The Sad And Increasingly Deplorable State Of Internet Security

David Piscitello and Stephen Kent

David Piscitello, president of Core Competence, Inc., is an internationally recognized expert in security technology and founder of the Internet Security Conference. He has chaired the Networld+ Interop Conference Program committee since 1996.
Dr. Stephen Kent is chief scientist, Internet Security at BBN Technologies. He has been involved with network security R&D for more than 20 years.

Feeling insecure about security? This article won't make you feel better, but it offers some concrete steps the industry needs to take.

At a time when we claim to be more focused on security than ever before, when the media routinely conjures images of forensic experts pursuing cyber-criminals in a dazzling game of network cat-and-mouse, and when we are readying the global IP infrastructure to carry all things voice, video and data, the title of this article no doubt constitutes a disturbing claim. But the sad reality is that overall, Internet security really is in horrible shape, arguably worse than ever before.

Internet security is, well, lame, and the situation may get worse before it gets better, if indeed, improving security is even achievable.

How Bad Is It?

The Computer Emergency Response Team Coordination Center (CERT/CC) reports and responds to computer and Internet security incidents, which they identify as any act that violates an explicit or implied security policy. Specific activities reported as security incidents include:

- Attempts to gain unauthorized access to a computer system or data contained therein.
- Unwanted disruption or denial of service.
- Unauthorized use of a computer system for the processing or storage of data, or hosting of an application (e.g., an underground chat room, or illegal software distribution, a.k.a., WAREZ site).
- Unauthorized modifications to computer hardware, firmware or software.
- Unauthorized installation and execution of software on a computer (e.g., a virus).

Having maintained records of incidents since 1988, CERT/CC provides excellent data to corroborate our claim. Figure 1 illustrates security incidents reported from 1993 to 2002, the World Wide Web era.

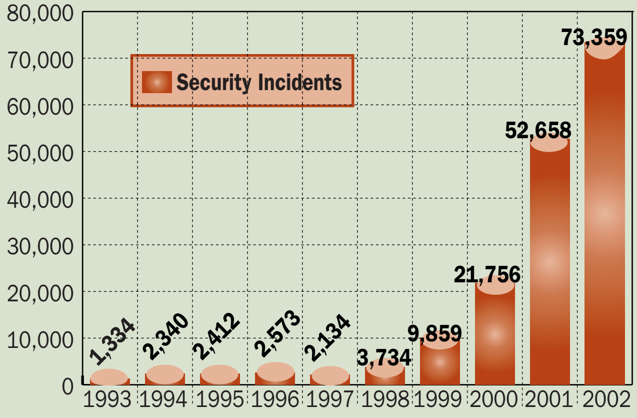
A total of 173,728 incidents were reported between 1998 and 3Q02. Extrapolating incidents from three quarters of data, we can project that nearly 100,000 incidents will have been reported in 2002, representing nearly 50 percent of incidents reported over the entire 15-year period. This data shows the security incident rate is doubling annually.

The 2002 CSI/FBI Computer Crime and Security Survey, conducted by the Computer Security Institute and San Francisco FBI Computer Intrusion Squad, attempts to gauge the scope and scale of cyber-crime in the U.S, and is acknowledged as one of the most thorough and encompassing studies. According to the survey, "the threat from computer crime and other information security breaches continues unabated and…the financial toll is mounting." Some disturbing statistics:

- 90 percent of the respondents (mostly large organizations and corporations) reported at least one security breach.
- 85 percent detected Internet viruses.
- 80 percent attributed a financial loss to a security breach.
- 75 percent cited an Internet connection as a frequent point of attack.

The financial toll is indeed mounting. One in five organizations experienced losses due to theft

FIGURE 1 Incidents Reported To CERT (as of 3Q02)



Source: FBI/CSI

The "patch and pray" approach is doomed to fail

of proprietary information in 2002, roughly consistent with prior-year percentages. However, the average losses reported by the respondents were more than $6.5 million, *a nearly seven-fold increase from 1997*. Better reporting practices and methods for ascribing value to sensitive information account for some of the increase, but the report soberly suggests that attackers are growing more sophisticated and motivated, and consequently are seeking out information of value.

