Interested in learning more about security?

SANS Institute
InfoSec Reading Room

This paper is from the SANS Institute Reading Room site. Reposting is not permitted without express written permission.

Windows Security Patch Management Case Study: Using Software Update Services to Deploy Critical Windows Updates

This paper covers the use of Microsoft Software Update Services (SUS) software to roll out Windows updates at a small office. The case study examines the recognition of a patch management problem within the organization, a look into the scope of the problem, the decision to use SUS as the software to manage patch roll outs, the technical points in rolling out patches with SUS, and the use of the system through the following months. Also, there is a look at how effective the software is, and why only examining software ...

Copyright SANS Institute

Author Retains Full Rights

AD

Table of Contents 1
Michael_Shepherd_GSEC.doc 2

© SANS Institute 2005, Author retains full rights.

Windows Security Patch Management Case Study: Using Software Update Services to Deploy Critical Windows Updates

Michael Shepherd

January 8, 2005

© SANS Institute 2005, Author retains full rights.

© SANS Institute 2005 Author retains full rights.

Abstract

This paper covers the use of Microsoft Software Update Services (SUS) software to roll out Windows updates at a small office. The case study examines the recognition of a patch management problem within the organization, a look into the scope of the problem, the decision to use SUS as the software to manage patch roll outs, the technical points in rolling out patches with SUS, and the use of the system through the following months. Also, there is a look at how effective the software is, and why only examining software tools for patch management masks a lack of an overall patch management process within this organization. A brief look at Windows Update Services (WUS), which is SUS version 2, is included as well.

Before

At the end of January 2004 I took over the position of IT Manager at a small office in downtown Washington, DC. The office is an industry membership association which employs 30 people. Each user was set up with a PC running Windows 2000, MS Office XP, and various other software packages. There were 7 Windows 2000 Servers providing various network services including email, Citrix, VPN/dial-in, Active Directory, and file and print services. There were also 4 OpenBSD servers which mainly provided layers of security to the Windows servers. The OpenBSD servers served as a firewall, an email relay, web proxy, and central logging server.

Security at the organization was in pretty good shape when I arrived. The former administrator had done a good job applying the principles of defense in depth, segmenting the network, and using hardened, security-minded OpenBSD systems as the Internet-facing, perimeter systems. There was, however, one glaring security weakness in this network. Although the exposed OpenBSD systems were up-to-date on patches, Windows server and desktop systems alike were missing critical security hotfixes, patches, and security packs.

There was no centralized, automated system for applying Windows security updates, and it seemed that the work of applying the updates manually had proved to be overwhelming. "Patch management is, without a doubt, one of the most critical and complex Windows-security-related issues these days."[1] The problem is that timely application of patches is so necessary in terms of security, while at the same time the task is extremely cumbersome and never ends.

Patching Windows systems requires testing of patches before deployment. Microsoft releases hotfixes listed as "important, moderate, or low" once a month. "Critical" patches are released as soon as they become available. Therefore, patch management is an ongoing task that may require a

[1] Policht, Marcin. "Windows Patch Management, Introduction." Server Watch. January 15, 2004. January 8, 2005 <http://www.serverwatch.com /tutorials/article.php/3299831>.

© SANS Institute 2005 Author retains full rights.

lot of effort. It is easy to get behind.

"The easy and destructive spread of worms, such as Blaster, Slammer, and Code Red, can be traced directly to exploitation of unpatched vulnerabilities."[2] These worms exploited unpatched vulnerabilities, but patches *were* available from Microsoft in every case. Consider the following table of high-profile worms:

Worm	Worm Public Discovery	MS Security Bulletin	MS Bulletin Date	Days to Patch
Sasser	April 30, 2004	MS04-011	April 13, 2004	17 days
MS Blaster	August 11, 2003	MS03-026	July 16, 2003	26 days
SQL Slammer	January 24, 2003	MS02-039	July 24, 2002	184 days
Nimda	September 18, 2001	MS01-027	May 16, 2001	125 days
Code Red	July 16, 2001	MS01-033	June 18, 2001	28 days

