

---

# The Complete Patch Management Book

---

A Technical Exploration  
and Practical Guide

**AUTHORS:** *Anne Stanton*  
President, Norwich Group  
*Susan Bradley*  
Microsoft Small Business Server MVP

*The Fundamentals of Patching – From a Technical and Historical Perspective*

## Introduction

Susan was having a quiet evening at home, a Friday night in late January, just trying to pay for an eBay purchase. “Dang, it’s still not going through”, she said after the paypal.com web site refused to accept her payment information and was extremely slow in responding. “They must be having issues with their server, I’ll have to try it tomorrow morning”, she told her sister who had just won the online auction and wanted to pay for the item quickly. The next morning, as the computer booted up and went to Susan’s home page she read the news and found out the reason why she could not complete her transaction the night before.

“Computer worm hits the ‘Net,” screamed the headline on CNN.

SQL Slammer, Sapphire, W32.Slammer whatever you want to call it, was tiny as worm files go, only 376 bytes of code designed for speed. Typical computer transactions involve a “hand shake” transmission process. One party offers a connection, another party accepts, and the transmission proceeds by means of traditional TCP/IP processes. SQL slammer, however, used another transmission standard. It transmitted UDP packets only, through a connectionless transmission. This worm did not wait for a response. It flooded all vulnerable connections it could find.

That high-speed little file was looking for a port that had behind it, ready and waiting, a listening application. The Internet Assigned Numbers Authority (IANA.org) maintains the listing of computer ports used by programs, applications, and typical connections on the Internet. A computer system has almost 65,000 ports to transmit information back and forth. Typically, these ports sit there waiting, but sometimes they are in “listening” mode, waiting to be called upon.

Most worms would typically try to find vulnerable systems on well-known ports, those ports from 0 to 1023. SQL slammer was different. It aimed at a port not used in previous attacks, port 1434. This is a port used by database programs such as Microsoft SQL server and something called MSDE or Microsoft SQL Database Engine. Microsoft SQL Server is a very powerful database program typically run on maintained and monitored servers. The other, MSDE, it a small but powerful database engine used by developers in many applications. Furthermore, port 1434 is unique. It is not a port used to transmit data; rather it monitors SQL transmissions. All Microsoft SQL servers listen on this port. Not all MSDE installations do however.

The tiny worm’s tale includes a couple of other twists. Developers use MSDE in many applications, but do not necessarily tell purchasers that their software uses MSDE to keep track of something needed for the application’s operation. At the time Slammer struck, patching SQL server or MSDE was difficult and cumbersome, needing the patch installer to understand SQL instances. While the original patch to fix the vulnerability came out in July of 2002, the rollup service pack had just recently come out about three weeks before Slammer appeared.

In early 2003, three weeks was not enough time to have server admins or application developers test and install a service pack. Database administrators were (and still are) reluctant to install patches on working databases, so installing the rollup pack was not a priority. Furthermore, all the unpatched computers equipped with MSDE were primarily in installations where the administrator had no idea that he or she had MSDE installed. Their software vendors had not informed them, nor did they have a tool to identify machines that were running MSDE.

Thus, the stage was set for the worm: unpatched machines, unidentified machines that were also unpatched, a worm built for fast connections, a port never used before for worm attacks, and a port not used for data, only monitoring. Overall, we had a “perfect storm” for unleashing the worm. The news reports indicated that the major ISPs and backbone providers of the Internet knew within minutes that something was up. Realize that unlike Code Red that took 24 hours to go around the world infecting the globe, SQL slammer was around the world in 30 minutes.<sup>1</sup>

## What is Patch Management?

**W**e define patch management as “the act, manner or practice of managing, handling, supervising and controlling the application of code to software in order to fix a bug, especially as a temporary correction between two releases.”

Many in our industry consider that patch management means applying security fixes that are typically not temporary corrections. In fact, patch management and its processes include the entire area of introducing “new code” into an existing environment.

While this document primarily focuses on tools used for testing, supporting, and evaluating Microsoft security patches, the basic concepts are the same for all software systems. IT managers and admins must logically control the introduction of new code into a network.

Computers drive world business. No longer can a financially healthy firm exist without technology. Further, maintaining, protecting, and securing computer systems is, as recent legislative initiatives make clear, simply good business. A computer, like any mechanical device, requires regular maintenance. Applying patches to systems that connect to the Internet is simply and fundamentally mandatory. Period

This document focuses only on the processes and procedures for patch management. We assume that you have taken steps to harden systems<sup>2</sup>, apply firewalls, and install antivirus protection. Patch management is only one of the steps needed to protect an environment, but it is an essential element nonetheless. We also assume that you understand the “business” reason for patching and that you have the proper buy in from management to add patch management to your security processes.

