A PROTOTYPE FOR FILESYSTEM INTEGRITY CHECKER IN USER-SPACE MOOD

ALQAHTANI, SAEED IBRAHIM S

A project report submitted in partial fulfillment of the
requirements for the award of the degree of
Master of Computer Science (Information Security)

Faculty of Computer Science and Information Systems
Universiti Teknologi Malaysia

OCTOBER 2009

Dedicated to

My beloved parents, my darling siblings and to all whom were beside me

ACKNOWLEDGEMENT

Having the chance to be in this honourable institute is an unforgettable moments in my life. The Centre for Advanced Software Engineering at UTM was an everyday opportunity to acquire the knowledge that I look for. Many thanks to each lecturer in there; they were my guidance to achieve my goals.

I would like to use this opportunity to thank my supervisor, Dr. Md. Asri Ngadi for his insights, patience, and helpful suggestion in guiding me through all the way from the very beginning to the accomplishment of this project. All the thanks go to Mr. Syaril Nizam Omar too, for being supporting and motivate all the time.

ABSTRACT

Today, improving the security of computer systems has become a vital and challenging problem. Attackers can seriously damage the integrity of filesystems. Attack detection is complex and time-consuming for system administrators, and it is becoming more so. One of the means to detect intruder's activity is to trace all unauthorized changes in a filesystem. Current user-space mood checkers, due to being slow detectors, suffer from the opportunity gap that occurs between filesystem checks. Basing on the principle of thinking like an attacker, this prototype is developed to minimize the total time taken for checking by focusing on critical files. The proposed technique will accelerate the checking process through acquiring specific file extensions from the filesystem rather than targeting the entire filesystem. Discrepancies in the filesystem are reported after comparing current files hashing values with original hashing values. This prototype is configured to use variety of hashing algorithms to measure the performance on different scales and to provide various choices for users. Research results on Windows Server 2003 show that the average total time taken for this prototype is in the range of three to four minutes. The elapsed time of filesystem checking by Windows System File Check tool “SFC” has been decreased to eighty five percent on this prototype.

ABSTRAK

Hari ini, usaha untuk menambahbaik sistem keselamatan komputer telah menjadi semakin rumit dan mencabar. Penyerang boleh memusnahkan sesebuah integriti sistem fail secara kritikal. Pengesan serangan sangat kompleks dan menjimatkan masa untuk pengurus sistem dan menjadi semakin baik. Salah satu cara untuk mengesan aktiviti penceroboh ialah dengan mengesan semua perubahan yang tidak dikenali di dalam sistem fail. Pemeriksa mud ruang-pengguna sekarang dengan yang menjadi pengesan yang perlahan, dibebani oleh peluang ruang yang berada diantara pemeriksaan sistem fail. Mengambilkira prinsip pemikiran penyerang, prototaip ini dibangunkan untuk meminimumkan jumlah masa yang diambil untuk pemeriksaan dengan menumpukan kepada fail yang kritikal. Tujuan teknik ini adalah dengan mempercepatkan proses memeriksaan dengan mengumpul fail tambahan yang tertentu di dalam sistem fail tersebut daripada mengsasarkan kesemua sistem fail. Perbezaan di dalam sistem fail boleh dilaporkan dengan membandingkan nilai fail semasa cincang dengan nilai cincang yang asal. Prototaip ini ditetapkan untuk menggunakan pelbagai algoritma cincang untuk mengukur keupayaan dari skala yang berbeza dan menyediakan pelbagai pilihan kepada pengguna. Keputusan kajian dari *Windows Server 2003* menunjukkan bahawa purata masa yang diperlukan oleh prototaip ini adalah tiga hingga empat minit. Masa yang digunakan oleh sistem fail ini yang diperiksa oleh peranti *Windows System File Check "SFC"* telah dikurangkan kepada lapan puluh lima peratus jika menggunakan prototaip ini.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

1.1 Introduction

The health of a server needs to be assessed and protected in much the same manner as the health of a person. The server's main task is to provide services to other computer programs and their users, either in the same or other computers. Monitoring is the use of automated processes to continually collect and analyze the operation of critical server services. Monitoring is an important part of delivering a high-quality messaging service and for achieving commitments of service level agreements. The overall goal of monitoring is to avoid possible service outages by predicting problems and by quickly noticing problems that cannot be predicted. This can be accomplished by either continually polling the monitored component or by receiving or detecting events from these components.