[3]

The facts in the table above illustrate three points. First, patching is a key component to a secure network. In the case of each worm in the chart above a huge negative impact was felt across the Internet, but machines that remained current with patches went unaffected. Secondly, in the case of SQL Slammer and Nimda where administrators had 4-6 months to test and deploy patches Microsoft was clearly providing critical updates to administrators in a timely manner. Thirdly, although Microsoft is doing a good job of providing timely updates for known exploits, some of the worms listed above were seen in the wild less than a month after Microsoft released a patch for the vulnerability they capitalized on. This last point reinforces the idea that patch management is something an administrator must have functioning continuously.

Aware of this critical need to apply patches, in the first few weeks of working in the office I downloaded and installed Microsoft's Baseline Security Analyzer on my computer, and ran a few scans of the computers on the LAN. Reports showed that, on average, servers and desktops were missing 6 months or more of Windows updates that were listed as critical security updates by Microsoft.

Upon generating these reports I would select a few of the desktop systems and go to the Windows Update website on those machines. I wanted not just to patch the systems, but to verify that the vulnerabilities determined by the MBSA matched the list of missing patches generated by the Windows Update website. The two lists always matched verifying that there was a serious wealth of vulnerabilities on this network. This also gave me confidence

[2] SANS Institue. "The SANS Top 20 Internet Security Vulnerabilities." SANS Institute. October 8, 2004. January 8, 2005 <http://www.sans.org/top20>.

[3] Microsoft Corporation. "Patch Management Process." Microsoft. January 29, 2004. January 8, 2005 <http://www.microsoft.com/technet/security/ guidance/secmod193.mspx>.

© SANS Institute 2005 Author retains full rights.

in MBSA as a scanner I could use to produce reliable data on the patch-level of my systems.

The attention to security on the perimeter and perhaps luck had kept a worm or hacker from exploiting these vulnerabilities to this point, but a solution to keep Windows servers and desktops up to date was necessary. Given the small IT budget at this organization it was understandable that the former administrator had not spent money to implement one of the commercial products available. However, some tool was needed to facilitate the roll-out of Windows updates.

During

Given the situation there were two ways to proceed. First of all, I could simply focus on the short-term and get the systems all patched. This could be done manually which would require 10 or 15 minutes at each desktop to access the Windows Update website and apply patches, and would need to be done when users were not on the machines. For the entire, office that would mean devoting an entire Saturday to getting the systems patched.

While this would get my feet back under me it would really not solve much of anything. Within a month Microsoft would release some new updates. Would I devote a Saturday a month to the tedious task of running Windows Update on each desktop and server manually? What if a critical update appeared mid-month? I would have to always be ready to drop weekend plans to update systems.

Instead of assuming a policy of manual patch management and eventually succumbing to the monumental effort and letting systems go unpatched for a few weeks or more, I rather easily decided that a form of automatic patch management was necessary. This of course left a lot of questions still. There was the Automatic Updates feature in Windows 2000 SP4, Microsoft Software Update Services, Microsoft Systems Management Server, and many 3rd-party products which can apply Windows patches.

My first thought was to look at the possibility of using Automatic Updates to automate patching. There were 2 main problems with this. I would have 30 computers all downloading the same software off the Internet. The bandwidth would be choked with unnecessary copies of the same download. The other problem was that I would not have any control whatsoever over the updates being installed. Turning on Automatic Updates would mean that all relevant updates would be installed. Even if one update caused a problem and I uninstalled it, Automatic Updates would reinstall it the next time it connected to the MS site and determines the uninstalled patch was missing.

Automatic Updates did not seem to be a very good choice for keeping systems patched throughout the office. At the same time a 3rd party product did not seem like the best idea either. The features of these products are geared to large organizations with hundreds or thousands of users. They are incredibly scalable, and offer inventorying systems with robust database backends whereas I could easily manage inventory in an Access database or even an Excel spreadsheet.