Some security vendors recommend installing the bare minimum of patches; however, that still means that you must rely upon consistent and appropriate processes and procedures.

### History of patching

The *CERT Guide to System and Network Security Practices*<sup>3</sup> has small sections on software patches. The author, Julia Allen recommends applying patches on redundant or duplicate systems before applying them to main production machines. Any vendor service level agreement (SLA) should clearly state that firewalls should remain in place during installation of operating system patches. Allen further argues that those in charge of firewall operating systems and software need to review those devices as well for updates. Not too long ago, “best practices” meant not applying software patches in a piecemeal manner, but waiting for the cumulative security patch that, presumably, was more tested. Now, delay and deferral invite crisis.

### Definition of Microsoft patches

In the Microsoft world, patch management included all of the following types of new code introductions:<sup>4</sup>

- **Critical Update** – this is not a security update but a fix for an issue in broadly applied software. It is publicly available and has an accompanying knowledge base article.
- **Driver Update** – this updates software that supports and controls hardware. Driver updates may come from either the software vendor or the hardware vendor.
- **Feature pack** – software package that includes non-critical additions to the base software program. It typically appears between major releases.
- **Hotfixes** – patches built to address specific issues. Recipients may not distribute hotfixes outside their organizations without written authorization from Microsoft. Get them FREE

- by calling Microsoft Product Support Services. Hotfixes do not receive the regular testing process.
- **Security Update** – this is what we traditionally refer to in patch management circles. There is a severity rating included with each update, as is a security bulletin discussing the issue and a Knowledge Base article describing the patch in detail.
  - **Service Packs** – are cumulative packages of hotfixes, security updates, critical updates, and updates. A service pack undergoes both internal and external testing.
  - **Software Update** – is any update, update rollup, service pack, feature pack, critical update, security update, or hotfix.
  - **Update** – addresses a non-critical, non-security issue.
  - **Update rollup** – cumulative package of hotfixes, security updates, critical updates, and updates, collectively tested for easy deployment.
  - **Upgrade** – this software updates and upgrades an application to a newer version while keeping the settings and data from the prior program.

Each of these categories requires the same process and procedures of testing, acceptance, and management sign off. Each must go through a process no matter the size of an organization. We will spend the bulk of our time in this document on security updates.

### What is a patch?

Software refers to the instructions mechanical devices receive to process commands in a certain manner. Typically, developers write software to perform a process in a prescribe fashion. However, as with any process deriving from human intelligence, there can be incorrect assumptions made, and the software might not perform as intended.

As Bruce Schneier and Niels Ferguson state in *Practical Cryptography* –

Most engineers have to contend with problems like storms, heat, and wear and tear. A bridge designer only has to worry about three threats – water, gravity and wind. All of these factors affect designs, but their effect is predictable to an experienced engineer. (This is) not so in security systems. Our opponents are intelligent, clever, malicious, and devious; they'll do things nobody had ever thought of before. They don't play by the rules, and they are completely unpredictable. That is a much harder environment to work in.<sup>5</sup>

A software designer has to worry about threats from an unlimited number of vectors. This means that the network administrator must consider such risks from these same threat vectors when analyzing a network's need for patches. Can the threats come from outside the organization or only from inside? What potential attack method might exploit a particular flaw? In a later chapter, we discuss resources for determining these threat vectors.

Software flaws threaten an organization in various ways. The most critical give an attacker full rights to a system. Many of these flaws stem from buffer overflows. Traditionally stack-based buffer overflows have been the largest category of security issues, and are places in the software where more data enters the system than the software is asking for. If the software designer did not anticipate this, the system would "crash." Perpetrators target buffer overflows to dump the processes to ensure that the system remains at an "administrative privilege" level, thereby forcing the system into "handing over the keys to the kingdom." When a patch bulletin indicates that the worse case scenario is that the attacker can "run code of his choice," this is the equivalent to that person logging in as administrator to a system.

