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.

Budget File and System Integrity Verification for Windows

Home users need an additional level of protection because the threats have increased and file and system integrity verification is able provide this. This paper defines file and system integrity verification; the threats to home users; how integrity verification works; cryptographic and non-cryptographic integrity verification algorithms; properties of good integrity verification algorithms; attacks against the integrity verification process; freeware windows tools; the future and the value of integrity verification to...

Copyright SANS Institute
Author Retains Full Rights

AD

Budget File and System Integrity Verification for Windows

© SANS Institute 2004, Author retains full rights.

GIAC Security Essentials Certification (GSEC)
SANS Practical assignment Version 1.4b – Option 1
Submitted by: Ditmar den Engelsen
Date: January 26, 2004

© SANS Institute 2004, As part of the Information Security Reading Room Author retains full rights.

Abstract

Home users need an additional level of protection because the threats have increased and file and system integrity verification is able provide this.

This paper defines file and system integrity verification; the threats to home users; how integrity verification works; cryptographic and non-cryptographic integrity verification algorithms; properties of good integrity verification algorithms; attacks against the integrity verification process; freeware windows tools; the future and the value of integrity verification tools to add protection to the home user.

Integrity verification clearly has benefits in preventing, detecting and recovering from attacks for the security conscious home user but it has a long way to go to deliver similar benefits to the average home user. Integrity verification doesn't depend on virus/attack signatures but verification of downloaded files still is very cumbersome and with the current tools it is not possible to automatically this. Automation and Operating system integration is required and upcoming.

© SANS Institute 2004, Author retains full rights.

© SANS Institute 2004, As part of the Information Security Reading Room Author retains full rights.

Table of Contents

1. Introduction 4
2. Definition Integrity verification 4
3. Threats to Home Users 5
4. Integrity verification value 6
5. Integrity verification process description 8
6. Integrity verification types 9
7. Algorithm requirements 12
8. Attacks 14
9. Integrity Verification Tools 16
10. Conclusion 24
11. Future 25
12. References 27
13. Appendix A: Integrity Verification Tools 30

© SANS Institute 2004, Author retains full rights.

© SANS Institute 2004, As part of the Information Security Reading Room Author retains full rights.

1. Introduction

Home users need an additional level of protection because the threats have increased because users are more and more using “always on” internet connections and the introduction rate of new viruses and worms is increasing rapidly.

Integrity verification can provide additional protection to home users. The target audience of this paper is the security conscious Windows home that needs this protection (above anti-virus software and a personal firewall) but cannot afford to pay the license fees of commercial integrity verification tools like for example Tripwire.

This paper covers the following topics:

- What is file and system integrity verification?
- The threats to home users.
- Why integrity verification tools are useful.
- How integrity verification works.
- Two types of integrity verification algorithms and how a non cryptographic algorithm can easily be defeated.
- The properties of good (cryptographic) integrity verification algorithms that cannot be easily defeated.
- What attacks are possible against the integrity verification process?
- Tools for file and system verification on the Windows platform that are freeware for private and non-commercial use
- The future of integrity verification
- The value of integrity verification tools to add protection to the home user.

List of references and the location and cryptographic checksums of the tools described are listed at the end of this paper.

2. Definition Integrity verification

A. Menezes, P. van Oorschot and S. Vanstone describe data integrity as the property whereby data has not been changed in an unauthorized manner since the time it was created, transmitted or stored by an authorized source. [1]

File integrity verification verifies data integrity for a single file. This means verification that the file has not been changed in an unauthorized manner. System Integrity verification aims to do this on a system level. In order to accomplish this, a database of cryptographic checksums or hash codes of the files on the system is created and the system is periodically compared against this database.

Several tools exist to automate the integrity verification process for both file and system integrity but the first and most famous is Tripwire. Tripwire was developed

© SANS Institute 2004, As part of the Information Security Reading Room Author retains full rights.

in 1992 by Dr. Eugene H. Spafford and Gene H. Kim and is available for both Unix and Windows. Tripwire is a commercial software package and is too expensive for the average Windows home user. Tripwire Academic Source Release for Linux is available free of charge.

Kim and Spafford state that the goal of integrity verification tools such as Tripwire is to detect and notify system administrators of changed, added or deleted files in a meaningful and useful manner. [2]

Integrity verification makes most sense for files that are relatively stable. For example executables are very good targets because, not considering software updates, these files should not change. Temporary-, log-, or user files are not very suitable for integrity verification because of their transient nature.

3. Threats to Home Users

The most important threat to home users comes from hackers of the internet. Hackers are individuals who break into computers. A common misconception is that hackers only target major organizations. This is not true; hackers also break in to home computers.

Hackers break in to home computers for several reasons;
- find information they can use for their own profit. For example credit card numbers;
- use computer resources. For example the hard disk to store mp3 files or illegal copied software;
- use the computer to launch attacks against other computers;
- hack the computer just for fun.

The CERT/CC[1] identified the following most common methods used by hackers to gain control of home computers [3]:

1. Trojan Horse Programs
2. Back door and remote administration programs
3. Denial of Service
4. Being an intermediary for another attack
5. Unprotected Windows shares
6. Mobile code (Java, JavaScript and ActiveX)
7. Cross-site scripting
8. Email spoofing
9. Email-born viruses
10. Hidden file extensions
11. Chat clients
12. Packet sniffing

[1] The CERT Coordination Center (CERT/CC) was formed by the US Government Defense Advanced Research Projects Agency (DARPA) in November 1988 in response to the needs identified during an Internet security incident. Their charter is to work with the Internet community in detecting and resolving computer security incidents as well as taking steps to prevent future incidents.

© SANS Institute 2004, As part of the Information Security Reading Room Author retains full rights.

The most important preventive actions a home user can take to protect their systems are according to CERT/CC [4]:

1. Install and use anti-virus software
2. Keep the system patched
3. Do not trust and execute email attachments unless you know they are safe
4. Install and use a firewall program
5. Make backups of important files and folders
6. Use care when downloading and installing programs
7. Use hard to guess passwords.

4. Integrity verification value

The security conscious home user should use integrity verification software in addition to the CERT/CC preventive actions to get an additional level of protection against attacks. Integrity verification can help to prevent attacks and integrity verification helps to detect and recover from attacks.

<u>(A) Prevent attacks</u>

Integrity verification can be useful to reduce the risk (prevent) of attacks. Verification of all downloaded software before installation reduces the risk of installing trojaned software and/or backdoor and remote administration programs. The software should be verified against a cryptographic checksum or fingerprint provided by the software vendor or a trusted reference database. This, however does not protect against an author with malicious intent. It's also hard to find a trusted source for the cryptographic checksums.

