Patching for Low-Bandwidth Communities: Maintaining security in developing countries' emerging technology markets.

Dominic White <project@singe.rucus.net>, Barry Irwin <b.irwin@ru.ac.za>
Computer Science Department, Rhodes University
Grahamstown, 6140
Tel: +2746-603-8291, Fax: +2746-636-1915

Abstract— **This paper identifies ways in which the current boon in patch management work can be used to benefit low bandwidth communities. This is particularly relevant to emerging information economies such as South Africa that do not have the broadband connectivity assumed by many existing patch management tools. The potential damage caused by malicious software is of as large a concern in this environment as in broadband environments, but for different reasons. This paper provides several recommendations on how patching can be made simpler and more viable in a low bandwidth environment.**

I. Introduction

SINCE the often referred to SQL Slammer worm of 2002[1] there has been a sharp increase in research and work into patching software. A large patch management industry has grown out of this, with sales hitting $80 million in 2003, and is projected to grow by 20% to 30% every year for the next four years[2]. The focus of the field has been to help system administrators patch corporate networks, with some work done by Microsoft and the Open Source community on keeping the home user up to date. However, much of this effort has assumed access to broadband connectivity. In bandwidth starved countries such as South Africa and other emerging information economies (Brazil, India etc.) where many small organisations have only an expensive dial-up or ISDN line, spending several hours downloading security patches is not a suitable solution and results in patches not being applied. Unfortunately, there has been a rise in malicious software, this malware[1] can cause a variety of problems these organisations cannot afford, for example; a significant or complete loss in computer performance, loss of connectivity, malware causing the machine to join a malicious network of computers (called a botnet), theft of financial details and being used as a SPAM relay. This paper will discuss methods of making patching in low bandwidth environments more viable. This is particularly important in third world countries with emerging technology markets who hope to compete in the international information economy. The discussion will try to remain operating system generic as this problem affects all operating system including Microsoft Windows, Red Hat, Debian and Ubuntu Linux.

II. Problem Description - low bandwidth makes patching hard

To better describe the problem, two brief examples will be discussed. The first is of a rural school with four donated computers, a modem for connectivity and a lack of technical savvy administrators. If their machines were to stop functioning or have their performance degraded it would be difficult for them to find someone to repair the situation. It is also difficult for them to download patches over a dial-up line. This is for two reasons. The first is the prohibitive cost of spending a few hours a day online and the second is the technical expertise required to patch successfully, for example setting the modem up to connect to the Internet regularly to automatically download patches.

The second example is of a small business selling stationary with two computers and an ISDN line. Due to the faster connection these machines are likely to spend more time connected to the Internet and will thus be more vulnerable to malicious software. As with the previous example the loss or significant drop in performance of a machine can be costly, especially if they contain business critical information. Additionally, identity theft and bank fraud can be very real threats, an example of this is the loss of several thousand rands from ABSA accounts in 2003 due to a compromise of end users machine's[3]. In this scenario it is still costly to download patches, but it becomes more complicated to download them more frequently due to the increased web usage.

In these examples the bandwidth being described refers to the external Internet connectivity and not the internal LAN connectivity, which, with the low cost of network interface cards, is usually quite high. Thus the focus of this paper will be in decreasing the effect of large patches on external bandwidth as this is where the problem lies. It must be emphasised that patching is not a panacea and other best practices such as desktop firewalls and anti-virus software should be effectively employed. However, those too need to be patched and updated.

III. Possible Solutions

This section will discuss possible solutions to the problem. These solutions fall into three categories:

- Methods of making the patches smaller.
- Methods to reduce the number of patches required.
- Methods to increase the available bandwidth.