1.2 Problem Background

Effective front-line monitoring of a server means watching the details for a large number of events and performance counters on potentially hundreds of servers. Effective monitoring requires more than simply knowing whether services are running and the databases are mounted. It is highly needed to know the details if it is required to monitor a server in a proactive rather than a reactive manner. Without detailed and diligent monitoring, our first indication of a problem will likely be a telephone call from an unhappy user or customer. At that point, system administrators probably have a service interruption, and service interruptions are bad news for an enterprise as an example.

The amount of hacking activity on the Internet has been revealed after one company set up an anonymous “dummy test” server and found it was maliciously attacked 467 times within 24 hours of being installed. That news has been published on ZDNet website by Graham Hayday [17], the server which contained no data and had no public profile, was attacked every single day over the next three weeks. All these statistics show that servers are still targeted by attacks day after day.

Another evidence of the activities against servers that are derived from the statistics that Cyber Security Malaysia website provides [23] proves that many breaches have been growing on the recent years. System intrusions represent big numbers year after year and that explain how much servers are suffering.

All those numbers and figures point that servers are still in need to be audited and scanned regularly. It has been mentioned by Don Mosley [28] when he claimed that the area of real-time Intrusion Detection and Prevention utilizing intelligent routers or various network attached appliances has received much press in the last few years. He continued “should any of these defenses provide less than 100% effective coverage the user will be left unaware of any 'mischief' that might have gotten through. There is still a need for non real-time scanning of system files to determine any unauthorized modifications. This type of audit is often the only effective way to spot malicious activity originating from inside the enterprise network.”

After all, there is still a lack of that kind of scanning over filesystem to find out any illegal alterations. This kind of audit is mostly the successful way to catch unpleasant activities originating from inside and outside. Whilst, windows servers are not really supported with many security features as many as what is there in

Linux servers. One of the most important is the server monitoring. In order to achieve better security, servers require monitoring in many aspects, one of which is foremost is the filesystem.

1.3 Problem Statement

Online security concerns grow day after day as new viruses and worms are released. Because of this, it is now more important than ever to monitor the server's filesystem for signs of compromise. In order to accomplish that, this project will answer the following questions; How to monitor the filesystem of Windows server operating system? What are the critical files that it should monitor? In what way and when does the server launch the alerts?

1.4 Project Objectives

The objectives of this project are intended to:

1. Investigate a method of server monitoring on Windows server provided by Microsoft.
2. Design and develop a prototype of server monitoring that enhances the investigated method from time perspective.
3. Test the prototype and compare the results to any available Windows server monitoring system.

1.5 Project Aim

The aim of the project is to develop the monitoring of filesystem integrity in Windows server. As well as, is to minimize the total time that the filesystem integrity checker takes in the checking process in order to avoid opportunity gab weakness.

1.6 Project Scope

The scope of this project is:

1. The study concentrates on Windows server 2003 only.
2. The process of the monitoring only focuses on configuration files in the filesystem.
3. The proposed enhancement will concentrate only on accelerating the checking process of the filesystem integrity checker.

1.7 Thesis Organization

The report is divided into 5 chapters.

1. Chapter 1 describes the introduction and background of the study, the project objectives, scope, and the layout of the report.
2. Chapter 2 gives literature review on information that relate to this research.
3. Chapter 3 demonstrates the project methodology.
4. Chapter 4 represents the design and implementation of this prototype.
5. Chapter 5 shows the results of this research and finding.
6. Chapter 6 is the conclusion and summary.