SANS mentions integrity verification as countermeasure against a trend to trojan software distribution sites: "To defend against this attack vector, make sure you check the integrity of the packages you download. Whenever I upgrade a software tool across the internet, I always download copies from at least three different mirrors. I then verify the checksum of the programs from each mirror to make sure they all match The idea here is that an attacker would have a more difficult time compromising all mirrors of the code" [5]

Verification of sendmail software would have prevented the installation of a trojaned sendmail daemon. On October 08, 2002 CERT/CC issued a warning that some copies of the source code of sendmail were modified to include a Trojan horse, but because the distribution was cryptographically signed, the Trojan copy would not pass integrity verification with Pretty Good Privacy (PGP). For easy integrity verification CERT/CC included MD5 hashes of the correct versions in the advisory. CERT/CC recommends users whenever possible to verify the integrity of downloaded software. [6]

© SANS Institute 2004, As part of the Information Security Reading Room Author retains full rights.

Debian advised developers to watch checksums after a security breach on November 24, 2003 on the servers that hosted the Debian Linux sources. [8] Integrity verification will also in this case show if files have been altered.

(B) Detect attacks

Integrity verification tools are especially good in detecting attacks because in virtually all attacks traces are left on the attacked systems. If for example, an attacker installs a root kit or remote administration program, he adds a number of files to the system and probably changes a few system files to avoid detection. Integrity verification tools will report the added and changed files.

Integrity verification tools are a big problem for hackers according to a hacker called Halflife: “Even the most incompetent system administration staff have begun doing checksums on their binaries. For the hacking community, this is a major problem since their very clever Trojan program are quickly detected and removed” [9]

Integrity verification prevented that a serious backdoor was inserted in the official release of the Linux 2.6 kernel. Integrity verification showed that someone had manually changed a kernel source code file that’s normally only changed by an automated process. Careful inspection of the source code showed that someone created a backdoor that looked like a normal error-checking routine. "If it had gotten out, it could have been really bad, because any Linux kernel that had this in it, anybody who had access to that machine could become root," says McVoy.[10]

Viruses can also be detected using integrity verification tools because viruses spread by attaching themselves to (read: change) executable files[2].

(C) Recover from attacks

The use of integrity verification tools makes it much easier to recover from an attack because it’s very clear which files are changed / added / removed especially when they are used in combination with a good backup solution.

Other applications

Integrity verification can also detect unintentional modifications for example transmission errors during file transfers. Creation of Reference Libraries is another application of integrity verification. According to the Automatiserings Gids [11] the Dutch police developed a reference library with hash codes of child porn images to reduce the number of images that police officers have to view during an investigation. During a meeting of the court in Almelo the first suspect has been convicted with help of software using this library.

[2] Detecting viruses that are attached to documents (e.g. macro viruses) will be a bit more difficult because documents are more likely to change as result of normal processing.

© SANS Institute 2004, As part of the Information Security Reading Room Author retains full rights.

The US Government National Institute of Standards and Technology (NIST) has developed a National Software Reference Library [12] of known software to be used by law enforcement organizations in computer forensics investigation. By eliminating known files this reference library makes investigation of seized computer systems much more efficient.

The Recording Industry Association of America (RIAA) also uses a reference library this time with mp3 hash-values. [13] With this reference library the RIAA attempts to prove that for example a user has downloaded files from the internet instead of copying them of a CD and 'accidentally' sharing them with the internet community.

5. Integrity verification process description

The Integrity verification process consists of the sub processes checksum generation and integrity verification.

Checksum generation

The checksum generation process is responsible for creating the checksum and storing it for future reference. This is done in two steps:

1. The input file is reduced to a small string called a checksum
2. The checksum is stored in a database for future reference

Figure 1: Checksum generation process

Integrity verification

The integrity verification process is responsible for verification of the integrity of an input file with a previously generated checksum. This is also done in two steps:

1. A checksum is calculated from the new input file. Same as step 1 of the checksum generation process.
2. The checksum stored in the database is compared with checksum generated in step 1. If the two checksums match then the file has not changed. If the two checksums do not match then the process returns a verification error.

© SANS Institute 2004, As part of the Information Security Reading Room Author retains full rights.

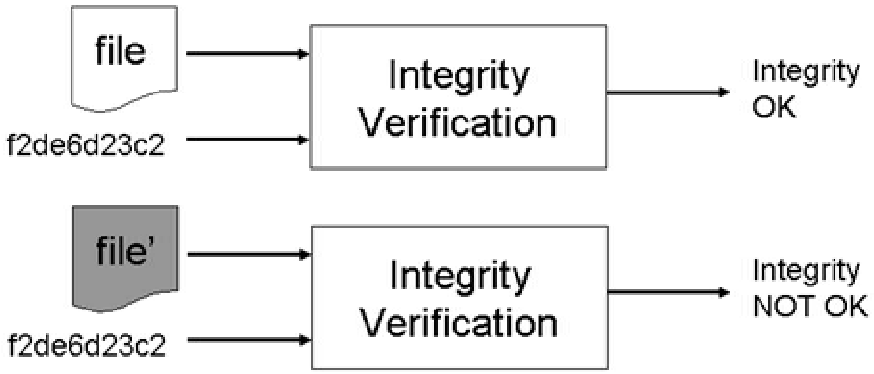

Figure 2: Integrity Verification process

6. Integrity verification types

Two integrity verification types exist: non cryptographic and cryptographic. The table below summarizes each type with example algorithms and the level of protection.

Type	Algorithms	Level of Protection	
		Unintentional Modification	Intentional Modification
Non cryptographic	CRC-16, CRC-32	YES	NO
Cryptographic	MD4, MD5, SHA-1, RIPEMD-160, HMAC	YES	YES

Table 1: Integrity verification types

<u>Non cryptographic integrity verification</u>
Non cryptographic integrity verification is good for detecting unintentional modifications. However, it does not protect against intentional modifications and is thus not useful to protect against attacks.