The statistics also indicate that advanced security technology, as currently deployed, has not proved effective. The majority of respondents in this study are organizations that employ state-of-the-art security technologies—advanced forms of authentication, including digital IDs, biometrics and encrypted logins. Reporting organizations use Internet firewalls (89 percent), intrusion detection systems or IDSs (60 percent), antivirus software (90 percent) and considerable physical security measures (90 percent).

These statistics and many more available from credible sources illustrate that despite widely heralded advances in security technology, and heightened awareness that Internet security is critically important, in practice, security is lax—and worsening. The frequency of incidents is increasing at an alarming rate. The cost per incident is rising.

Security Will Get Worse Before It Gets Better

Today, the vast majority of the security problems that plague us arise from three sources: insecure architectures, poor software engineering and sloppy management by users and systems administrators. Only by analyzing and committing ourselves to mitigate these root causes will we ever improve Internet security.

■ **Insecure Architectures:** The first of the Internet's most disturbing security problems arises due to various factors: time-to-market priorities, inadequate security understanding by product architects and the (perceived) conflict between ease of use and security. Several dreadful outcomes are abundantly evident in Internet-related products on the market today:

1. Design criteria always favor ease of installation and use over secure operation. To make computers and networks easy to install, popular operating systems and network hardware are designed to "plug and play" and facilitate open networking. The out-of-the-box or default configuration of the majority of equipment and software allows access to administrative facilities with weak or no authentication. Sensitive file systems are not protected with access controls. Networked file and printer sharing is enabled by default, and many system resources can be accessed without authentication. Guest accounts with no or well-known passwords are enabled. Application services are enabled and run automatically, again using default configurations that allow access (intended or not) to all hosted information by any user. There's no better evidence that security is misunderstood and poorly practiced than the ubiquitous "Save Password?" prompt by client applications.

2. Circumventing and overlooking security measures, intentionally or out of ignorance, are common. Many non-technical users are entirely unaware that their systems are easily exploited and their sensitive data are left vulnerable when they turn on their computers or connect to the Internet or enterprise networks using wireless LANs, cable and even dial-up modems. Even competent system administrators find it difficult to identify and rectify every configuration parameter that permits a potentially dangerous action or enables an unprotected service on a server.

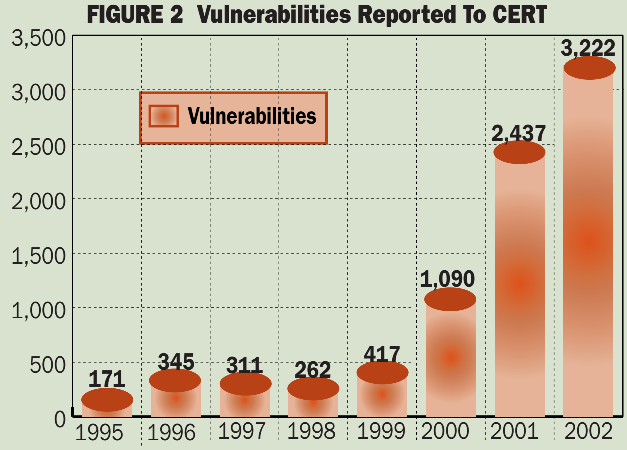
This problem is compounded by the too-frequent introduction of new versions of software and installation of untested and unauthorized software. Organizations are often at the mercy of so-called power users, who operate Web, chat and peer-to-peer applications on client computers with no appreciation of the vulnerabilities they introduce. When you combine these occurrences with outcome number one described above, it's easy to anticipate the train wreck.

3. To satisfy our desire to make computing and networking easy and ubiquitous, we have taken the concept of "common platform" to an extreme. When the Morris worm struck in 1988, the set of computer operating systems connected to the Internet was far more diverse than it is now, and in part that helped limit the rate and extent to which that first large-scale worm affected the Internet population. Today, Microsoft provides the dominant set of OSs and applications for the systems connected to the Internet, certainly in terms of client machines. Sun's Solaris and Red Hat Linux account for much of the remaining server system OSs. This high degree of homogeneity in OSs and applications provides an environment ripe for both exploitation and propagation of malicious code among Internet-connected computers, further contributing to our security problems.

