensive account on how the 2003 worms affected, both directly and indirectly, the critical infrastructure we rely on to live normally every day.

The outbreak of worms in 2003 also can be seen as the adoption of known efficiency principles and common sense in attack trends that I will explore in the following section.

**The principle of less effort**
Optimizing costs, achieving greater efficiency, and applying the minimum necessary effort to accomplish your goal are not foreign concepts to

---

**The 2003 worms affected, both directly and indirectly, the critical infrastructure we rely on to live normally every day.**
Attack Trends

buck” philosophy. Therefore, it is not difficult to identify the same approach in the vulnerability exploitation techniques and attack trends seen in 2003.

We can easily explain the appearance of many and more efficient worms as a result of attackers’ attempts to maximize their bang for the “bug”: Compromise a very large number of systems with minimal effort.

The use of mass-mailing techniques as an attack vector and aiming attacks at workstation software also supports the principle of less-effort. In “The Weakest Link Revisited” (March/April 2003, pp. 72–76) I touched on the concept of achieving a higher degree of effectiveness by exploiting widely deployed software on workstations that users who are not exactly security savvy control and use.

The steadily increasing amount of cross-site scripting and SQL injection vulnerabilities discovered and disclosed during 2003 point to another path of less resistance into vulnerable networks. These vulnerabilities have rather simple ways of exploitation, yet they provide casual attackers with a high yield: Direct access to internal networks, compromise of database servers and their content, and indirect ways of attacking unsuspecting users of third-party systems.

More-sophisticated, harder-to-spot, and, allegedly harder-to-exploit vulnerabilities such as the ones that Dave Ahmad described in “The Rising Threat of Vulnerabilities Due to Integer Errors” (July/August 2003, pp. 77–82) could become suitable attack vectors if exploitation of such bugs yields promising benefits.

You can find an example of such a rewarding outcome for an attacker dedicating time and effort to exploit a “hard to get” vulnerability, a bug for which writing exploitation code requires a considerable amount of technical skill and time in the November 2003 report of the compromise of a popular Linux distribution’s systems (www.debian.org/security/2003/dsa-403).

Attackers exploited the previously undisclosed vulnerability in the Linux operating system’s kernel code, the do_brk() bug (www.kb.cert.org/vuls/id/301156), which required careful manipulation of the kernel memory to prevent system outage. This allowed them to obtain elevated access privileges on several systems that maintained and delivered the Debian distribution of the Linux open-source operating system. The systems were allegedly compromised through a remotely exploitable vulnerability in the rsync server program (www.kb.cert.org/vuls/id/325603).

By combining the two vulnerabilities, attackers could gain full control of the Linux distribution site, allowing them to insert malicious code in the software that several million Linux operating system users downloaded. Clearly, this great effort attackers put into discovering and exploiting these bugs would nonetheless provide a highly valued asset: The launching point for an effective attack against thousands or millions of computers with minimal additional effort. Fortunately, the attack was detected and the as yet undistributed software was verified clean before actual downloads occurred.

The addition of the attackers’ choice of malicious code in the software development and distribution processes is described and outlined as a serious attack trend in the “Poisoning the Software Supply Chain” article (May/June 2003, pp. 70–73).

The less-effort principle might also apply to attack vectors and vulnerability exploitation and to the use of compromised systems after successful attacks. The level of sophistication in worms seen in 2003 and the installation of backdoors and tools with elaborate communication protocols and auto update capabilities indicate that attackers are also trying to optimize the management of large amounts of newly acquired assets.

A blended future

In 2003, we witnessed the emergence of an attack type that combines exploitation of server and workstation vulnerabilities with the characteristics of virus and Trojan horses. By using more efficient attack vectors and, therefore, minimizing the human effort required to deliver attacks and use the compromised systems, the risks related to newly discovered vulnerabilities moved up in the risk measurement scale. The threats posed by attacks combining these variables are generally referred to as blended or hybrid threats, and are no longer fiction for information security practitioners.

The quantity and quality of the new vulnerabilities that will become evident in 2004, combined with the fact that blended threats are a real-world concern and not security research futurology, outline a complex scenario for information security practitioners. The emergence of blended threats as a new trend requires a more comprehensive and realistic view of the current best practices for risk assessment and protection of information assets.

As attacks become more complex, effective risk mitigation measures will require us to use our understanding of attackers’ goals, motivations, and resources to devise blended countermeasures that combine the basic concepts of risk assessment, software protection, security auditing, defense-in-depth, and proactive and reactive security effectively.

Iván Arce is chief technology officer and cofounder of Core Security Technologies, an information security company based in Boston. Previously, he worked as vice president of research and development for a computer telephony integration company and as an information security consultant and a software developer for various government agencies and financial and telecommunications companies. Contact him at ivan.arce@coresecurity.com.