

## **Information Security: The Inevitability Of Failure The Flawed Assumption of Security in Modern Computing Environments**

In the last installment in the “Information Security” series, [The Curious Paradox](#), we spoke fondly of the “Short Sermon.” In summary, the sermon laid the foundation for an ever-vigilant approach to your company’s network security. Yes folks; to become complacent is to become a target. Staying current on the new and different threats facing your network, both internal and external, is an absolute necessity. The basic idea behind the paradox was that, despite the security measures in place, the majority of companies we referred to still sustained substantial damage to their networks. To continue our conversation from the last article we need to revisit the main question posed: why? Why are security measures, implemented by companies, ineffective? In the last article we gave you the short answer, meant to get you ready to start taking a continuously active role in your own information security. Though the need for this “vigilance,” when approaching security, is an immutable truth, the reasons for failed efforts, such as the ones described in previous writings, can get much more complex.

The idea behind this Information Security segment spawned from a 1998 report published by the National Security Agency (NSA), appropriately titled: “The Inevitability Of Failure: The Flawed Assumption of Security in Modern Computing Environments.” Though the information in this report was assumedly out-of-date we thought it would still be of some interest. As we read, we realized that, though the article was dated, much of the information it brought to light still rings true. As such we will use this information as our “jumping off point,” making it current as the article progresses.

Four years ago, as the need for greater interconnectivity in and between computer systems grew, so did public’s awareness of the need for proper information security. Then, as well as now, there were a multitude of network security products on the market. The NSA report made a strong argument (and concluded) that even with many of these measures in place “the threats posed by the modern computing environment cannot be properly addressed without certain security features in operating systems.” The rationale behind this conclusion arose from several scenarios, of which we will provide a basic overview. Foremost, it is important to note these necessary security features.

Per the NSA, the two main features missing from mainstream operating systems (but present in a properly secure system) are *mandatory security* (policies/mechanisms) and *trusted path* (mechanisms). To make this point, general examples of access control and cryptography are utilized. Here, mandatory security is used to mean, policies that are always applied, even for “super-users” and administrators and mechanisms in the operating system to support and enforce these policies. In the example of access control a “mandatory security mechanism in the operating system may be used to ensure that all accesses to the protected objects are mediated by the ‘enforcer’ component (of the application space access control mechanism).” Trusted path refers to the communication between a single individual and particular software, specifically, “a trusted path is a mechanism by which a user may directly interact with trusted software, which can only be activated by either the user or the trusted software and may not be imitated by other software.” In the example of cryptography,

“A trusted path mechanism obviates the need for a separate physical interface for activation by permitting the cryptographic token to identify its callers and enforce fine-grained controls over the use of services, algorithms, sessions, and keys.”

Albeit a little dry to read, this information is extremely important when considering the necessity of a secure OS (operating system). From the NSA’s perspective, these features are a priority if one is to consider an operating system to be secure. It is important to note that these are not the only essential

features in a secure OS. These were the features that the NSA specifically saw as imperative back in 1998.