© SANS Institute 2005 Author retains full rights.

All of these features and scalability come at a high cost as well. I had a small budget and none of it was allocated for patch management software when I arrived. Besides, I was in a bit of a crunch to get these systems patched. This meant there was not a lot of time to spend researching and testing various products from a multitude of vendors. However, any commercial product would have cost a significant portion of my budget so jumping into a solution headlong could prove to be an extremely costly mistake.

I was aware of Microsoft Software Update Services though, and since it was free it deserved a good look. First and foremost, SUS centralizes control over the patching process in the hands of the network administrator. If I were using Automatic Updates and found 1 patch within a download batch that I didn't want I'd have no way to get out of installing it unless I turned off Automatic Updates. With SUS I can simply log onto the SUS server, and select each particular new update to approve for distribution. As long as I have implemented Group Policy properly each machine will install those updates on the schedule I specified.

SUS does not support any inventorying, it cannot patch anything but the Windows operating system, and even the patch management features are minimal, but given that SUS provided this basic level of functionality at no cost it seemed like the clear choice for this small environment. "Since SUS did not provide any inventory reporting functionality, it must be combined with MBSA or a comparable utility."[4] I would continue to scan the office with MBSA in order to determine if patches were being applied or not, and at this point I was confident in the reported patch-levels it was reporting.

It was time to get started getting the office updated with patches and outfitted with an automated patch management system. I began by reading the white paper from the Microsoft web site detailing how to set up and use Software Update Service in an organization. The white paper is available here from the Microsoft website, and provides clear instructions concerning all aspects of the SUS deployment process including the key GPO settings that should be used in an Active Directory environment.

I had a Windows 2000 Server that was running our backups and was also set up as the Symantec Antivirus Server. I did not want to run SUS on either of our domain controllers which were already providing other services, nor did I want to add a new role to our file and print server. So the "utility" server that was running backups and acting as our AV server was the best choice for this additional task.

I began by installing IIS on the machine, and then I downloaded and ran the IIS lockdown tool. After that I downloaded the latest version of SUS which included SP1 to the server, and began the installation following the directions outlined in the deployment white paper. The setup is pretty straightforward. Luckily, when the server was set up originally there was an 8GB partition created where extra programs like Symantec AV were installed. I made sure to

---

[4] Policht, Marcin. "Windows Patch Management, Introduction." Server Watch. January 15, 2004. January 8, 2005 <http://www.serverwatch.com /tutorials/article.php/3299831>.

© SANS Institute 2005 Author retains full rights.

install SUS to this partition and this ensures that all of the content that SUS downloads from Windows Update and pushes to clients is stored on this partition. Otherwise all of the content would be stored on the system partition which is acceptable, but not a great idea.

When the installation finished and the SUS administration page appeared I went to the Set Options page. Here I set the server name to its NetBIOS name. I kept the proxy server set to automatic since our proxy operates transparently, and I left the server set to update content directly from Windows update. Then I set the option to automatically approve new versions of previously approved updates, and only handle English language updates.

Then I went to the synchronize server page, and set up a synchronization schedule. The SUS server checks Windows Update for new patches every day at 2 AM. Then I selected Synchronize Now. Since the server had never synchronized before it had to download metadata and the actual patch for every available update to Windows 2000/XP/2003 at this point so I let it run and left for the day.

The next day the server was done synchronizing and I began sorting through the long list of updates. I approved all of the available updates to start. After this I would allow monthly updates to sit unapproved on the server to allow some time for me to discover if there were any patches that might cause problems. I do not have a lab to test in, the time is simply allotted so that if other administrators are having problems I will have a chance to hear or read about it. This is an undesirable scenario and will be discussed again in the "After" section.