Software patches themselves are also threats. After each security bulletin release, even with the rigorous testing done by the vendors on those applications typically broken by previous security updates, issues still occur. Documentation accompanying a security update addresses any known issues likely to develop following release of the update. Remember that issues directly

caused by security updates qualify for a **no-charge** support call. In the United States, call 1-866-PC-SAFETY. To resolve International issues contact a local Microsoft office.<sup>6</sup>

Testing patches, ensuring that they do not adversely affect systems and that they protect systems as intended, as well as applying patches, requires approval from management and may even require approval from critical line of business vendors. We discuss change management processes later and in detail. Nevertheless, you need adequate resources if these testing, ensuring, and applying processes and procedures are to work correctly.

### Identifying the flaw

Very briefly, software vulnerability begins when someone looks at code or attempts to reverse engineer code. While Linux and its variants work under the Open Source licensing model wherein the source code accompanies the product, Microsoft does not release its source code. Thus, many security researchers use various techniques to identify flaws. In some cases, the interaction and connectivity needed between a UNIX or Linux system may provide clues to a researcher for potential flaws. Other freely available tools include Dave Aitel's SPIKE ([www.immunitysec.com](http://www.immunitysec.com)), Todd Sabin's DCE-RPC tools (<http://razor.bindview.com>), Netcat ([www.atstake.com/research/tools\\_/network\\_utilities](http://www.atstake.com/research/tools_/network_utilities)), Ethereal ([www.ethereal.com](http://www.ethereal.com)), and many of the utilities at the Sysinternals web site ([www.sysinternals.com](http://www.sysinternals.com)).

If the researcher or security company agrees to responsible disclosure techniques, it contacts the software vendor ahead of time and allows the vendor to correct the flaw or respond to the issue. Eeye.com<sup>7</sup> is one vendor that notifies but does not disclose a flaw publicly at any time prior to the release of a security update. Irresponsible vendors and security researchers post the vulnerability to listserves such as Full Disclosure<sup>8</sup> along with information that provides a "proof of concept" that has later been exploited by others and turned into automated exploits. Once contacted, the vendor reviews the reported information to see if the issue truly is a security flaw. If there is an issue, the process of building the patch and testing the patch begins.

### Why do we patch?

It is obvious that we patch because software is not processing commands correctly. This mis-processing could range from elevation of privilege to information disclosure. *Threat Modeling*, a text that explores what an adversary might attain by exploiting a flaw defines the following threat categories<sup>9</sup>

- Spoofing identity
- Tampering with data (also called integrity threats)
- Repudiation
- Information disclosure
- Denial of service
- Elevation of privilege

Patch management ensures that correct code replaces incorrect code. However, it is not the only way to reduce risk. The patch management process also includes mitigation techniques that are not actual patches but include additional procedures to protect networks if the patch is not available, or if admins cannot apply it to a network, or if there are other reasons that preclude applying the patch.

### What is included in a Microsoft patch?

Let's roll up our sleeves, get technical, and examine what is included in each type of Microsoft patch<sup>10</sup>. Security patches, critical updates, updates, update rollups, drivers, and feature packs fall into the general distribution releases (GDR) category. These go through testing across different

platforms and applications to ensure proper functionality, and that the program or update that includes new features performs as intended. However, Hotfixes developed by Microsoft Product Support Services for a specific situation are not as tested as those included in general distribution releases. Microsoft Knowledge Base articles, freely available from Microsoft Product Support Services, always accompany these **QFEs**.

In the Windows 2003 Server environment, the product update packages may include two or more copies of the same files to support two different types of install environments for a system. When the security patch, critical update, update, update rollup, driver, or feature pack install, the installer package looks to see what files already exist on a system. Possible install environments include:

- GDR environment
  - Original released version (RTM)
  - Service pack version
  - General Distribution release
- QFE environment
  - Hotfix

Having discovered the appropriate environment, the installer package installs the applicable file set. To see what version of a file exists in a Windows 2003 server environment, review the following formats:<sup>11</sup>