We suffer from operating systems that offer too little protection, and applications that, in the name of flexibility, offer too many opportunities for manipulation by attackers. OS and application security is not getting better. Each new version of Windows, Web server, database and other mission-critical applications adds more features and attendant vulnerabilities. An endless stream of patches is issued to fix problems that, all too often, are uncovered not by the vendor but by other parties. This "patch-and-pray" approach to OS and application security is doomed, yet we see little or no substantial efforts by vendors to improve on basic software engineering procedures.

■ **Poor Software Engineering:** This seems to be a very long-term problem: We do not know how to write good software! Haste to market is certainly responsible for some portion of the bugs in software that expose systems and applications to

FIGURE 2 Vulnerabilities Reported To CERT



Source: FBI/CSI

unauthorized access and misuse. But our industry in general does not pay adequate attention to software quality assurance. The result is that security vulnerabilities—flaws in software that expose a computer system or application to attack or exploitation—are identified too frequently for nearly every commercial software product (Figure 2).

Secure code review is a little-practiced art form within software companies, and too few companies permit third-party review of their source code for fear of revealing what they believe to be novel or inventive. Worries over intellectual property and copyrights take precedence over any concern that users will be left vulnerable to attack.

Vendors are not solely responsible for software woes. Many user organizations employ staff to write and customize application software, develop Web content and computer gateway interfaces (CGIs), the programs and scripts used to create dynamic and interactive content. This code is among the most exploited today.

■ **Sloppy Management:** This thoroughly pervasive problem is at least partly a side effect of the human condition: We're not very disciplined. The reasons Internet security is likely to worsen before it improves have less to do with advances in security technology and *everything* to do with how we view and practice security.

Observe passengers at airport checkpoints as they are asked to unpack carry-on bags, discard nail clippers and remove clothing and shoes. Security measures in the physical world are regarded as progress-impeding, time-consuming inconveniences. As the events of 9/11 fade into memory, security will be relaxed or become lax as economically-strapped airlines and impatient passengers seek to eliminate the delays and inconveniences. Imagine the uproar if we were to ask for identification at tunnel, bridge and turnpike toll booths. Societies at large, and democratic societies in particular, interpret intrusive security as rights-threatening and entitlement-stripping.

While there are many ways our undisciplined and lax behaviors weaken Internet security, the most egregious offender and best example is our truly lame authentication practices.

Persistently Lame Authentication

Authentication is fundamental and necessary for security. Basically, if you can't confirm the identity of an individual or computer based on credentials he, she or it presents, you shouldn't exchange information. Yet the majority of authentication practiced on the Internet today is based on a user identity and associated password, which is hopelessly inadequate.

Authentication practices are lame, lame, LAME!

Passwords, in most of the ways they are used, are a fundamentally flawed, individual authentication mechanism. Most users understand the vulnerability posed by static passwords transmitted over insecure communication channels. Yet static passwords are usually tolerated, and even organizations that prohibit static passwords do not enforce meaningful password composition rules. Many enterprise and ISP-provided email systems fail to support encryption (e.g., via SSL or Kerberos) of passwords used to authenticate a user to email (e.g., POP) servers. Even when encrypted for transmission, a static password is subject to guessing attacks, and the common countermeasure—locking down an account after some number of consecutive wrong entries—provides a ready-made denial-of-service attack opportunity.

The use of public key technology for user authentication provides a much more secure alternative to passwords, but it is deployed only in limited contexts. For example, Secure Sockets Layer (SSL) offers the option of employing client certificates to authenticate users, but this option is rarely employed because of the perceived difficulty of using certificates.

In fact, it is easy to issue certificates to Web users who already have an account with a website. And issuing a client certificate makes future website visits easier for the client. How? SSL provides a facility by which a server can communicate to a client the ability to authenticate via a certificate, and the server can specify a list of approved Certificate Authorities. In theory, each website could operate its own Certificate Authority exclusively for authenticating that site's users. As a result, users wouldn't have to remember and associate identities and passwords for multiple sites, and the lax practice of reusing the same password for all sites, or storing names and passwords in poorly protected browser files, could be eliminated.

Few sites offer this option, however, in part because they may have been scared away from the enabling technologies: Public key technology in general, and PKI in particular. For too many users and administrators, PKI is a four-letter word (which also suggests a spelling problem). PKI products have often been hard to use, and PKI services have been expensive.