The only thing left to do at this point was to have all the client machines start pulling updates from this SUS server. Since all desktops were running Windows 2000 with Service Pack 3, or more often Service Pack 4, the updated Automatic Updates client that is needed for this to work was already installed. One point I should note here is that I did not want to have SUS push updates to my servers at the same time it pushed updates to all the user desktops. I wanted to be able to have updates go to clients on weeknights if necessary, but if a server went offline either due to a faulty patch or because it could not be restarted I would walk in to a disaster the next day. If the same thing happened to one or two desktops it wouldn't be much of a problem.

To accommodate this I placed all of the servers in one OU in Active Directory and I placed all of the desktops into another OU. When I set up group policy to control SUS I made sure that only the desktop OU was linked to the SUS GPO. This was sort of a way to allow me to target deployment even though targeted deployment is not a feature of the current version of SUS.

There are only 3 GPO settings necessary to have Software Update Services function properly. First, I created a GPO and linked it to my desktop OU. Then I set the following configuration:

[Computer Configuration\Administrative Templates\Windows Components\Windows Update\Configure Automatic Updates]
This option was set to "Enabled" and "04 – Auto download and schedule the

© SANS Institute 2005 Author retains full rights.

installation." The schedule was set for every day at 3AM.[5]

[Computer Configuration\Administrative Templates\Windows Components\Windows Update\Specify intranet Microsoft update service location]
This was set to enabled and the URL for the newly minted SUS server was used.[6]

The other 2 GPO settings under Windows Updates were left unchanged. I was going to start training users to leave their computers on all the time, and simply log out when they left the office. So there was no need to put settings in place for accommodating user schedules and making contingencies in case a user was on the system at the time. This would all happen at 3AM when the user was at home asleep. As a side note, getting users to leave their systems on all the time allows administrators to perform automated, network-wide AV scans during off-hours as well.

In any case, the GPO was now in place. The SUS server was in place. All I needed to do was start teaching users to log out at night instead of shutting down. This process took several weeks to get 100% compliance although most users caught on within the first few days. I used MBSA to run scans in the following days and weeks to see which machines were still not fully patched. When someone's machine had not received a batch of updates I would go ask if they had left their computer on overnight. Invariably the response was "oh, you want us to do that *every* night?" Yes, every night means every night.

The fact of the matter is I was glad I was getting that response. It meant that SUS and the GPO were functioning flawlessly, and all I needed was some time to get users on board with the new policy. By about 2 months after the rollout I stopped seeing machines miss updates. Users were 100% compliant.

As I said, I did not use this method to keep servers up to date. For those I usually log in remotely over the weekend and use Terminal Server to log on to each one and run Windows Update the old fashioned way. This is only manageable for a small number of servers, and I need to be prepared to head into the office if a system does not come back online when a reboot is needed. Still, it's better than having a server go belly up on a Tuesday night and having to fix it Wednesday morning when you have users justifiably clamoring for service immediately.

After

The SUS server deployment described here went online and started

---

[5] Microsoft Corporation. "Software Update Services Deployment White Paper." Microsoft. January 19, 2004. January 8, 2005 <http://www.microsoft.com/windowsserversystem/sus/susdeployment.mspx>.

[6] Microsoft Corporation. "Software Update Services Deployment White Paper." Microsoft. January 19, 2004. January 8, 2005 <http://www.microsoft.com/windowsserversystem/sus/susdeployment.mspx>.

© SANS Institute 2005 Author retains full rights.

pushing updates at the end of February in 2004. Two months later the Sasser worm hit the Internet. All of the Windows systems in the office were patched against the worm as a result of having a useful patch management tool in place, and none of the machines was infected by the Sasser worm. Even if the update had not been approved and pushed out by the time the worm hit I could have heard the first rumors of it, and used this existing infrastructure to get every system patched without too much issue or delay.

The fact that all of the systems were patched should not take credit for the worm not infecting the network though. The fact is that most of the other layers of a "defense in depth" strategy were already in place when I took over the job. A centralized AV server was set up and all Windows systems were running as managed clients. This allowed me to see at any moment if all systems were up to date on their virus definitions and whether or not any clients had detected a threat. Hardened perimeter systems left very few holes for worms or other attackers to enter the network, and effective email filtering made sure that most email-borne viruses never reached the desktop.