A. Making Patches Smaller - Binary Patching

Traditionally patches are distributed by packaging files to be replaced instead of packaging the differences between the two versions. The advantage of the traditional method is that the same package can be used to upgrade from any (or many) previous versions or for new users to perform a fresh install. Thus, the software maintainer's job is made easier. However, if the difference from one version to the next is only a small change, the user will still be forced to download a full copy of the new software. An alternative is to package the incremental difference between the two files: this would result in smaller patches, particularly when only a small change has been made, as is often the case with security patches. Below is a comparison of two binary patching tools, namely Xdelta[4] and bsdiff[5]. As can be seen in the table (I) , the binary patches provide anywhere between a 90% to 25% reduction in size compared to a full binary download. The last example was a test case where two completely different files were used (i.e. there were no similarities between the two files).

However, there are some disadvantages to binary patches. A binary patch can only patch from one specific version to another, thus if the end user is likely to have several different versions of a vulnerable software package, multiple binary patches may have to be distributed.

This research is supported by DAAD, the NRF and the Center of Excellence in Distributed Multimedia at Rhodes University, Grahamstown

[1]Malicious software such as viruses, worms, spyware or trojans.

	Patch Tool					
	bzip2 compressed download		xdelta		bsdiff	
Binary	bytes	percent	bytes	percent	bytes	percent
gaim	317 699	100%	3 877	1.22%	782	0.25%
gaim-remote	4 979	100%	157	3.15%	140	2.81%
lsusb	20 673	100%	17 837	86.28%	15 731	76.09%
usbmodules	5 040	100%	3 815	75.69%	2 944	58.41%
BSD ls -> GNU ls	36 026	100%	36 919	102.48%	37 604	104.38%

TABLE I
TABLE COMPARING FILE SIZES OF DIFFERENT METHODS OF DISTRIBUTING THE SAME FILE.

	Year						
Kernel Version	1999	2000	2001	2002	2003	2004	2005
2.2	3	4	17	2	1	0	3
2.4	n/a	1	6	5	12	30	11
2.6	n/a	n/a	n/a	n/a	2	33	35
Total	6	7	20	8	15	50	40

data source: http://cve.mitre.org/

TABLE II
TABLE DEPICTING VULNERABILITIES IN THE DIFFERENT LINUX KERNEL VERSIONS OVER TIME.

This can sometimes make a binary patch larger than a traditional patch, this is certainly the case with Microsoft's binary patching[6]. With careful package management this risk can often be mitigated by tailoring the delivered patches to the systems requesting them (i.e. a semi-intelligent patch tool) or by attempting to keep software versions in lock-step. The last disadvantage is that it is harder for a software maintainer to manage binary patches with one version bump requiring several binary patches to handle users who are not running the immediately previous version, as patch will be required for each version to the current. This is a process that can be fairly well automated.

B. Reducing the Number of Patches

1) Patch Selection: Not every patch that is released is applicable to an organisation. For example if there is a vulnerability in a mail client that only affects people using the IMAP mail protocol, then users of the software who do not use IMAP but rather POP3 can ignore the patch. Thus an intelligent choice of which patches should be installed can reduce the number of patches downloaded. This requires that the user make an informed decision which may not be possible in the scenarios with little technical support. This can be partially mitigated by several factors:

- Modularisation of software to allow automated tools to only patch affected modules.
- Clearer and more accessible patch information.
- Third party organisations such as Internet Service Providers providing advice or managing patches for clients.

In addition, not every patch is a security patch, some provide functionality updates. Users should attempt to minimise change by only applying patches that they require[7]. Thus, most functionality patches should be ignored unless the new functionality is required.

2) Software Selection: Some software is patched more than others. Unfortunately, it is not as simple as figuring out which software has less patches, as this is no indication of actual security. Older, more mature software, will often have a bigger use and support community which will result in more people finding vulnerabilities, and due to its popularity, more people looking for vulnerabilities. Thus, it may have more patches being released and be more secure than a new software project that fulfils the same functionality but has not had the same level of security review. End users are not always capable of making such analyses, thus, vendors should make these decisions for them and bundle appropriate mature and secure software.