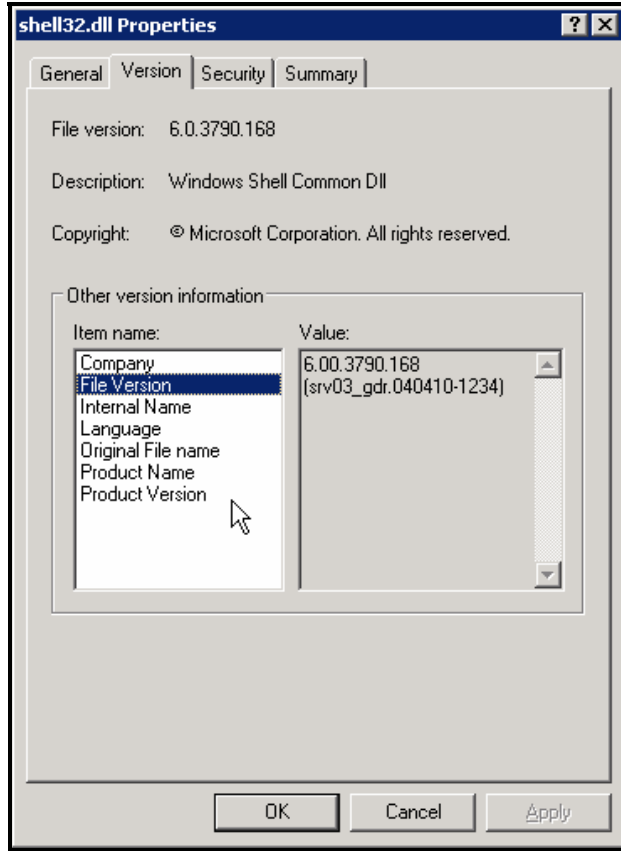
File Version	Source of file
Srv03_rtm.mmmmmm-nnnn	This file is from the original RTM version of the product and has not been updated by any security patch, critical update, update, update rollup, driver, feature pack or hotfix.
Srv03_gdr.mmmmmm-nnnn	This indicates that the file is from a security patch, critical update, update, update rollup, driver, or feature pack and has not been updated by a hotfix.
Srv03_spx.mmmmmm-nnnn	This indicates that the file is from a SP and has not been updated by a security patch, critical update, update, update rollup, driver, and/or feature pack.
Srv03_qfe.mmmmmm-nnnn	This indicates that the file is from a hotfix.

In our server, we can see that the file on our server is a GDR version. Thus, it indicates that the patch engine did not find a hotfix and instead found a GDR version.

For example, let's look at the file included in Security bulletin 04-024 (04 for the 2004 year, -024 meaning the 24<sup>th</sup> bulletin of the 2004 year). Find his bulletin at <http://www.microsoft.com/technet/security/bulletin/MS04-024.mspx> and the sample below is the patch for the Windows 2003 platform. It includes updates to one file shell32.dll. Inside the installer package are two files. One expects that the server will still have one of the original dll's categorized as a GDR package the other anticipates a hotfix.

13-May-2004 00:07 6.0.3790.168 8,168,960 Shell32.dll RTMGDR  
*This version is used to apply to servers that have original released version (RTM), Service packs or General distribution versions.*

12-May-2004 23:29 6.0.3790.169 8,168,960 Shell32.dll RTMQFE  
*This version is used to apply to servers that have received a hotfix version.*



The shell32.dll File Version Window

While security bulletin 04-024 includes an update to only one file, many patches contain a series of files that replace existing files on a system. Other security patches may include a series of files needed to correct the condition. In the Security patch Microsoft Security Bulletin MS04-022: Vulnerability in Task Scheduler Could Allow Code Execution (841873)<sup>12</sup> the patch includes a series of files needed to remove the vulnerability from the system:

Date	Time	Version	Size	File name	Folder
08-Jun-2004	22:01	5.1.2600.105	48,640	Browser.dll	RTMQFE
08-Jun-2004	22:01	5.1.2600.155	251,392	Mstask.dll	RTMQFE
03-Jun-2004	22:54	5.1.2600.155	9,728	Mstinit.exe	RTMQFE
08-Jun-2004	22:01	5.1.2600.122	301,568	Netapi32.dll	RTMQFE
08-Jun-2004	22:01	5.1.2600.155	159,232	Schedsvc.dll	RTMQFE
08-Jun-2004	22:02	5.1.2600.1564	260,096	Mstask.dll	SP1QFE
08-Jun-2004	19:59	5.1.2600.1564	10,752	Mstinit.exe	SP1QFE
08-Jun-2004	22:02	5.1.2600.1562	306,688	Netapi32.dll	SP1QFE
08-Jun-2004	22:02	5.1.2600.1564	172,544	Schedsvc.dll	SP1QFE
18-May-2004	03:46	5.1.2600.1555	593,408	Xpsp2res.dll	SP1QFE

Applying new executables and dll files introduces change into a stable system. As evident from the files listed above, the security update includes both executables and dynamic link library files. An exe file is a file that a computer can directly “run” or execute. A DLL file contains a range of functions accessed by other Windows applications. The standard functions in the Windows Application Programming Interface (or API) are accessed using DLL files. This standardization eases collaboration among disparate applications. Without these building blocks, applications

would look and act much differently. A DLL can have the extension of .exe, .dll, .drv, or .fon. In any case, patching introduces new files and new code into a stable system. Thus, test to ensure that you have tested the install and uninstall processes, as well as any potential rollback issues.