However, the fact remains that if any of these other layers had failed, the Sasser exploit would search in futility for a vulnerability that did not exist on this network any more. This fact remains today. There will certainly be another worm of epic proportions. This network has the infrastructure in place to patch against the vulnerability that worm will attempt to exploit.

Now for the bad news. In retrospect, there is a lot to be desired here. Some of these deficiencies I have been aware of the whole year as I've been using this system, but others have come to light as a result of researching for this paper, and reading what others have to say about patch management.

Therefore, instead of focusing on what went right here in the "After" section, I would like to focus on what went wrong or could use improvement. I think this sort of approach actually fits nicely when discussing this topic for 2 reasons. First of all, we have noted that patch management is a cyclical process in which there is no end. When one batch of patches has been deployed you look ahead to the next batch and the task of deploying them. Even as you monitor things to be sure the current changes have no ill effects, you are looking into the future. So will we, here in the "After" section. The second reason a forward-looking posture is appropriate here is because Microsoft is preparing to release SUS version 2.0, which will actually be called Windows Update Service (WUS).

The most obvious problem here is that I have no lab environment in which to test patches before deploying them to the production environment. This is a very serious issue because the lack of targeted updates in SUS means that when an update goes out to 1 of the office systems it goes out to all of them. This could easily be replaced by the statement, "when an update crashes 1 office system it crashes all of them."

The rumors about a Microsoft security patch that caused a mission critical server to crash are common in network administrator lore. However, they are more than simply tales of bogeymen. On the morning of October 27, 2004 I arrived to the office with a message that one user's computer was cycling

© SANS Institute 2005 Author retains full rights.

through the boot process over and over again. When I went to check on the situation I found that the computer was booting, getting to the point where the white Windows 2000 splash screen would appear, it would blue screen with a message about a kernel mode trap error, and begin to reboot again.

What I did to fix the problem was log onto the SUS server and look at the list of updates I had pushed out the night before (although there is not much reporting in SUS the list of approved updates shows dates patches were published by Microsoft). Then I booted into "Last Known Good Configuration" on the desktop system and uninstalled each of the new updates 1 at a time allowing the system to reboot between each uninstall. Eventually I got to the hotfix from Microsoft bulletin KB840987, and when the computer rebooted it could boot up successfully.

At that point I began reinstalling all the other new updates manually. Once that was done the computer was still functioning normally. I went back to the SUS server and changed the KB840987 hotfix from approved to not approved. As staff arrived in the morning a few more reported the same problem, and booting to "Last Known Good Configuration," and then uninstalling the KB840987 hotfix solved the problem each time. The only systems affected by this were the handful of Dell Optiplex GX150 systems we had on the floor so luckily the damage was limited.

This situation does illustrate the need for proper testing of patches using a lab that closely mirrors your production network. The lab should be equipped with hardware and software representative of the different configurations that are found in use in your organization. Updates should be applied in the lab as soon as possible after publication in order to ensure ample time for testing before they need to go into production.

This point also brings us to something that was not clear to me at first, but I have discovered through reading and research for this paper. "'We see people looking for a tool that will solve all their problems, but what you need is a process; it's not just about the tool,' says Felicia Nicastro, senior network systems consultant for International Network Services, a consulting firm that kicked off a patch management service in September."[7]

This is basically what I did. I looked for a tool instead of implementing a sustainable process. The tool itself has proven very useful, but a process would really act as a manual on how to use the tool. Also, when a tool becomes outdated you do not lose everything. You keep the overarching patch management process which is the backbone of what makes your patch management a success, and in fact you lose very little when the tool needs to be replaced.