But these problems are not intrinsic to PKI; they are side effects of how vendors have chosen to design products and services. Moreover, many organizations suffer a misconception that issuing certificates for user authentication implies some sort of legal liability. If certificates are configured for use in a very limited context, there should be no more liability than for passwords.

Other strong methods of authentication are likewise neglected. For example, the commercial sector makes very limited use of hardware-based

How Did We Fall So Far From Grace?

The origins of the Internet are often—and inaccurately—depicted as laying the groundwork for our current security problems. The U.S. Department of Defense was the primary funding agent for the Internet in its early days, starting with the ARPANET (1969). The purpose of ARPANET development was to foster collaborative R&D by facilitating the sharing of information across geographically disparate sites, and security was *always* a concern. By the mid-'70s, the U.S. Navy was using the ARPANET to transfer classified data, employing BBN's Private Line Interface (PLI) devices, forerunners of today's VPN technology, to provide cryptographic protection.

While PLIs were being developed, Vint Cerf and Bob Kahn designed the TCP/IP protocol suite that still defines the Internet. As TCP/IP was emerging, cryptographic network security was developed that operated in this new protocol domain. The first of these devices, the Black-Crypto-Red (BCR), also developed by BBN under ARPA funding, made use of a Key Distribution Center (KDC) of the type that was later popularized by the MIT Kerberos project.

Thus, contrary to popular belief, security technologies for sensitive data were developed hand in hand with computer operating systems and the Internet protocols. So, if DoD was funding security R&D and developing prototype security technologies for network and computer operating system security as the Internet evolved, what went wrong?

Network Security: A Matter of Context

First, the threat model adopted by the DoD in the Internet context was very different from the way commercial and most academic users view security. The DoD started with the assumption that adversaries were able to intercept essentially all communications and thus encryption was a necessary starting point for security. There also was a concern that an adversary might introduce malicious software (e.g., a Trojan Horse) into a computer containing sensitive information, and use that software to exfiltrate the sensitive data. Thus, communications security measures had to be implemented in high-assurance devices, e.g., separate, inline network security hardware, not software executing in user computers, to ensure they were not tampered with or bypassed.

Thus, the DoD plan for secure use of Internet technology was based on the use of special purpose, highly reliable hardware to provide data confidentiality, integrity, authentication and access control for inter-computer communication. The plan addressed the problem of secure communication among members of a community operating at a common *sensitivity* level, e.g., Secret or Top Secret: a community of "good guys" was insulated from the "bad guys" who might gain access to the communication media.

This style of Internet security is very robust, and these measures are indeed applicable to the commercial sector. However, these mechanisms do not address all the subtle problems that arise in the public Internet. Here, many autonomous communities exist. Each has its own (usually unarticulated) definition of sensitivity level(s) and requirements. Moreover, these communities frequently collaborate openly, without regard to security whatsoever. Here, the line between the good guys and the bad is not easily defined, making security considerably more difficult to enforce.

Computer OS Security And The Advent Of PCs

DoD funded the development of secure operating systems in the '70s and '80s, and created the National Computer Security Center (NCSC) to evaluate OS security. This work was motivated in large part by a desire to allow a single computer to process information at different sensitivity levels using a commercial OS. Unfortunately, this effort coincided with the introduction of personal computers (PCs). PCs were initially viewed as single-user systems, not connected to networks, and thus their operating systems offered much less security than the multiuser timesharing systems that were the focus of the NCSC efforts. As a result, PC OSs were significantly less secure than their mainframe (and Unix workstation) counterparts, a legacy that persists today.

The DoD security model also did not address many problems posed today by systems handling unclassified data connected to the public Internet, because the public Internet did not exist when the model was conceived. In that sense, the DoD unclassified user community faces all the same problems as the rest of the Internet user community, and it arguably is not much better at dealing with these problems. Finally, the DoD did not encourage open publication of the results of much of its sponsored work in this area, and thus the larger Internet community was not aware of much of what had been done.