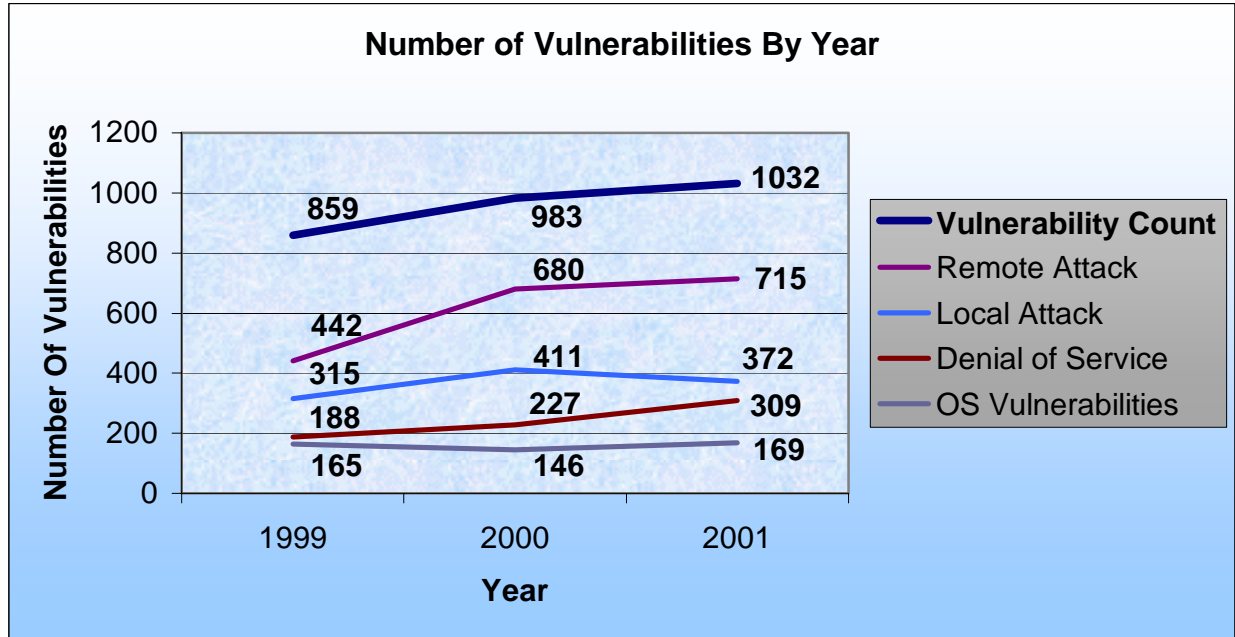
In addition to the general examples in the NSA report, there are also some more concrete scenarios listed. These scenarios revolve around vulnerabilities in Java and Java Virtual Machine (JVM); Kerberos; and IPSEC and SSL. In each of these cases the NSA explains how features as simple as mandatory security and trusted path mechanisms in the operating system can provide a viable defense against any or all of these vulnerabilities. Rather than bore you with specifics on the varied vulnerabilities of these functions in 1998, it is time to bring this article back to the future, so to speak.

When evaluating would-be dated information, such as this, we are tempted to say: “Though these issues may have been of great concern four years ago there is no way that they could still be as problematic as they once were.” And generally speaking this is true. As we all know, four years in the technology industry is an eternity. Since the NSA report was published there been a myriad of technological advancements, not to mention, the cornucopia of various patches, upgrades, and “fixes” of many sorts. It is easy to assume that these vulnerabilities are now as obsolete as much of the technology we were using back in '98. Ah, if it were only that easy. And if you have read any of our previous articles you know it never is.

Through a bit of research we were able to establish that many of the issues listed in the NSA report are still problems today. In taking just a few moments to search one vulnerability database, provided by the National Institute of Standards and Technology, we found lists of vulnerabilities that apply to the same functions mentioned in the report. Furthermore, many of these said vulnerabilities were not dissimilar to the ones specifically listed by the NSA. To illustrate this point refer to the following table. It lists each function mention by the NSA accompanied by a recent corresponding vulnerability. This list is not all-inclusive. It is merely meant to help us demonstrate our point.

Specified Function	Vulnerability Example
Access Control	slapd in OpenLDAP 2.0 through 2.0.19 allows local users, and anonymous users before 2.0.8, to conduct a "replace" action on access controls without any values, which causes OpenLDAP to delete non-mandatory attributes that would otherwise be protected by ACLs.
Cryptography	Common Cryptographic Architecture (CCA) in IBM 4758 allows an attacker with physical access to the system and Combine_Key_Parts permissions, to steal DES and 3DES keys by using a brute force attack to create a 3DES exporter key.
Java	A cross-site scripting vulnerability in Apache Tomcat 3.2.1 allows a malicious webmaster to embed Javascript in a request for a .JSP file, which causes the Javascript to be inserted into an error message.
Kerberos	Memory leak in Microsoft 2000 domain controller allows remote attackers to cause a denial of service by repeatedly connecting to the Kerberos service and then disconnecting without sending any data.
SSL	Running Windows 2000 LDAP Server over SSL, a function does not properly check the permissions of a user request when the directory principal is a domain user and the data attribute is the domain password, which allows local users to modify the login password of other users.
IPSEC	Buffer overflow in IPSEC authentication mechanism for OpenBSD 2.8 and earlier allows remote attackers to cause a denial of service and possibly execute arbitrary commands via a malformed Authentication header (AH) IPv4 option.