Microsoft has a rather detailed 4-stage patch management process which it advocates. This process entails the same basic tasks outlined by the industry leaders in the Network World Fusion article. Administrators need to maintain an inventory of their network hardware and software so that they know what

---

[7] Fontana, John. "How to handle patch management." Network World Fusion. December 1, 2003. Network World. January 8, 2005 <http://www.nwfusion.com/research/2003/1201howtopatch.html >.

© SANS Institute 2005 Author retains full rights.

configurations are in production at all times. This inventory can help administrators create a lab environment which accurately mimics their production environment, and that will ensure that the testing process is more effective in weeding out problems.

Administrators also need to identify patches. This means not only recognizing what patches are out and need to be applied, but it also means identifying the most critical patches. Often, the critical patches need to be prioritized in testing so they can be deployed more quickly. Naturally, what follows identification is the testing phase.

Finally, administrators need to deploy patches. An effective way to do this is in a staged rollout where a few groups of systems receive the update in waves. This puts less of a burden on IT staff if there is a problem, but it requires a deployment tool that allows for targeted deployments. Part of the deployment phase is also the post-mortem follow-up where the network is closely monitored, and any irregularities need to be investigated for possibly links to the patch deployment. Reporting is also used to ensure the patch was applied successfully to all the systems.

In reality, of these four stages, SUS is really just a tool that can be used as part of the testing and deployment phases. It is not a patch management *solution*, but a useful part of the overall patch management process. Had I put a patch management process in place I could have avoided the crashes I saw in October. It would also mean I would have an inventory of the network which would not only prove useful in patching, but in other areas such as budgeting and planning as well.

In addition, having a patch management process in place would likely mean that transitioning to a new patch management tool would be easier. For one, I would not lose the structure of the process which would provide stability in the time of transition. It would also help hone a search for a new patch management tool. The specifics of the process would require certain capabilities from the tool, or would at least be aided by them. I could then begin looking for a new patch management tool with something more to go by then budget restrictions. I would know what features were most important or helpful to the process I had in place, and could try to find software that provided those capabilities.

So, the discussion of new patch management software brings us to a look ahead at the replacement to SUS- Windows Update Service. This new version will bring a slew of new features and enhancements. To start, Microsoft is trying to create a standard install routine for all of it's patches and will be cutting down from several different installers to just 2. One of the effects of this as well as other program design goals in the patch development team is fewer patches that require reboots.[8]

The backend to Microsoft's various patch management products will also become standardized. WUS will use the same backend patch database as

---

[8] Thurrott, Paul. "What You Need to Know About Windows Update Services." Windows IT Pro Magazine. April 2004. January 8, 2005 <http://www.windowsitpro.com/Article/ArticleID/41969/41969.html>.

© SANS Institute 2005 Author retains full rights.

SMS, and the new version of their vulnerability scanner, MBSA 2.0. This will ensure that their entire line of products produce consistent, reliable results. Again, this is not something I have found a problem with in my testing, but of course I have a very small office to support.

I think the most incredible new feature WUS will bring is that it will support a wider range of patches. SUS was only useful for rolling out security patches for Windows. WUS, however, will allow administrators to roll out Microsoft Office, SQL Server, and even Exchange Server updates. This feature alone is worth making the upgrade for since Windows updates are only a fraction of the software patches administrators need to apply. Although the Windows operating system is the single piece of software that needs the most patches, all software needs to be updated with important patches periodically. WUS will centralize more of the patching task into a single solution which is fantastic news.

Another big feature improvement is the support for targeted updates. WUS will allow administrators to target updates to systems based on OUs in Active Directory or manually-created groups in workgroups. This will greatly improve the deployment stage as well as facilitate testing using just one WUS server and a "test lab" OU. There are added bandwidth-saving features as well, and improved scalability. All of this, and the software remains free just as SUS was.

Perhaps the only downside of WUS is the timing. According to the Windows IT Pro Magazine article WUS was supposed to be in beta testing in the summer of 2004, and would be released to the public at the end of the year. The reality is much different though. WUS did not reach beta testing until November, and will be released later this spring.