Featurism, At Internet Speeds

As TCP/IP evolved in the '80s and '90s to accommodate many new features and applications, the not-so-small matter of how new protocols and changes to existing protocols would affect security was largely ignored. The IETF did initiate several major security protocol efforts in the early-mid '90s—IP and Transport Layer Security (IPSec and TLS), and Secure Multipurpose Internet Mail Extensions (S/MIME)—but by this time, the rapid adoption of Internet technology inhibited security standards definition, approval and deployment. An undesirable precedent emerged that has profoundly and negatively influenced Internet security: Deploy it now, secure it later□

cryptography for per-computer network security or user authentication. Smart cards and similar hardware tokens for user authentication hold promise, but not as a standalone mechanism; they should be viewed as an additional layer of security on top of crypto-based authentication. Unfortunately, the costs associated with issuance and maintenance of hardware tokens, plus interface problems, suggest that these technologies are not appropriate for universal use.

But if we start down the road to using public key, or preferably PKI, for user authentication in software, we will have already laid the foundation for a transition to hardware-based tokens for improved security. Protocols for certificate-based authentication are the same whether the private key is protected by software or hardware, which means that one can selectively employ hardware for better security with minimal changes to the fundamental authentication system.

Masking, Not Mitigating The Shortcomings

The security industry emphasizes products that focus on detecting attacks (firewalls and IDSs) and, more recently, products that try to respond to attacks, i.e., to block the attack before it's completed or to limit the damage done by a successful attack. These are generally unsatisfying strategies. Many activities may look like an attack, or more likely, like an attack precursor, and thus result in false alarms.

The result is ironic: First we develop products to detect potentially adverse behavior, then when the products yield too many false alarms, we develop more products to help sort through the output to help reduce the false alarm rate. Could any but the software industry get away with selling products that create problems, selling more products to solve the problems they created, then selling *more* products to solve the second generation of problems?

Moreover, despite all these efforts, very capable attacks can usually evade IDS software and manage to gain unauthorized access anyway, by operating "below the radar," e.g., using "low and slow" attacks that look like noise relative to all the other probes and normal traffic. What we really should do is refocus vendors on eliminating design and implementation vulnerabilities, rather than escalating efforts to detect attacks.

Is There Hope?

It is conceivable that Internet security will improve, but there are many obstacles to overcome, and both users and vendors must "get religion" before substantial progress is likely. Vendors must focus more on reliable, secure designs and implementations, and less on time to market.

Vendors and users alike would benefit from a "feature moratorium." Instead of adding still more features to products that are already feature-heavy, invest a commensurate effort to make software more secure and reliable. We speculate that software engineering would actually improve and ultimately, better products would be implemented in reasonable time frames at reasonable costs.

What could drive software manufacturers to alter their existing course in this manner? Perhaps it's time for organizations with influence (read: large purchasing power) to refuse the standard EULA (End User License Agreement), which protects the vendor but does little for anyone else, in favor of a software reliability agreement. The latter can be negotiated in the same manner as a service level agreement from a service provider.

Users, system and network administrators must become more disciplined, better able to account for the hardware and software and associated configuration data that characterizes computing environments. Archival practices must be improved; in particular, administrators and users alike must learn to appreciate the importance of saving working configuration data along with their other mission-critical information. ISPs must do a better job of configuration management for their components, and must offer attack tracing and traffic filtering capabilities to assist subscribers in response to distributed denial-of-service (DDoS) and other attacks where subscriber resources cannot suffice.

What could cause this to happen? One possibility is that the insurance companies will begin to reward vendors and service providers who take significant steps to reduce the vulnerabilities of their products and services, and that the legal system will begin to impose liability on those who create these problems by their negligence in product design.

Given the litigious nature of our society, of these two, the latter is most likely. Organizations already have concluded that pursuing attackers in criminal courts is only a partial remedy, and one that offers little financial compensation. Simply put, attackers don't have deep pockets. Savvy attorneys will seek relief in civil courts from such security incidents as denial of service and worm attacks repeatedly emanating from a conclusively identifiable source. If they prove the source is obstructing their business, they may succeed in having the court order the ISP serving the site from which the DoS attacks emanated to terminate access service. Eventually, organizations will file civil suits seeking compensation from companies that have failed to meet accepted best security practices (today) and industry or federal security standards (tomorrow)□

Technology isn't enough. We need changes in behavior

Companies Mentioned In This Article

Microsoft (www.Microsoft.com)

Red Hat (www.redhat.com)

Sun Microsystems (www.sun.com)