We could write numerous articles entirely on the current existing vulnerabilities in these categories but that is not the goal of this writing. The goal here is to point out the ever-increasing difficulty of properly securing your information assets without a secure OS. The basic point here is that despite standard measures taken; network security remains vulnerable unless one ALSO employs system security at the OS level. To help drive this point home, refer to the graph below. It helps stress the necessity for a secure operating system by illustrating the growing number of vulnerabilities by year (starting in 1999, where the NSA left off).



- National Institute Of Standards And Technology (NIST) Analyzed Vulnerabilities

To really put emphasis on this point the following table lists both the number of vulnerabilities, shown above, as well as the percentage each category makes up. It is vital to note that these categories are not mutually exclusive. These percentages are based on the varied characteristics of each vulnerability. Since said characteristics may fall into more than one category, these percentages total more than 100% in each section.

Vulnerability Type	YEAR			
	1999	2000	2001	2002
Remote Attack	442 (51%)	680 (69%)	715 (69%)	17 (77%)
Local Attack	315 (37%)	411 (42%)	372 (36%)	7 (32%)
Denial of Service	188 (22%)	277 (28%)	309 (30%)	5 (23%)
OS Vulnerabilities	165 (19%)	146 (15%)	169 (16%)	6 (27%)
<b>Vulnerability Count</b>	<b>859</b>	<b>983</b>	<b>1032</b>	<b>22</b>

- National Institute Of Standards And Technology (NIST) Analyzed Vulnerabilities

This table shows the distribution of various vulnerability characteristics. The raw number in each cell is the number of vulnerabilities that meet that particular characteristic for that year. The percentage to the right of each raw number is the percentage of vulnerabilities having that particular characteristic for that year. These categories are not mutually exclusive.

Overall these numbers are steadily on the rise. In fact, the best trend that can be interpreted, given the information, is that these counts will possibly level off (not likely). OS vulnerability characteristics, in and of themselves, already comprise 27% of the assessed weaknesses in 2002 (and its only March).

Though this information helps us see the problems information security is faced with, it does little to convince us of the necessity of a secure operating system.

To help us to that conclusion we consulted the “Top Security Threats Affecting All Systems” compiled by the SANS Institute. We also gathered information on how many of these threats could be counteracted by implementing secure operating system technology. Most of the threats listed by SANS dealt with either vulnerabilities at the operating system level or weaknesses in company security policies. In either case, a strong argument can be made for a secure OS. There are a total of 20 “threats” listed by SANS but since we are simply illustrating a point we have chosen to highlight only the top few.

Security Threats	Security Technologies & Approaches					
	Firewall	Intrusion Detection	ID & Authentication	Access Control	Encryption	O/S Level Security
Default O/S Installations						✓
Weak or No Passwords			✓	✓		✓
Open Ports	✓			✓		✓
Incorrect/Incomplete IP Filtering	✓					✓
CGI Script Vulnerabilities		✓ <sup>+</sup>		✓ <sup>*</sup>		✓
Buffer Overflows		✓ <sup>+</sup>				✓

**FIGURE 1: Security Threats and Approaches to Threat Prevention**  
(Threats Taken from SANS Top Security Threats Affecting All Systems)

- + - If attack is previously known and updated in signature registry.
- \* - If vulnerability is previously known.

The long and the short of it is that network/information security is getting more and more complicated. The utilization of a secure operating system has consistently shown itself to be one of the only ways to provide adequate security. In the next *Information Security* segment, we will discuss a class of software categorized as secure application environment (SAE) technology. Such software was spawned from research much like what the NSA provided back in 1998. SAE technology provides security at the operating system level by locking down each individual application. So, join us next time, when we bring the secure operating system home with SAEs.