I have recently registered and downloaded the beta version of WUS. As the new budget year starts, and overdue desktop upgrades have been scheduled I will be able to create a lab from a handful of old and new systems. As I said, the lack of a testing lab was a problem I recognized early on, but was unable to remedy due to the previous administrator not budgeting for it last year. But in the coming weeks the lab should start to come together.

Testing in this lab will begin with the evaluation of the beta release of WUS. Although this means the lab will not accurately reflect production, the patches being tested will remain the same. It will also allow time to become comfortable with the new software before putting it into production.

More importantly however, a true patch management process is being developed. It begins with going through our desktop computers and standardizing the deployed software, and documenting the installed software. It includes generating some WMI scripts to use in login scripts to keep hardware and software inventories up-to-date. Documentation of the use of and programming of the scripts as well as archiving and management of inventory data will need to be developed.

WUS and MBSA will play a key role in identifying patches as well as testing them in the lab. Documentation of policies and procedures for handling normal patches, critical patches, and testing will be developed to complement

© SANS Institute 2005 Author retains full rights.

this phase. Finally, WUS will be instrumental in rolling out the patches, and WUS or MBSA will be used to generate reports showing the successes and failures of each patch deployment. There will be standards for how to store, archive, and use this information.

Although this all sounds like a great deal of effort, it should make administration easier. There will no longer be surprises where a patch causes a serious problem on certain systems. Also, all of the documentation that will structure this process does not need to fill volumes. It simply needs to provide a written procedure which will guide the patch management process in minimal stress and uncertainty.

The benefit of having SUS to roll out new updates throughout the past year has been enormous. The added features in WUS will make it an even more invaluable tool in any administrator's hands. Most importantly though, these tools need to be part of a patch management process, not a substitute for it. This is the main thing I hope to remedy in the coming weeks.

© SANS Institute 2005, Author retains full rights.

© SANS Institute 2005 Author retains full rights.

Bibliography

CERT. "CERT Advisory CA-2001-19 Worm Exploiting Buffer Overflow in IIS Indexing Service DLL." CERT. July 19, 2001. January 8, 2005 <http://www.cert.org/advisories/CA-2001-19.html>.

CERT. "CERT Advisory CA-2001-26 Nimda worm." CERT. September 8, 2001. January 8, 2005 <http://www.cert.org/advisories/CA-2001-26.html>.

CERT. "CERT Advisory CA-2003-04 MS-SQL Server Worm." CERT. January 25, 2003. January 8, 2005 <http://www.cert.org/advisories/CA-2003-04.html>.

CERT. "CERT Advisory CA-2003-20 W32/Blaster worm." CERT. August 11, 2003. January 8, 2005 <http://www.cert.org/advisories/CA-2003-20.html>.

Fontana, John. "How to handle patch management." Network World Fusion. December 1, 2003. Network World. January 8, 2005 <http:// www.nwfusion.com/research/2003/1201howtopatch.html >.

Microsoft Corporation. "Microsoft Security Bulletin MS01-027." Microsoft. May 16, 2001. January 8, 2005 <http://www.microsoft.com/technet/security/ bulletin/ MS01-027.mspx>.

Microsoft Corporation. "Microsoft Security Bulletin MS01-033." Microsoft. June 18, 2001. January 8, 2005 <http://www.microsoft.com/technet/security/ bulletin/ MS01-033.mspx>.

Microsoft Corporation. "Microsoft Security Bulletin MS02-039." Microsoft. July 24, 2002. January 8, 2005 <http://www.microsoft.com/technet/security/ bulletin/ MS02-039.mspx>.

Microsoft Corporation. "Microsoft Security Bulletin MS03-026." Microsoft. July 16, 2003. January 8, 2005 <http://www.microsoft.com/technet/security/ bulletin/ MS03-026.mspx>.

Microsoft Corporation. "Microsoft Security Bulletin MS04-011." Microsoft. April 13, 2004. January 8, 2005 <http://www.microsoft.com/technet/security/ bulletin/MS04-011.mspx>.