REFERENCES

1. Apvrille, A, Gordon, D, Hallyn, S, Pourzandi, M., and Roy, V. DigSig: Runtime authentication of binaries at kernel level. *In Proceedings of the 18th USENIX Large Installation System Administration Conference*. (LISA 2004), November 2004. Pre-print.
2. Aranya, A., Wright, C., and Zadok, E. (2004). Tracefs: A File System to Trace Them All. *Proceedings of the Third USENIX Conference on File and Storage Technologies.* USENLX.
3. Bace, R, and Mell, P. *Intrusion Detection Systems.* NIST Special Publications SP 800-31, 2001.
4. Beattie, S, Black, A, Cowan, C, et al. (2000). CryptoMark: Locking the Stable door ahead of the Trojan Horse. Technical report.
5. Berndtsson, M. and Hansson, J. (2008). *Thesis Projects: A Guide for Students in Computer Science and Information Systems.* (2ndedition.) London: Springer-Verlag.
6. Blaze, M. (1993). A cryptographic file system for Unix. In 1st ACM Conference on Communications and Computing Security. ACM.
7. Brenner, B. SearchSecurity Information Resource. *Report: Attackers Socking it to Servers, Websites.* Retrieved April 15, 2009. From: http://searchsecurity.techtarget.com/news/article/0,289142,sid14_gci1083102,00.html
8. Catuogno, L., and Visconti, I. (2002). A Format-Independent Architecture for Run-Time Integrity Checking of Executable Code. *In Proceedings of the Third Conference on Security in Communication Networks.* Dipartimento di Informatica ed Applicazioni, Universita di Salerno.

9. Christensen, S, Sorensen, K, and Thrysoe, M. (2004). *Umbrella: We can't prevent the rain... - But we don't get wet!* Master's thesis, Aalborg University.

10. Craig, W and McNeal, P. (2003). RadMind: The Integration of Filesystem Integrity Checking with File System Management. *In Proceedings of 17th* USENIX Large Installation System Administration Conference. LISA.

11. Danseglio, M. *Securing Windows Server 2003*. (1st edition). Gravenstein: O'Reilly Media. 2004

12. Daugherty, M. (2004). *Monitoring and Managing Microsoft Exchange Server 2003.* Hewlett-Packard Development Company.

13. DISA for the DOD: DISA Field Security Operation. *Web Server: Security Technical Implementation Guide.* (Dec 2006).

14. Dunlap, J. K. and Karels, J. M. *Name Server Operation Guide for BIND.* Release 4.9.4. Berkeley. Retrieved March 20, 2009. From: http://www.dns.net/dnsrd/docs/bog/bog.html.

15. Filesystems: *Information on File Systems.* Retrieved March 18, 2009. From: http://filesystems.palconit.com/

16. Forristal, J, Schuh, J and Shipley, G. Network Computing. *Think Like an Attacker.* Retrieved March 18, 2009. From: http://www.networkcomputing.com/showArticle.jhtml?articleID=163105313

17. Hayday, G. (2003). *ZDNet Tech News, Blogs and White papers for IT Professional.* Exposed Server: Magnet for Hack Attackers. Retrieved April 15, 2009. From http://news.zdnet.com/2100-1009_22-127291.html

18. Jones, D. and Rouse, M. *Microsoft Windows Server 2003 Delta Guide.* (2st edition). New Jersey: SAMS. 2003

19. Kaczmarek, J and Wróbel, M. (2008). Modern Approach to File System Integrity Checking. *Proceedings of the 2008 1st International Conference on Information Technology.* Gdansk Poland: IEEE press.

20. Kim, G. and E. Spafford. (1994). Experiences with Tripwire: Using Integrity Checkers for Intrusion Detection. *In the Proceedings of the Usenix System Administration, Networking and Security*. SANS III.

21. Krebs, B. *The Washington Post: Security Fix.* Hundreds of Thousands of Microsoft Web Servers Hacked. Retrieved April 15, 2009. From:

http://voices.washingtonpost.com/securityfix/2008/04/hundreds_of_thousands_of_micro_1.html

22. Lettice, J. (2004). *Security Report: Windows vs. Linux.* Retrieved March 18, 2009. From: http://regmedia.co.uk/2004/10/22/security_report_windows_vs_linux.pdf

23. Malaysian Computer Emergency Response Team. *MyCERT Statistics from 2006-09.* Retrieved April 15, 2009. From: http://www.mycert.org.my/en/services/statistic/mycert/2009/main/detail/625/inde``x.html

24. Microsoft Help and Support. *Limitations of the FAT32 File System in Windows XP*. Retrieved March 19, 2009. From: http://support.microsoft.com/kb/314463

