SKYVIEW
PARTNERS, INC

Virus Got You Down?

by Carol Woodbury and Patrick Botz

Or maybe we should have asked, “Does a virus have your server down?” Perhaps it’s the latest worm, Trojan horse, buffer overflow or denial of service attack that’s got you or one of your servers down. While one of these bugs may be affecting one or more of your servers in your enterprise, it is highly unlikely that the server affected is an iSeries server running OS/400. OS/400 may be running your core business applications or it may be hosting your web site or running Domino. Whatever its function within your enterprise OS/400 has remained unaffected by the recent virus and worm attacks. Why is that?

Viruses and other ailments spread by infecting a host that is vulnerable. Let’s take a look at how OS/400 and the applications running on it can remain unscathed by the viruses and worms that are so prevalent today. We first define each “ailment” and then describe the defenses and protection mechanisms provided by OS/400 to ward off the attack.

Definitions

Virus \'vI-r&s\ n. a computer program usually hidden within another seemingly innocuous program that produces copies of itself and inserts them into other programs and that usually performs a malicious action (as destroying data)

A virus is typically spread by duping the unsuspecting users into installing and running an infected program on their system. Often e-mail attachments are used to facilitate the duping. Some operating systems make it very easy for an external attacker to cause programs to execute on your system. Therefore, if the virus writers can get you to detach the attachment or somehow copy the file to your system, they can use any number of mechanisms to cause that program to run.

Worm \'w&rm\ n. a self-replicating virus that does not alter files but resides in active memory and duplicates itself. Worms use parts of an operating system that are automatic and usually invisible to the user. It is common for worms to be noticed only when their uncontrolled replication consumes system resources, slowing or halting other tasks.

A worm is typically spread by gaining access to your system externally through a vulnerable interface and installing itself in memory. The worm then attempts to use your system to access the same vulnerable interface on other systems.

Trojan horse \'trO-j&n 'hors\ n. a seemingly useful computer program that contains concealed instructions which when activated perform an illicit or malicious action (as destroying data files)

A Trojan horse attack is often used to help spread viruses. Some use this term interchangeably with virus.

Buffer overflow \'b&-f&r "O-v&r-'flO\ n. the condition that occurs when a program or process tries to store more data in a buffer (temporary data storage area) than it was intended to hold

A buffer overflow is used by attackers to gain “root” control of your system. Root control means the program has ultimate authority to do anything. For those familiar with OS/400, root control is equivalent to being signed on as QSECOFR. The attack relies on a common programming flaw found in many programs.

This attack is particularly onerous because an attacker can randomly choose any vulnerable machine on a network and attempt to gain access to the system. Attackers don’t have to spend a lot of time and energy trying to get you to install Trojan horses or dupe you into changing the configuration of an infected machine. They just look for addresses on the Internet and try the attack. If it works - great! If not, they try the next address.

Denial of Service Attack `\di-'nI(-&)l &v 's&r-v&s &-'tak\ n.` an incident in which a user or organization is deprived of the services of a resource they would normally expect to have.

A denial of service attack can take many forms but the end objective is the same: to prevent you or your customers from using your IT resources to conduct business. This is a particularly difficult attack to defend against.

In the past, each of the “ailments” or attacks described above was often thought of individually, however, attackers have become more sophisticated. They now use combinations of all of these attacks to wreak their havoc. The term virus as is commonly used by the general public means any malicious attack that uses one or more of these techniques. For simplicity, we’ll use the term virus in this way as well.

What Makes OS/400 Different?
Companies are spending more and more on virus prevention and yet viruses seem to permeate the corporate network with alarming frequency. Viruses can be very destructive, erasing data or gleaning access to private data stored on the computer. What is the risk of OS/400 being infected with a virus such as W32.Sobig.*? Highly unlikely. Why? The answer lies in the OS/400 architecture, integrated security management, and the overall development process.

Defending a system against viruses involves the prevention, detection, and avoidance of attacks. It is necessary not only to defend the front door, but also to protect the back door. OS/400 protects your backside, while you exploit the rich, low-cost, easy-to-use integrated security management functions to help detect and avoid attacks that may be trying to waltz in the front door.

OS/400 Architecture: Defending the Back Door
From the definitions above, you can see that viruses are just as likely to try to sneak in the back door to your system as they are to use the front door. OS/400 helps defend your back door so you can concentrate on making sure only well-known and trusted entities are allowed through the front door of your system.

PC viruses cannot execute on OS/400 for much the same reason that VHS tapes could not be viewed on tape players using the Beta format – the architectures are different. Does this mean OS/400 is “safe” simply because of its different architecture? Not necessarily. What if the virus programmers turn their attention to a new and more daunting challenge - creating a virus specifically aimed at OS/400? Would OS/400 fare any better against a targeted attack? The answer is unequivocally yes.