<u>Cyclic Redundancy Code (CRC)</u>
CRC is a simple algorithm that uses repetitive cycles to produce the output checksum. Examples of non cryptographic algorithms are CRC-16 which computes a 16-bit checksum and CRC-32 with a 32-bit checksum output. The CRC algorithm is for example used in the popular compression utility WinZip [14]

Gene H. Kim and Eugene H. Spafford do not consider CRC checksums suitable for integrity checking: “Originally intended for hardware-based error-detection, CRC functions were designed to detect multiple bit errors in data streams. Reversing the CRC function to yield a desired signature is a well-understood process, and tools to assist a potential intruder are widely available” [2]

© SANS Institute 2004, As part of the Information Security Reading Room Author retains full rights.

The text below from the Dutch national anthem “Het Wilhelmus” shows that it is possible to insert the word *altered* into the text without changing the CRC-16 of the file[3].

Original text:	Altered text:
Wilhelmus van Nassouwe	Wilhelmus van Nassouwe
ben ik, van Duitsen bloed,	ben ik, van Duitsen bloed,
den vaderland getrouwe	den vaderland getrouwe
blijf ik tot in den dood.	blijf ik tot in den dood.
Een Prinse van Oranje	Een Prinse van Oranje
ben ik, vrij onverveerd,	ben ik, vrij onverveerd,
den Koning van Hispanje	den Koning van Hispanje
heb ik altijd geëerd.	heb ik altijd geëerd.
Mijn schild ende betrouwen	Mijn schild ende betrouwen
zijt Gij, o God mijn Heer,	zijt Gij, o God mijn Heer,
op U zo wil ik bouwen,	op U zo wil ik bouwen,
Verlaat mij nimmermeer.	Verlaat mij nimmermeer.
Dat ik doch vroom mag blijven,	Dat ik doch vroom mag blijven,
uw dienaar t'aller stond,	uw dienaar t'aller stond,
de tirannie verdrijven	de tirannie verdrijven
die mij mijn hart doorwondt.	die mij *altered*broorwondt.
Filename: wilhelm.txt	Filename: wilhelm2.txt
Date: 17-11-2003	Date: 17-11-2003
Time: 17:03	Time: 17:03
Size: 418	Size: 418
CRC-16: 0x1537	CRC-16: 0x1537

To really protect against intentional modification cryptographic integrity verification must be used.

<u>Cryptographic integrity verification</u>
Cryptographic integrity verification is good in both detecting unintentional and intentional modifications.

A cryptographic checksum, also known as a hash-value, serves as compact representative image of an input string [e.g. file], and can be used as if it were uniquely identifiable with that string. [4]

Two classes of cryptographic algorithms exist and both can be used for integrity verification:

[3] The altered text was created with a tool called provecrc from Chuck Gilmore downloaded from ftp.simtel.net and verified with online crc-16 calculator http://www.digsys.se/js_crc.html (January 14, 2004). In our case only the CRC-16 matched but not the CRC-8 or CRC-32.

© SANS Institute 2004, As part of the Information Security Reading Room Author retains full rights.

1. Unkeyed cryptographic algorithms take only the message as input. Unkeyed checksums are suitable if the origin and medium can be trusted. An example of this will be a checksum database burned on cd that has been stored in a safe.
2. Keyed cryptographic algorithms take two inputs; a message and a secret key. The key is used for origin verification because one must know the key to produce the keyed checksum value. With keyed cryptographic checksums it's possible to leave the database on a hard disk because without the knowledge of the key it's infeasible to change the cryptographic checksum without detection. Keyed cryptographic checksums are also known as Message Authentication Codes (MAC's)

Unkeyed cryptographic algorithms
The most widely used unkeyed cryptographic algorithms are: Message Digest, Secure Hash Algorithm and RIPEMD.

Message Digest Series
The MD2, MD4, MD5 algorithms are used to create unkeyed cryptographic checksums and have been created by Ron Rivest. All three algorithms create a 128 bits hash-value from an input message with an arbitrary length. MD2[4], developed in 1989, is the oldest of the three algorithms and is designed for 8 bits computers. It's still considered to be secure but it's also extremely slow.

MD4, MD5[5] have both been designed for 32 bits computers. MD5 improved and more secure version of MD4. MD4 was designed too much for speed and is now considered broken. [15] MD5 adds an additional substitution and transformation round. This enhances security but it makes the algorithms also slower. MD5 is the most used algorithm for creating cryptographic checksums. Because of the limited length (128 bits) of the cryptographic checksum, leading cryptographers are now declaring that future attacks on MD5 will have a good change of success! [16] Van Oorschot and Wiener estimate that a machine specifically designed for MD5 (costing $10 million in 1994[6]) could find two files with the same MD5 checksum in 24 days on average." [17]

Secure Hash Algorithm
The Secure Hash Algorithm (SHA)[7] has been developed by the National Institute of Standards and Technology (NIST) and the National Security Agency (NSA). The algorithm produces a 160 bits cryptographic checksum and is based on MD4. The original version called SHA has been replaced in 1995 by SHA-1 because of a weak point discovered by the NSA. Details of this weak point have

[4] See RFC 1319 for more information on MD2
[5] See RFC 1320 for more information on MD4 and RFC 1321 for more information on MD5.
[6] According to Moore's law this cost is expected to halve every 18 months. This means that in the year 2003 the cost of this machine is only little bit more than $156.000,-
[7] See the Federal Information Processing Standards Publication 180-1 for more information on the Secure Hash Algorithm

© SANS Institute 2004, As part of the Information Security Reading Room Author retains full rights.

never been released to the public but the SHA-1 algorithm is considered more secure as MD5. [21] The machine described by Van Oorschot and Wiener that could break the MD5 algorithm in 24 day's would need more than 4000 years to break the SHA-1 algorithm. And if required a modified version of the algorithm: SHA-2 generates even longer (256, 384 and 512 bit) cryptographic checksums.

RIPE Message Digest algorithm

The RIPE Message Digest algorithm (RIPEMD-160)[8] is an improved version of the RIPEMD algorithm and is developed by Hans Dobbertin, Antoon Bosselaers, and Bart Preneel. The output of the algorithm is a 160-bits cryptographic hash. The algorithm has been designed to replace the MD series of Ron Rivest because according to the authors, a 128-bit hash result does not offer sufficient protection anymore. Extensions are available for 128 bit, 256 and 320 bits. The 128-bit extension has only been provided for compatibility reasons where MD4 or MD5 were used and is not recommended for new implementations.

Keyed cryptographic algorithms

The Internet Engineering Task Force standard for creating keyed cryptographic checksums or MAC's is the HMAC algorithm.