### Historical Patch Process window

Until recently, an administrator could be somewhat lax in applying patches. According to Forrester Research, the average time between the release of a patch and the attack of a worm was 305 days in March 2003. However, that window of opportunity has been shortening.

**Historical window of patching in 2003<sup>13</sup>**

Common Name	Attack Date	Patch Issued	Advance Notice	Impact of Attack
SQL Slammer	1/25/03	7/24/02	185 days	Infections doubled every 8.5 seconds
Bugbear	9/30/02	5/16/01	502 days	More than 2 million infected computers
Frethem	7/17/02	5/16/01	427 days	12 variants in the first two months of activity
Yaha	6/22/02	5/16/01	402 days	Intercepted in one of every 268 emails at peak
Elkern	4/17/2002	5/16/01	336 days	Detected in more than 40 different countries
Klez	4/17/02	5/16/01	336 days	\$9 billion worldwide productivity loss
Badtrans	11/24/01	5/16/01	192 days	Message Labs has seen 458,359 instances
Nimda	9/18/01	10/17/00	336 days	Spread worldwide in 30 minutes
Code Red	7/19/01	6/18/01	31 days	Infection doubled every 37 minutes

As the table shows, in March 2003, administrators had many opportunities to test patches and even wait until a Service Pack before deploying. However, this window has been shortening in to include instances where patches were not available. More recently, the time between the patch and the worm for an exploit commonly known as MSBlaster was **16 days**. In June 2004, Microsoft's Internet Explorer browser suffered several security issues left unpatched by Microsoft for many weeks. Therefore, while stressing patch application as the best security prevention, we include remediation techniques as well in this patch management process.

### Finding out about patches

We next need to identify resources that help identify which patches an organization needs. Whether you patch manually or use patch management tools, ensure that your team or your Security officer is aware of the patches available for the products in your network. For most operating systems, this is rather easy to do, as all major vendors have e-mail or RSS notification. For Microsoft products, subscribe to get e-mail notifications at [www.microsoft.com/security/bulletins/alerts.mspx](http://www.microsoft.com/security/bulletins/alerts.mspx).

```
-----BEGIN PGP SIGNED MESSAGE-----
*****
Title: Microsoft Security Bulletin Summary for August 2004
Issued: August 10, 2004
Version Number: 1.0
Bulletin: http://go.microsoft.com/fwlink/?LinkId=29234
*****

Summary:
=====
This advisory contains information about all security updates
released this month. It is broken down by security bulletin severity.

Moderate Security Bulletins
=====

    MS04-026 - Vulnerability in Exchange Server 5.5 Outlook Web
              Access Could Allow Cross-Site Scripting and Spoofing
              Attacks (842436)

              - Affected Software:
                - Exchange Server 5.5 Service Pack 4
              - Affected Components:
                - Outlook Web Access

Update Availability:
=====
An update is available to address these issues.
For additional information, including Technical Details,
Workarounds, answers to Frequently Asked Questions,
and Update Deployment Information please read
the Microsoft Security Bulletin Summary for this
month at: http://go.microsoft.com/fwlink/?LinkId=20833

Support:
=====
Technical support is available from Microsoft Product Support
Services at 1-866-PC SAFETY (1-866-727-2338). There is no
charge for support calls associated with security updates.
International customers can get support from their local Microsoft
subsidiaries. Phone numbers for international support can be found
at: http://support.microsoft.com/common/international.aspx
```

Sample of a Security bulletin

Another way to stay current is to sign up for RSS or “really simple syndication.” Using a feed reader, you can receive these security bulletins quickly and easily. For more information about choosing and installing an RSS reader go to [www.microsoft.com/technet/security/bulletin/secrssinfo.msp](http://www.microsoft.com/technet/security/bulletin/secrssinfo.msp).



RSS feed of Microsoft Security bulletins (Outlook 2003 using Newsgator)

Microsoft moved to a once a month patch schedule that has patches releases on the second Tuesday of the month between 10:00 a.m. and 11:00 a.m. However, there may be times that a patch will be released “out of band” if an active exploit is “in the wild.”