Let's take a look at the OS/400 architectural features that provide a line of defense against the most common techniques used for writing viruses and launching attacks. OS/400 is an object-based operating system. While most other systems have generic files, OS/400 has distinct object types -- objects are strongly typed. This means that objects that wish to run instructions must claim to be an "executable" type such as a program or service program. A program disguised as a non-executable object type, such as .gif file, cannot be independently executed on OS/400. (Running hidden instructions in non-executable object types is a very popular virus technique in other operating system architectures.)

OS/400 is shipped with a "trusted translator." It is one of the key architectural features in OS/400 that provides protection against viruses. The translator takes intermediate code produced as a result of a compile operation and changes it into hardware instructions. The trusted translator only produces "valid" OS/400 programs. By definition, a valid program does not contain viruses.

Valid programs cannot "manufacture" pointers. (A pointer is used inside a computer program to "point at" data that the program is currently reading or updating.) One common technique that virus writers use is to turn data into a pointer and then use the manufactured pointer to read or destroy data. Imagine a whiteboard eraser wiping your board – that is, your hard drive - clean. You get the picture. Fortunately, data cannot become pointers in valid OS/400 programs.

In OS/400, the instruction stack is separate from the data stack. While it is possible for an OS/400 program to be vulnerable to buffer overflow attacks, the attacker cannot turn this into an opportunity to gain control of the system. Because data and instructions are stored separately in OS/400, a buffer overflow, at worst, simply causes a failure in the targeted program. (The program stops running and may send a nasty error message.) However, on OS/400 this type of attack will not result in the attacker gaining administrative control of your system as is possible on other systems. OS/400 architecture prevents this from happening.

These architectural features provide the solid line of defense that make OS/400 virus resistant. These traits make it extremely difficult for an OS/400-targeted virus to be spread from system-to-system via back door exploits.

Integrated Security Management: Protecting the Front Door

You still have to worry about viruses coming in the front door. Who you let in or what you let on to your system is ultimately your decision. To make it easier for you to protect your front door, OS/400 provides you with a wide range of easy-to-use, integrated security management functions that you can use to either avoid or detect viruses trying to get onto your system.

To protect OS/400, IBM digitally signs all portions of the operating system. The command, Check object integrity (CHKOBJITG), computes the object's current digital signature and compares it to the one computed by IBM. This allows OS/400 to detect whether the operating system itself has been infected or modified in a way that could affect the integrity of the system. In addition, when programs claiming to be a part of the operating system are restored, the restore process automatically checks these signatures. Independent Software Vendors can also sign their applications with the OS/400 digital signature APIs providing their customers the same assurances that they get with OS/400.

Three global controls (system values) allow iSeries administrators to take complete control over what is restored onto the system.

- You can use the Verify Object Restore (QVFYOBJRST) system value to configure the system to examine the digital signature of an object before it is restored. You can require that all "signable" objects (such as programs) are indeed signed and that the signature is valid. If the object is not signed or the signature is not valid, the object will not be restored.
- You can use Force Conversion on Restore (QFRCCVNRST) system value to require that all executable objects be "re-translated" (i.e. re-generate the hardware instructions) by the system's trusted translator before they are restored. This assures that if an executable object being restored to your system did contain a virus, the executable object is clean after it is restored. If an executable object fails re-translation, it will not be restored to the system. The ability to re-generate hardware instructions and remove potential viruses from a program object is unique to OS/400.
- You can use the Allow Object Restore (QALWOBJRST) system value to control whether security-sensitive programs are restored to your system. You can use this value to allow security sensitive programs only from those you personally trust on the system. This helps prevent valid executables which are malicious from being restored on your system.

In addition, settings within the service tools allow you to prevent these and other security-related system values from being changed -- even by security administrators. With this much control, you have the tools required to help prevent or avoid harmful programs from slipping into your system unnoticed. You can also prevent your system from being unknowingly placed in a less-secure configuration.

For argument's sake, let's assume that a virus has somehow infiltrated OS/400. While we've demonstrated that this is an unlikely occurrence, what if the unthinkable happens? How do you limit the scope of the damage?

One of the best ways to protect your system against viruses running rampant is to implement a robust security policy. If you don't manage security on OS/400 using the plethora of built-in functions, you will be vulnerable to security breaches. Object level security is required to implement any robust security policy. We also strongly recommend limiting the number of security administrators on your system. For a virus to inflict large scale damage, it would have to have sufficient privileges to do so. Limiting the privileges a user has to only those required to perform the duties of their job will limit the scope of what a virus could do when launched under a normal user's authority.

Limiting the number of security administrators as well as requiring them to frequently change their passwords will limit the risk of having this level of privilege exploited.

Built on a Firm Foundation

Because of its openness and adoption of open standards, OS/400 can run applications that are designed to be cross-platform. Are these applications offered the same defenses and protection as more traditional OS/400 line-of-business applications? Yes and no. They absolutely benefit from the OS/400 features described above, but there are some vulnerabilities to be aware of.