Hashed Message Authentication Code

The Hashed Message Authentication Code (HMAC) algorithm[9] describes a mechanism for using a key with an unkeyed cryptographic algorithm. HMAC can be used with for example MD5 and SHA-1, in combination with a secret shared key. The HMAC is as strong as the unkeyed cryptographic algorithm that has been used. This means that the resulting MAC's will be stronger when the HMAC algorithm is used in combination with SHA-1 than when used with MD5.

7. Algorithm requirements

Algorithms used to generate cryptographic checksums or hash-values suitable for integrity verification should have following properties[10]:

1. Compression
2. Ease of computation
3. One-wayness
4. Local one-wayness
5. Weak Collision resistance
6. Near-collision resistance
7. Non correlation
8. Avalanche

1. *Compression.* Regardless whether the input file is 1, 10 or 100MB the generated output checksum should be fixed and small (for example 128 or 160 bits) Compression is important for integrity verification because it saves space and makes the integrity database manageable. Without compression a

[8] See the RIPE paper for more information on RIPEMD-160.
[9] See the RFC 2104 For more information on HMAC
[10] The author has selected the properties that are most suitable for integrity verification. For a complete list see Menezes, van Oorschot and Vanstone's Handbook of Applied Cryptography [1]

© SANS Institute 2004, As part of the Information Security Reading Room Author retains full rights.

duplicate copy of each file would have to be stored. This would result in huge storage requirements for integrity verification.

2. *Ease of computation.* A cryptographic checksum or hash-value should be easy to compute from an input file. Ease of computation is closely related to speed and the speed of the cryptographic checksum algorithm is extremely important in integrity verification. For the average home user having to wait 2 hours before an integrity database has been created is not acceptable but waiting 10 minutes is much more acceptable.

   Klaus Bosau has compared the speeds of several checksum algorithms on a Pentium 200. [16] The table below summarizes the results. Today's systems are much faster but the table has still value in comparing the relative speeds.

Algorithm	Throughput in MB/s	Relative Speed (MD2=1)
CRC-16	16.2	54
CRC-32	9.3	31
MD4	14.4	48
Haval	10.7	36
MD5	7.2	24
SHA	5.4	18
Snefru	1.4	5
MD2	0.3	1

Table 2: Relative speed of checksum algorithms

3. *One-wayness.* For a particular checksum (for example: f2de6d23c2) it should be infeasible to find input file that was used to create the checksum or uncompress the checksum to get the input file. One-wayness makes handling of the integrity verification database easier because the resulting checksums do not have to be kept as confidential as the files that were used as input. One-wayness is also known as preimage resistant.

4. *Local one-wayness.* Local one-wayness means that is should also be infeasible to uncompress the checksum to get a piece of the input file that was used to create the cryptographic checksum. This also makes handling of the database with cryptographic checksum easier. Local one-wayness is also known as partial-preimage resistance

5. *Weak collision resistance.* For a particular file is should also be infeasible to find a second file that has the same cryptographic checksum. This means that in the CRC-example finding wilhelm2.txt with the same checksum as wilhelm.txt should not be possible for a cryptographic function. If cryptographic checksums were not weak collision resistant an attacker could easily replace files with altered, possible trojaned files that would have the

© SANS Institute 2004, As part of the Information Security Reading Room Author retains full rights.

same cryptographic checksum and the altered files would pass the integrity verification process. Weak collision resistant is also known as 2nd-preimage resistant.

6. *Near-collision resistance.* Near-collision resistant means that it should be infeasible to find two different files for which the cryptographic checksum is only slightly different. This makes hiding file changes not possible and makes comparing the cryptographic checksums from your computer screen possible. On screen verification is important when for example you suspect the integrity verification tool has been attacked. (for more information on attacks see the next section)

7. *Non-correlation.* It should not be possible to correlate the bits of the cryptographic checksum to parts of the input file used to create the checksum. This means that the first digit of the checksum in the CRC-example should not be computed from the first part of the anthem. Non-correlation makes it harder for an attacker to change the file while maintaining the cryptographic checksum.

8. *Avalanche.* The avalanche property means that every input bit should affects every output bit thus that even a slight change in an input string should cause the cryptographic checksum to change drastically. This also hiding file changes infeasible for an attacker.

8. Attacks

A hacker has several options to circumvent the protection of integrity verification tools. As listed in the following table the hacker can for example attack the algorithm, software, operating system or the database with cryptographic checksums.

Attack type	Attack name
Cryptographic algorithm	Brute force
Cryptographic algorithm	Birthday
Software	Integrity verification software hack
Operating System	Kernel hack
Database alteration	Database

Table 3: Attack types

<u>Brute Force search</u>

A brute force search means, for a modified file, trying to make additional changes to that file to get back the original checksum. If there are no shortcuts this means the cryptographic algorithm is not flawed and that on average half of all possible options need to be investigated. For a 128-bits MD5 hash this means searching

© SANS Institute 2004, As part of the Information Security Reading Room Author retains full rights.

2^{128-1} possible options. With today's computers this is not feasible. Therefore a brute force search is strictly spoken not an attack. Mitigation against a brute force search is choosing large (at least 128-bits) keys.

Birthday attack

A birthday attack is also an attack against the cryptographic algorithm. The goal of this attack is to find two files that have the same checksum. This is much easier as finding an input file from a specific cryptographic checksum. For an MD5 hash finding two files with the same cryptographic checksum only takes $2^{128/2}=2^{64}$. With many modern computers in parallel the required (2^{64}) calculations for a birthday attack seem feasible. Luckily in practice an attacker cannot mount a birthday attack against the integrity verification algorithm because an attacker cannot choose arbitrary files, the cryptographic checksums have already been generated and stored in the integrity verification database. Mitigation against a birthday attack is choosing large checksum keys (160-bit SHA-1 values rather than 128-bit MD5 values) and creating several cryptographic checksums for the same file (SHA-1 plus MD5)

Integrity verification software hack

An attacker can also attack the integrity verification software itself. For example the integrity verification engine could be modified so that it would verify the integrity of file even if the cryptographic checksum doesn't match. The software vendor could mitigate the risk of this type of attacks by protecting the code from modification by for example internal verification of the code at execution time. This risk could be eliminated by mounting the disks with the files that have to be verified and running the integrity verification software from another clean OS.

Kernel Hack

Instead of attacking the integrity verification software the kernel could also be hacked in order to give wrong information about the altered files. Halflife: "Integrity checking programs need to trust the operating system [..] In code that is designed to detect compromises […] you can not trust anything, including your own operating system." [9] The software vendor could mitigate this risk by protecting the kernel or operating system from modification. This risk could be eliminated by mounting the disks with the files that have to be verified and running the integrity verification software from another clean OS.