Microsoft Corporation. "Patch Management Process." Microsoft. January 29, 2004. January 8, 2005 <http://www.microsoft.com/technet/security/ guidance/secmod193.mspx>.

© SANS Institute 2005 Author retains full rights.

Microsoft Corporation. “Software Update Services Deployment White Paper.” Microsoft. January 19, 2004. January 8, 2005 <http://www.microsoft.com/windowsserversystem/sus/susdeployment.mspx>.

Policht, Marcin. “Windows Patch Management, Introduction.” Server Watch. January 15, 2004. January 8, 2005 <http://www.serverwatch.com/tutorials/article.php/3299831>.

Policht, Marcin. “Windows Patch Management, Introduction.” Server Watch. January 15, 2004. January 8, 2005 <http://www.serverwatch.com/tutorials/article.php/3304921>.

SANS Institue. “The SANS Top 20 Internet Security Vulnerabilities.” SANS Institute. October 8, 2004. January 8, 2005 <http://www.sans.org/top20>.

Thurrott, Paul. “What You Need to Know About Windows Update Services.” Windows IT Pro Magazine. April 2004. January 8, 2005 <http://www.windowsitpro.com/Article/ArticleID/41969/41969.html>.

US-CERT. “Vulnerability Note VU#753212.” US-CERT. April 13, 2004. January 8, 2005 <http://www.kb.cert.org/vuls/id/753212>.

© SANS Institute 2005, Author retains full rights.

Last Updated: June 19th, 2013

Upcoming SANS Training

Click Here for a full list of all Upcoming SANS Events by Location

Digital Forensics & Incident Response Summit 2013	**Austin, TXUS**	**Jul 09, 2013 - Jul 16, 2013**	**Live Event**
SANS London Summer 2013	**London, GB**	**Jul 09, 2013 - Jul 16, 2013**	**Live Event**
SANS Rocky Mountain 2013	**Denver, COUS**	**Jul 14, 2013 - Jul 20, 2013**	**Live Event**
SEC528 SANS Training Program for the CompTIA® New Advanced Security Practitioner (CASP) Certification	**Washington, DCUS**	**Jul 22, 2013 - Jul 26, 2013**	**Live Event**
SANS Mumbai 2013	**Mumbai, IN**	**Jul 22, 2013 - Jul 27, 2013**	**Live Event**
SANS San Francisco 2013	**San Francisco, CAUS**	**Jul 29, 2013 - Aug 03, 2013**	**Live Event**
SANS Boston 2013	**Boston, MAUS**	**Aug 05, 2013 - Aug 10, 2013**	**Live Event**
SANS SEC 560: Network Penetration Testing @ Bangalore 2013	**Bangalore, IN**	**Aug 05, 2013 - Aug 10, 2013**	**Live Event**
Critical Security Controls Summit	**Washington, DCUS**	**Aug 12, 2013 - Aug 18, 2013**	**Live Event**
Industrial Control Systems Security Training - DC	**Washington, DCUS**	**Aug 12, 2013 - Aug 16, 2013**	**Live Event**
SANS Virginia Beach 2013	**Virginia Beach, VAUS**	**Aug 19, 2013 - Aug 30, 2013**	**Live Event**
SANS Thailand 2013	**Bangkok, TH**	**Aug 19, 2013 - Aug 31, 2013**	**Live Event**
SANS Melbourne 2013	**Melbourne, AU**	**Sep 02, 2013 - Sep 07, 2013**	**Live Event**
SANS Capital City 2013	**Washington, DCUS**	**Sep 03, 2013 - Sep 08, 2013**	**Live Event**
Network Security 2013	**Las Vegas, NVUS**	**Sep 14, 2013 - Sep 23, 2013**	**Live Event**
SANS Canberra 2013	**OnlineAU**	**Jul 01, 2013 - Jul 13, 2013**	**Live Event**
SANS OnDemand	**Books & MP3s OnlyUS**	**Anytime**	**Self Paced**