Because of the popularity of Domino on iSeries, we're going to use that as our example. Many cross-platform applications (including Domino) are implemented using one of the file systems in OS/400's Integrated File System (IFS) rather than using OS/400's "traditional" library architecture. However, Domino server code is implemented and runs in the traditional OS/400 library system. This means that the features that defend the backside of OS/400 - strong object typing and separation of the data stack from the instruction stack – protect these applications as well.

User applications that run within Domino, on the other hand, use the "POSIX" file system. This gives these applications the ability to run on iSeries, on Intel platforms, on UNIX, and on zSeries with little or no change. The downside of this cross-platform flexibility is that user applications don't necessarily benefit from all lines of defense available to applications using the QSYS file system. User applications that run within Domino have the same built-in Domino integrity protection on OS/400 that they have on other platforms, but basically no more than other platforms.

However, using the features OS/400 provides to protect your system, you can reduce your risk to be below that of Domino applications running on other platforms. For example, Domino objects stored in the IFS are protected by OS/400 object-level security (in addition to the Domino access control lists). And Domino users and programs launched by a Domino application cannot perform any functions that the user does not have authority to perform.

Bottom line? OS/400 can help reduce your risk when running applications such as Domino that have been designed to run on multiple platforms.

Development Process

Security is not a new initiative for OS/400. Security considerations and standards have long been a part of the OS/400 development process and new security and integrity features are routinely provided in new releases. All new and changed OS/400 function goes through a rigorous design and review process intended to prevent the types of vulnerabilities commonly found on other systems. A quick glance at the number of security-related, (in IBM terms "integrity") PTFs versus the number of patches issued by other companies bears this out.

Ill or Just Making Others Sick?

We have shown you why PC viruses cannot be executed on OS/400. We have also shown you how OS/400 defends itself and allows you to help avoid your system being infected with OS/400-targeted viruses -- should one exist. However, can a virus that doesn't affect OS/400 be stored on an OS/400 system? In other words, can OS/400 carry viruses that infect non-OS/400 systems? The answer again is yes.

OS/400 is often used as a file server for non-OS/400 systems. Like any other file server, files being served to non-OS/400 systems can be infected with as well as house and propagate viruses. That's why we highly recommend that you scan your Domino database using a commercially available anti-virus (AV) product such as those provided by Trend Micro or Symantec. When using OS/400 as a file server, you scan your server for PC viruses using the AV product from Bytware.

Conclusion
Are we saying that OS/400 can never be attacked and has no vulnerabilities? No. It was designed and built by humans. Humans cannot anticipate the evolution of all aspects of technology nor anticipate the actions produced by the sinister minds of people with evil intent. However, compared against other operating systems, we believe OS/400 provides more prevention, more detection, and more avoidance technology than other systems. The OS/400 architecture, integrated security management features, and development process combine to provide you a low cost, easily managed, highly securable, virus resistant alternative to other commercially available systems today.

About the Authors

Carol Woodbury
Carol is President and co-founder of SkyView Partners, a firm specializing in security consulting and services. Prior to forming SkyView, Carol spent 16 years with IBM in Rochester, MN. She served for more than 10 years as the AS/400 Security Architect and Chief Engineering Manager of Security Technology for IBM's Enterprise Server Group and leader of the OS/400 Security Development Team. During this time Carol provided security architecture and design consultations with IBM Business Partners and large AS/400 customers. She is known worldwide as an author and speaker on security technology, specializing in AS/400 and iSeries security issues. Carol co-authored the popular book, Implementing AS/400 Security, from 29th Street Press as well as co-authored with Patrick Botz, Experts' Guide to OS/400 and i5/OS Security, also from 29th Street Press. Carol has 14 years of experience in the field of security and has written numerous articles on the topic. She is a technical editor for the iSeries edition of the eServer Magazine as well as a subject matter expert on security for COMMON, security editor for the IBM Experts Journal, contributing author on security for iSeriesNEWS and the security expert for search400.

Carol can be reached at carol.woodbury@skyviewpartners.com

Patrick Botz
Pat is a Senior Technical Staff Member responsible for OS/400 Security Architecture and a member of IBM's eServer Security Architecture Team building On-Demand security functions. Pat has worked on the architecture, design and development of OS/400 Security for 10 years. Prior to that, Pat spent 12 years administering and developing applications for distributed UNIX ™ workstations. Pat started his career in Silicon Valley in 1983 doing compiler maintenance and debugging tool development. He then worked for a supercomputer manufacturer where he was responsible for all workstation-based ECAD application development and maintenance. Pat joined IBM in 1989. Pat has written numerous articles for IBM and external trade press magazines such as iSeriesNEWS. He is a regular speaker at conferences worldwide and consults with companies on the subject of OS/400 security.

Pat can be reached at botz@us.ibm.com

© SkyView Partners, Inc, 2003-2006. All Rights Reserved.