Database

Altering the database with cryptographic checksums is possibly the easiest option if this database is not correctly protected. To hide an attack, all the attacker has to do is to replace the original checksums with the checksums of the altered files. Therefore it is essential to protect the database with cryptographic checksums by for example storing it on read-only and/or removable media for example CD, USB-stick. If this is not feasible at least add another level of integrity protection by creating a cryptographic checksum of the database and/or encrypting this database. The first protective action will prevent the attacker from

© SANS Institute 2004, As part of the Information Security Reading Room Author retains full rights.

altering the database. The second protective action will detect the alteration of the database. Another advantage of the first protective action is that the database will help recovering from the attack when used together with a restore because it will be clear which files have been changed and how the attacker has broken into the system.

9. Integrity Verification Tools

A number of integrity verification solutions exist from tools that can verify the integrity of a single file to automated system integrity verification programs that are able to verify a complete network of systems.

The tools mentioned here are all freeware for private and non-commercial use and use unkeyed cryptographic algorithms. Unkeyed algorithms are used because these algorithms have the advantage that they can be automated. Tools based on keyed algorithms would require manual intervention for secure entry of the key. Examples for tools using keyed cryptographic algorithms to create digital signatures or MAC's are PGP and Gnu Privacy Guard (GnuPG).

The cryptographic MD5 and SHA checksums and download information of the tools described in this paragraph are in Appendix A Integrity verification tools.

<u>File verification tools</u>

File verification tools are standalone tools that used to monitor the integrity of a single or multiple files. They require manual intervention.

To demonstrate the working of file integrity verification tools a utility from Foundstone: Fport (v2.0) is downloaded from the intrusion detection section the Foundstone website[11]. See the screenshot below for the description of Fport. Also shown in the screenshot from packetstormsecurity.net[12] below is the MD5 cryptographic checksum (66c742a94e4f1f3881b0cd9d84727e4e)

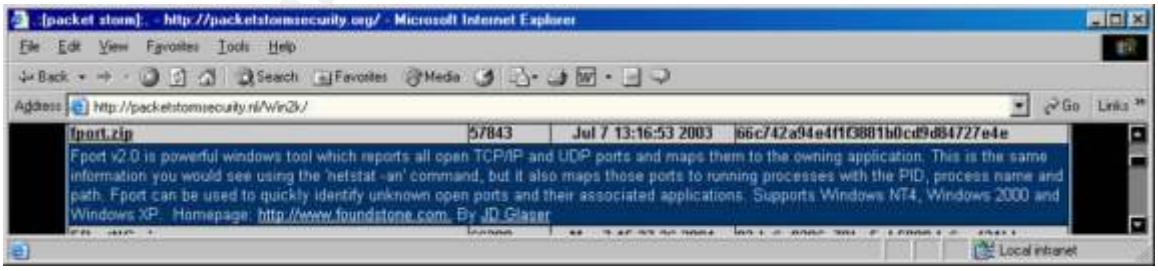

Figure 3 Fport.zip Md5 cryptographic checksum from Packetstormsecurity website

[11] Foundstone software can also be verified using keyed cryptographic checksums with PGP/GnuPG on their website but this is not in scope of this paper

[12] In order to provide additional security the cryptographic checksum has to be downloaded from (1) a trusted and (2) another site than the downloaded software.

© SANS Institute 2004, As part of the Information Security Reading Room Author retains full rights.

The tools DigestIT and SummerProperties are used to verify the integrity of the downloaded Fport utility with the cryptographic checksum published on Packetstormsecurity.

Digest IT 2004

DigestIT version 2004 integrates itself into the Windows Explorer context menu to provide a right-click to hash interface. The software is capable of computing MD5 and SHA-1 cryptographic checksums. It's possible to select multiple files and save the result to a file or copy to the clipboard. Digest IT is distributed as msi script that requires the windows installer for installation. The author of Digest IT has published the MD5, SHA-1 and a keyed cryptographic checksum (using PGP) on his website for verification of the integrity of the distribution.

Right-click fport.zip and then choose Digest IT 2004/Verify MD5 hash in order to verify the integrity of our example fport download. Enter the MD5 cryptographic checksum from packetstormsecurity.

Figure 4 Digest IT 2004 checksum verification input

A message "Digest matches. Verification succeeded" appears if the MD5 checksum of the fport.zip file matches the checksum downloaded from packetstormsecurity.

SummerProperties 1.1

SummerProperties version 1.1 works similar as Digest IT, this time another page is added to the property sheet. The program computes CRC-16, CRC-32, MD5 and SHA-1 for the selected file. It's possible to select the algorithms that the program should calculate. The program doesn't have the option to select multiple files and export the output to a file but it provides an option to improve readability by dividing the output into words.

Right-click fport.zip and then choose "Properties" in order to verify the integrity of our example fport download. As shown in the screenshot below the MD5 checksum matches the checksum downloaded from packetstormsecurity.

© SANS Institute 2004, As part of the Information Security Reading Room Author retains full rights.

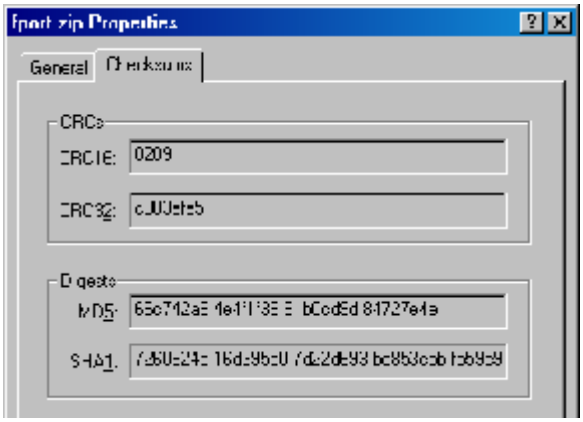

Figure 5 Checksums page SummerProperties

System Verification Tools

System verification tools aim to verify the integrity of all files on the system. Their main usage is to detect unauthorized changes to a system. These tools range from standalone tools capable of manually monitoring multiple files to system monitoring solutions capable of automated monitoring of computers.