25. Microsoft Tech Net. *Windows Server 2003 Operating System.* Retrieved April 15, 2009. From http://technet.microsoft.com/en-us/windowsserver/bb429524.aspx

26. Pegley, N. TechiWarehouse. *Types of Servers.* Retrieved March 18, 2009. From: http://www.techiwarehouse.com/cms/engine.php?page_id=fb1f7b2a

27. Radack, S. (2009). *The Cryptographic Hash Algorithm Family: Revision of the Secure Hash Standard and Ongoing Competition for New Hash Algorithm.* Retrieved from: http://csrc.nist.gov/publications/nistbul/March2009_cryptographic-hash-algorithm-family.pdf

28. Mosley, D. (2006). *Implementation of a File Integrity Check System.* East Carolina University.

29. Motara, Y and Irwin, B. (2005). *Information Security South Africa (ISSA).* File Integrity Checkers: State of the Art and Best Practices. Sandton Conference Centre: Johannesburg. July 2005.

30. Nowicki, B. (1094). *NFS: Network File System Protocol Specification*. Technical report, Sun Microsystems, Inc., March 1989.

31. Parmjit, S. et al. (2006). *A comprehensive guide to writing a research Proposal.*

32. Patil, S., Kashyap, A, Sivathanu., G and Zadok, E. (2004). *In the Proceedings of the 18th USENIX Large Installation System Administration Conference*.

I3FS: An In-Kernel Integrity Checker and Intrusion Detection File System. Atlanta, GA: LISA.

33. Samhain Labs. *Samhain: File System Integrity Checker.* Retrieved March 21, 2009. From: http://samhain.sourceforge.net.

34. Sandberg, R., Coldberg, D., Kleiman, S., Walsh, D., and Lyon, B. (1986). *Design and implementation of the sun network filesystem.* USENIX.

35. Serafim, V, and Weber, R. *The SOFFIC Project*. Technical report, UFRGS Security Group - GSeg. Universidade Federal do Rio Grande do Sul (UFRGS), Porto Alegre, RS, Brazil, 2002.

36. Shepler, S, Callaghan, B, Robinson, D et al. *RFC 3010 - NFS version 4 Protocol*. Technical report, Sun Microsystems, Inc.; Hummingbird Ltd.; Zambeel, Inc.; Network Appliance, Inc., December 2000.

37. Skoudis, E and Liston, T. (2006). *Counter Hack Reloaded: A Step-by-step Guide to Computer Attacks and Effective Defence.* (2nd edition.) New Jersey: Prentice Hall PTR.

38. Stein, C., Howard, J., and Seltzer, M. (2001). Unifying file system protection. *In proceedings of the general track: 2002 USENIX Annual Technical Conference*. Berkeley, USENIX Association.

39. Tiensivo, A. *Securing Windows Server 2008: Prevent Attacks from Outside and Inside your Organization.* (1st edition). Burlington: Syngress. 2008

40. The Computer Technology Documentation Project: *Windows 2000 File Systems.* Retrieved March 19, 2009. From: http://www.comptechdoc.org/os/windows/win2k/win2kfiles.html

41. Tomonori, F and Masanori, O. (2003). Protecting the integrity of an entire file system. *Proceedings of the First IEEE International Workshop on Information Assurance.* IEEE press.

42. Tripwire, Inc. Tripwire for Servers Datasheet. Technical report, Tripwire, Inc, 2005.

43. Wang, X., Yin, Y and Yu, H. (2005). *In Advances in Cryptology*. Finding collisions in the full SHA-1. Vol3621 of Lecture Notes in Computer Science. Springer-Verlag, Berlin, Germany, 2005.

44. Williams, M. (2002). *Anti-Trojan and Trojan Detection with In-Kernel Digital Signature testing of Executables.* Technical report, Security Software Engineering: NetXSecure NZ Limited.

45. Williams, R. and Walla, M. *The Ultimate Windows Server 2003 System's Administrator Guide.* (1st edition). Boston: Addison-Wesley Professional. 2004

46. W3schools web developer site. *Web Statistics and Trends: OS Platform Statistics.* Retrieved April 15, 2009. From: http://www.w3schools.com/browsers/browsers_os.asp