To this end it is hypothesised that older software that fulfils all requirements and is still being maintained (such as the Linux kernel or the Debian stable distribution) will have less announced vulnerabilities *and* will be more secure. To test this, vulnerability data for the Linux kernel was collected from the Common Vulnerabilities and Exposures List[8] and analysed. The results seen in table II[2] demonstrate that older kernel versions have less vulnerabilities over time, and hence less patches to fix those vulnerabilities, than their newer counterparts. This is due to three primary reasons:

- There is less new functionality with potential security holes.
- The older software has existed longer and most of it's security holes have been discovered and closed.
- Fewer people are using the older version and hence less researchers are examining it.

The last point is not applicable to the scenarios in this paper as patching is reactive and will not provide a defence against a skilled attacker with access to unreleased exploits, no matter what version of the software. Thus, if and older software version is still being security maintained and provides all required functionality, it is often better to use the older version over the newer version as this will reduce the number of patches required without adversely affecting security. This is particularly true in the case of developing countries where the computers used often do not require the latest drivers and functionality.

C. Increasing Bandwidth - Patch CDs

Internet connections are not the only method in which patches can be delivered to machines that require them. It is possible to use other media such as portable storage devices (e.g. flash sticks, mp3 players, digital cameras) and removable media (e.g. CDs, DVDs, removable hard-drives). With the reduction in cost and increase in storage capacity of portable and removable media it may be more effective to travel to a place with faster connectivity and download the required patches onto a portable media. Alternatively a service could be launched where a disc is physically mailed to people requiring patches once a month. The money saved by not having to pay for a download could be used to finance this operation. Such a service is already being used to provide high latency, high bandwidth, Internet connectivity to some rural schools[9].

IV. CONCLUSION

This paper has shown that the focus of the patch management industry has been on broadband users and larger organisations, with little work being done to ensure low bandwidth communities are being effectively patched. Additionally, this paper has shown that low bandwidth communities are also at risk from malware and require regular patching. Several recommendations have been made as to how the situation can be improved for the low bandwidth user. These recommendations have focused on how to make the patches smaller, how to lower the number of patches required and how to increase the available bandwidth to these communities. Many of the ideas can be implemented in isolation. It is hoped that they can, either together or individually, provide some benefit to end-users and technology services operating in low bandwidth environments.

[2]The total columns do not add up correctly as some vulnerabilities affect multiple kernel versions. For example in 2004 there were 13 vulnerabilities which overlapped.

REFERENCES

[1] Microsoft. (2002, oct) Microsoft security bulletin MS02-061. Microsoft Corp. [Online]. Available: http://www.microsoft.com/technet/security/bulletin/MS02-061.mspx

[2] T. Strategies, "Patch management sector report," Tech. Rep., may 2004. [Online]. Available: http://www.trustedstrategies.com/nl1/rnr.php

[3] E. Lombard. (2003, jul) Hacker cleans out bank accounts. Sunday Times. [Online]. Available: http://www.suntimes.co.za/2003/07/20/news/news01.asp

[4] J. McDonald. (2005). [Online]. Available: http://xdelta.blogspot.com/

[5] C. Percival, "An automated binary security update system for freebsd," Master's thesis, Computing Lab, Oxford University, Oxford, 2003. [Online]. Available: http://www.daemonology.net/freebsd-update/binup.html

[6] Microsoft, "Binary delta compression," Tech. Rep., mar 2004. [Online]. Available: http://www.microsoft.com/downloads/details.aspx?FamilyID=4789196c-d60a-497c-ae89-101a3754bad6

[7] S. Microsystems, "Solaris patch management: Recommended strategy," Tech. Rep., aug 2004. [Online]. Available: http://docs-pdf.sun.com/817-0574-12/817-0574-12.pdf?biga=15

[8] (2005, jun) CERT/CC common vulnerabilities and exposures. Website. CERT. [Online]. Available: http://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=linux+kernel

[9] A. Rabagliati. (2004) Wizzy digital courier. [Online]. Available: http://www.wizzy.org.za/

Dominic White is an MSc student at Rhodes University. His thesis is on patch management and his interest is in computer security.is an MSc student at Rhodes University. His thesis is on patch management and his interest is in computer security.i