Windows File Protection

Since Windows 2000, Microsoft has included an internal integrity protection, using SHA-1 cryptographic checksums for the most important system files; called Windows File Protection (WFP). The database called the catalog is protected with a digital signature from Microsoft. The hash-values are stored in the file %systemroot%\system32\dllcache\nt5.cat and nt5inf.cat. Microsoft has included the program sigverif.exe to verify the files manually. Only selected windows files can be verified using this method. [30]

The advantage of WFP is that it's built in Windows 2000 and XP and doesn't require an extra installation. WFP can also run from a diskette. The possibility to run the program and database from diskette when booting from a clean operating system give it superior protection against Software, Operating System and integrity database hacks. Database attacks are easy to detect because Microsoft has digitally signed the database. WFP can only be used to verify Microsoft distributed files.

Creating the database

It's not possible to create the database because Microsoft has already created the database.

Integrity verification

To start the Microsoft integrity verification tool execute 'sigverif' from start -> run or the command line. Click Advanced to save the results to a log file and click start to start the verification. WFP verified 3067 files in 1 minute and 26 seconds on a freshly installed Windows XP computer with Service Pack 1a.

© SANS Institute 2004, As part of the Information Security Reading Room Author retains full rights.

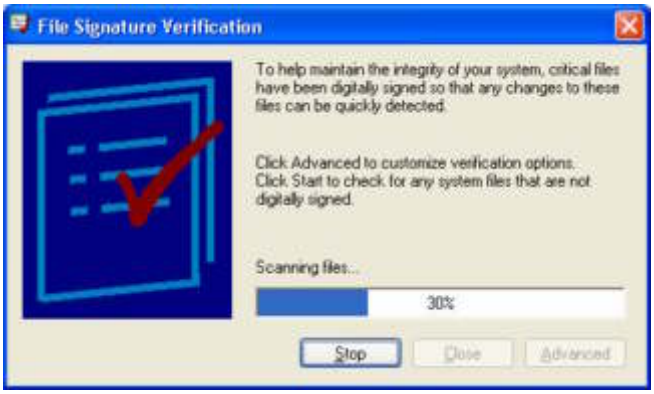

Figure 6 Integrity verification using Windows File Protection

MD5Summer

MD5summer version 1.2 is a relatively small application for Windows designed for creating and verifying MD5 cryptographic checksums. MD5Summer is able to creates and verify files created with other applications because it uses the MD5Sum file format.

The advantages of MD5Summer are that it's easy to use, doesn't need to be installed and can run from a floppy together with the signature database.

MD5Summer is an integrity verification tool designed for verification of multiple files. MD5Summer is not a system integrity monitoring solution because it cannot exclude files based on file extensions, doesn't detect files added to the system and cannot be automated.

Creating the database

Select the drive root, select all files and click add recursively to create the database for the whole volume. MDSummer starts creating MD5 checksums for all files selected. After completion review the errors (mostly created by locked files) and stored the database safely. Protect the database against alteration by writing it to a diskette or removable drive (USB-stick).

© SANS Institute 2004, As part of the Information Security Reading Room Author retains full rights.

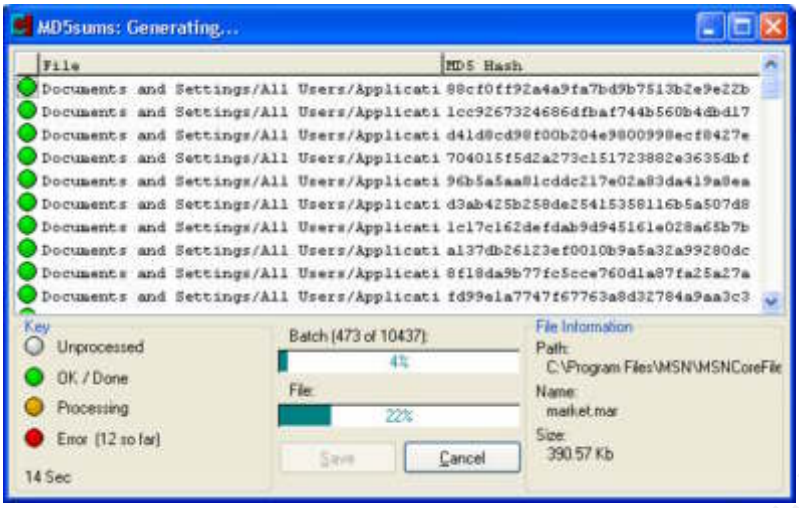

Figure 7: Generating Cryptographic Checksums with MD5Summer

Integrity verification

To protect against kernel hacks the system is booted from a PE Builder CD. This is a bootcd with a limited working Windows XP version on it. [25] The MD5Summer program and database are copied from a write protected diskette. (This because I could not get my USB drive to work in PE Builder) Before the database can be used for verification it has to be copied back to the original volume (C: in my case). As shown below only the directory information relatively from the root has been stored in the database.

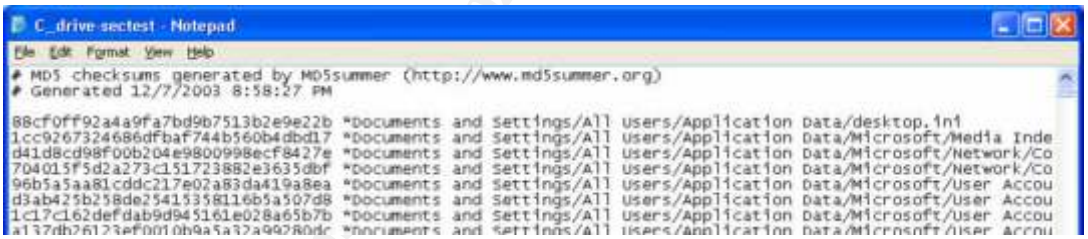

Figure 8: MD5Summer Checksum output file

Select the database file and click open to start the verification process. On my computer the verification process completes in 7 min 32 sec. Three temporary files failed integrity verification.

<u>Fsum</u>

Fsum is a integrity verification command utility from Slavasoft. Fsum is freeware and is able to check and verify multiple files and recurse directories. Fsum is able to use the following algorithms for creating checksums: MD2, MD4, MD5, SHA-1, SHA-2 (256,384,512), RIPEMD-160, PANAMA, TIGER, CRC-32, and the hash used in eDonkey (ed2k)/eMule tools

© SANS Institute 2004, As part of the Information Security Reading Room Author retains full rights.

The ability to select files based on filename and/or extension is a major advantage of Fsum because it enables us for example to select only executable files and this will reduce false positives. Integrity verification is very fast; more than 3 times faster than MD5Summer. Fsum's ability to generate multiple checksums for a single file makes it very secure. Fsum doesn't need to be installed and can run from a floppy together with the signature database.