For the Red Hat Enterprise Linux platform, there is also many ways to obtain notifications directly from the vendor by subscribing to email or RSS feed at the following web site: [www.redhat.com/security/team/advisories.html](http://www.redhat.com/security/team/advisories.html)<sup>14</sup>



RSS feed of Red Hat security bulletins (Outlook 2003 using Newsgator)

Software vendors often provide additional support resources. For some applications installed on your network, it may be more difficult to track down notification sources. Secunia.com, for example does provide security information for quite a few vendors at [secunia.com/vendor/](http://secunia.com/vendor/). For your “line of business” applications, you will need to contact the vendors regarding the manner in which they notify customers.

## The Road Ahead

In the coming sections we will discuss patch methodologies, risk management as applied to patching, and development of a patch team. We will analyze a security bulletin to find resources to determine if exploit code is already on the Internet and thus having an impact on a patch timing decision. We will discuss ways to set up a lab environment for testing patches and the procedures that you should review when ensuring that the security updates perform as expected. We will point to resources for proof of concept code or other techniques you can use to confirm patch status. We will discuss ways of enforcing patch management policy. We will discuss cases where you cannot patch because of vendor limitations or the lack of a patch, and what mitigation information and resources you can look for. While we focus primarily on Microsoft technologies, we will be vendor neutral in the use and recommendation of patching tools and give you resources to assist you in making your choice.

Whatever technology or process you use to manage patches, both the application and the reporting are key elements in ensuring that systems remain safe and secure. Ensuring that the patch or mediation technique performs as expected is central to any patch strategy.

---

<sup>1</sup>“Analysis of the Sapphire Worm - A joint effort of CAIDA, ICSI, Silicon Defense, UC Berkeley EECS and UC San Diego CSE,” retrieved August 29, 2004 from <http://www.caida.org/analysis/security/sapphire/>

<sup>2</sup>“Threats and Countermeasures: Security Settings in Windows Server 2003 and Windows XP,” (Redmond, WA: Microsoft, Inc., 2004) from <http://www.microsoft.com/downloads/details.aspx?FamilyId=1B6ACF93-147A-4481-9346-F93A4081EEA8&displaylang=en>

<sup>3</sup>Allen, Julia H. *The CERT Guide to System and Network Security Practices*, New York: Pearson Education, 2001.

<sup>4</sup>“Description of the standard terminology that is used to describe Microsoft software updates,” (Redmond, WA: Microsoft, Inc., 2004) from <http://support.microsoft.com/default.aspx?kbid=824684>

<sup>5</sup>Ferguson, Niels and Bruce Schneier *Practical Cryptography* New York: John Wiley & Sons, Inc., 2003.

<sup>6</sup>“10-Step Emergency Response Plan for Security Attacks” Redmond, WA: Microsoft Corporation, 2004 from <http://members.microsoft.com/Partner/Campaign/SecurityOutreach/sell/10-StepEmergencyResponsePlanforSecurityAttacks.pdf>

<sup>7</sup>“Upcoming Advisories” Aliso Viejo, CA: eEye Digital Security, 2004 from <http://www.eeye.com/html/research/upcoming/index.html>

<sup>8</sup><http://lists.netsys.com/mailman/listinfo/full-disclosure>

<sup>9</sup>Swiderski, Frank and Window Snyder *Threat Modeling* Redmond, WA: Microsoft Press 2004.

<sup>10</sup>While we will discuss alternative OS, due to market share and desktop impact, the bulk of the discussion will surround the Microsoft operating systems.

<sup>11</sup>“Description of the Contents of a Windows Server 2003 Product Update Package” Redmond, WA: Microsoft Corporation, 2004 <http://support.microsoft.com/default.aspx?scid=kb;en-us;824994>

<sup>12</sup>“Microsoft Security Bulletin MS04-022” Redmond, WA: Microsoft Corporation, 2004 <http://www.microsoft.com/technet/security/bulletin/MS04-022.mspx>

<sup>13</sup>Koetzle, Laura, Ted Schadler, Charles Rutstein, and Robert Whiteley *Can Microsoft Be Secure?* Cambridge, MA: Forrester Research, March 2003

<http://www.forrester.com/ER/Research/Report/0,1338,16275,FF.html>

<sup>14</sup>For purposes of this document we use only Redhat’s Enterprise Linux distribution in the examples, however all of the Linux distributions provide similar services