Files added to the system are not detected because Fsum only verifies files that are present in the database.

Fsum is an integrity verification tool designed for verification of multiple files. Fsum is not a system integrity monitoring solution because it doesn't detect files added to the system and cannot be automated.

Creating the database
Use the following command to create a database C-drive-sectest-fsum.md5 with MD5 sums for all files in the current volume. Note that in my case the executable and the database are both stored on a removable USB drive (E:)

```
C:\>e:\fsum -jm -r *.* > E:\C-drive-sectest-fsum.md5
```

Fsum completes the scanning in 2 minutes and 55 seconds.

Integrity verification
The integrity of the C volume can be verified with the following command.

```
C:\>e:\fsum -c -jf e:\C-drive-sectest-fsum.md5
```

Note that it's not required to copy the database because Fsum uses the current directory as starting point and in our case this is the C-volume. Fsum completes the verification in 2 minutes and 57 seconds.

<u>GFI LANguard System Integrity Monitor</u>
GFI LANguard System Integrity Monitor (SIM) is a freeware system integrity monitoring solution for Windows from GFI. SIM uses the MD5 algorithm to create the database with cryptographic hashes.

The advantage of SIM is that the integrity verification can be fully automated and it's possible to send email to a specified address with the results of the verification. The graphical user interface (GUI) also makes it easy to add files and/or directories to a integrity verification job. The ability to exclude specific file extensions is very welcome because it reduces the number of false positives. SIM requires installation but it is possible to install the program (1.7MB) and database with cryptographic hashes (in my case 2.7MB) on a removable medium. The database can not be manually verified because all cryptographic checksums are stored in binary format.

© SANS Institute 2004, As part of the Information Security Reading Room Author retains full rights.

General configuration
The SMTP server and email address need to be specified in SIM before automated email messages with the verification result can be send. Usually the SMTP server can be obtained from your internet provider. Sometimes the “from” email address should also be the address specified from the internet provider because some SMTP servers block email from email addresses they don’t know.

Creating the database
The database is automatically created with a default scan job. All files on the system except the files with excluded file extensions are added to the default scan job. The database is called cfdata.mdb and is stored in <installation root>\GFI\System Integrity Monitor 3\Data. To add files to the database open the SIM configuration program and right-click the file(s) (see the screenshot below)

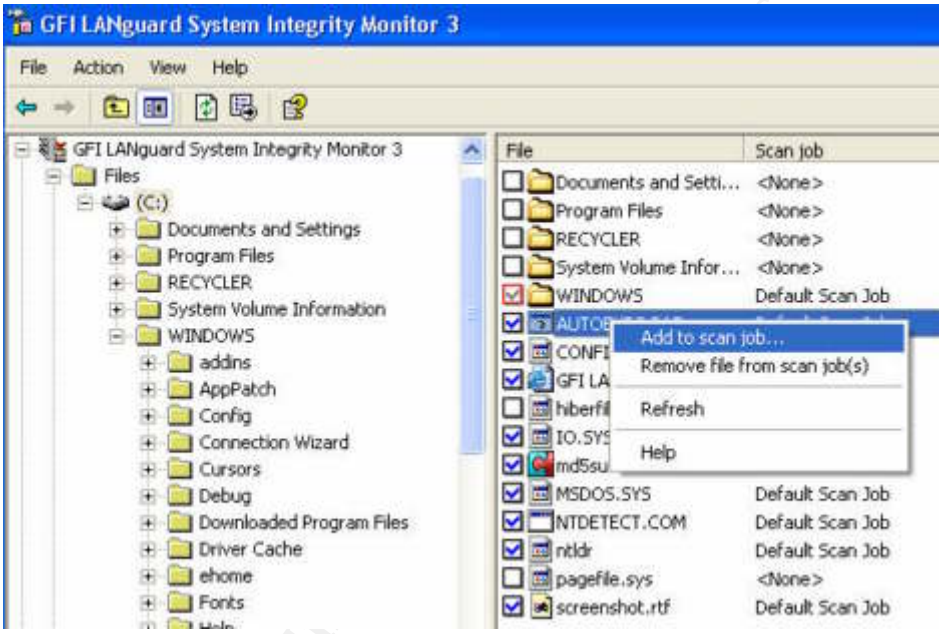

Figure 9: Adding files to scan job in GFI LANGuard SIM

Integrity verification
Integrity verification for the default scan job is by default done automatically every two hours. This however can be customized and with the “scan now” button the scanning can be executed immediately. If changes are detected this is logged to the GFI LANguard System Integrity Monitor section of the Windows event log and send by email to the address specified in the scan job.

<u>Osiris</u>
Osiris is a client-server system integrity management solution from the Shmoo group. Osiris is designed to scan multiple computers from a management computer. To do this Osiris has three parts: the scanning service and the management service and the management application. The scanning service

© SANS Institute 2004, As part of the Information Security Reading Room Author retains full rights.

should be installed on any computer that will be monitored. The management service and application should be installed on a trusted computer but most home users will install all parts on one system.

The advantages of Osiris are that it's highly automated once you've configured it, that it can use multiple cryptographic checksum algorithms (SHA, MD5, RIPEMD, haval) and publishes a hyperlink for easy acceptance of changes. The Osiris database is read-only and cannot be changed within Osiris. This however doesn't mean that it is impossible to alter the database outside of Osiris. The process in Osiris to accept changes on the computer is to swap the original database with the new database created at scanning time.

The disadvantages of Osiris are that it's harder to configure and doesn't include a GUI at the moment but Osiris is work in progress so this can change. The authors aim Osiris to include at least as much functionality as their commercial brothers (like Tripwire).

General configuration
Osiris requires a lot of configuration; this has been described in detail in the user guide. Configuration requires changing the administrator password and creating the management part initial configuration. Most of the time defaults can be accepted but to be able to see results in email it is, the same as with SIM, required to enter an email address and SMTP server.

See the management configuration below for a single windows computer

```
syslog_facility = DAEMON
syslog_level = NOTICE
log_intensity = LOW
control_port = 2266
http_port = 2267
http_host =
notify_email = <Email-address>
notify_smtp_host = <SMTP-server-address>
notify_smtp_port = 25
hosts_directory =
allow = 127.0.0.1
```

Creating the database
Before the database can be created the scanner requires configuration with the new-host command. Then Osiris asks to initialize the database and whether the default OS configuration should be used. If the defaults are accepted this creates a database with MD5 cryptographic checksums for all files in the C:\windows\system and C:\windows\system32 directory.

© SANS Institute 2004, As part of the Information Security Reading Room Author retains full rights.

See the scanner (host) configuration below for a single Windows computer

```
host                => sectest
hostname/IP address   => 127.0.0.1
description           => Security test pc
host type             => generic
log enabled           => yes
archive scans         => yes
notifications enabled => yes
notifications always  => yes
notify email          => (managment config)
scans starting on     => Mon Dec 22 23:00:00 2003
scan frequency        => every 1440 minutes
enabled               => yes
```

Integrity verification
Osiris integrity verification is an automated process with a daily (1440 minutes) frequency as specified in the scanner configuration. (see below)

```
scans starting on     => Mon Dec 22 23:00:00 2003
scan frequency        => every 1440 minutes
enabled               => yes
```

It's also possible to start the scanning manually by invoking the start-scan command. Changes and other notification are written in the application log in the event-log on the computer. If a valid email and SMTP server has been specified in the management host configuration changes are also send to an email address. This email contains a hyperlink that makes it very easy to accept the changes.

10. Conclusion

File and system integrity verification clearly has benefits in preventing, detecting and recovering from attacks for the security conscious home user but at the same time File and system integrity verification has a long way to go to deliver similar benefits to the average home user.

<u>File integrity verification</u>
File integrity verification of downloaded files still is very time consuming and with the current tools it is not possible to automatically this.

The hard part of file integrity verification (using unkeyed cryptographic hashes) is finding a trusted source for the cryptographic checksums. For security tools, other high risk tools and utilities the additional effort in searching for trusted cryptographic checksums is worth the effort because it reduces the risk of installing corrupted or trojaned versions. But file integrity verification currently is

© SANS Institute 2004, As part of the Information Security Reading Room Author retains full rights.

not useful in intrusion prevention for the average home user, it is too hard or sometimes impossible to find trusted cryptographic checksums. However this is likely to change because HP, IBM, Installshield, RSA Security, Sun and Tripwire have announced to create a trusted File Signature DataBase. (FSDB) [26]

Integrity verification would really add value to the home user if Microsoft would make it possible in their Next Generation Secure Computing Base to seamlessly and automatically verify the integrity of downloaded files against a multiple chosen trusted signature databases like for example the FSDB.

Digest IT 2004 and SummerProperties are equally good at file integrity verification. They both integrate in the context menu and can be used to verify MD5 and SHA-1 checksums.

System integrity verification

System integrity verification adds value in detecting and recovering from attacks. In order to be able to detect attacks the tool must be capable to monitor changed, removed and added files. The freeware system Integrity verification tools for the Windows platform currently lack monitoring the registry for changes. This should be added because this is the place where most configuration information is stored and unauthorized alteration of some registry setting can make the system a lot less secure.

GFI LANguard System Integrity Monitor is the best tool for System Integrity verification on a home computer. The reason for this is because it is freeware, has a GUI and a good default configuration. Osiris is the best tool when implemented in client server mode on multiple systems. MD5Summer and Fsum are not suitable for standalone system integrity verification but will certainly add value in addition to SIM and Osiris to verify that the OS can be trusted. However this requires running these tools from a Boot OS CD.

11. Future

Integrity verification will become more important but less visible. In an effort to improve, security vendors are moving from denying access to known bad files or attacks to only allowing access to actions that can be trusted. This transition and the increased importance of integrity verification can be seen in three area's; (1) The Next Generation Secure Computing Base initiative from Microsoft, (2) The file signature database from HP, IBM, Installshield, RSA Security, Sun and Tripwire and (3) The future of Antivirus

1. *Next Generation Secure Computing Base from Microsoft.* The Secure Computing Base from Microsoft will embed integrity verification in the hardware and Operating systems. This way all code running on the computer will be integrity verified automatically using cryptographic checksums and modified code will not be allowed to run on the system.

© SANS Institute 2004, As part of the Information Security Reading Room Author retains full rights.

2. *File Signature Database*[13]*.* On August 5, 2003 HP, IBM, Installshield, RSA Security, Sun and Tripwire announced to create a cross-platform File Signature Database. (FSDB) [26] The database already contains over 11 million “known good” signatures from multiple vendors and suppliers. At the moment they are talking with Microsoft and RedHat to have them join the initiative.

   IDC welcomed this initiative as shown in the following quote from Chris Christiansen, Vice President of IDC’s Security Products program:

   "We believe that the emergence of the Tripwire-driven FSDB project marks a transition from tracking 'known bad' files, such as viruses and other signature-based malicious code" and "The traditionally reactive approach to new threats ensures that customers are always one step behind the bad guys. By knowing what the 'good state' is, improper and corrupted files can be eliminated by exception before they execute their poisonous instructions."

3. *Future Antivirus.* According to Robert Vibes future antivirus software needs to incorporate heuristic scanning (also known as behavior scanners) and integrity checkers to prevent viruses from doing any harm [28]. Integrity Verification can prevent viruses if only trusted software is allowed to run on the computer and changed (infected) programs are denied access. This is required because the current antivirus technology is ending and reactive. Robert Rosenberg quotes network expert Tony Bradley in his paper [29] “The entire model of developing a signature […] will eventually become too cumbersome in my opinion anyway. [..] This method also means that the security experts and antivirus vendors are always one step behind the malicious code writers.

[13] For more information see the FSDB-FAQ

© SANS Institute 2004, As part of the Information Security Reading Room Author retains full rights.

12. References

[1] Menezes A., P. van Oorschot and S. Vanstone, *Handbook of Applied Cryptography*, CRC Press, October 1996, chapter 9, 321 – 383, URL: http://www.cacr.math.uwaterloo.ca/hac/about/chap9.pdf (January 14, 2003).

[2] Gene H. Kim and Eugene H. Spafford, "The Design and implementation of Tripwire: A file system integrity checker", Purdue University, February 23, 1995, URL: http://sunsite.bilkent.edu.tr/pub/security/cerias/papers/gene-kim/Tripwire.pdf (January 14, 2003).

[3] CERT/CC, "Home network security", Carnegie Mellon University, December 5, 2001, URL: http://www.cert.org/tech_tips/home_networks.html (January 14, 2003).

[4] CERT/CC, "Home computer security", Carnegie Mellon University, 2002, URL: http://www.cert.org/homeusers/HomeComputerSecurity (January 14, 2003).

[5] *SANS,* "SANS Workbook Track 4 – Hacker Techniques, Exploits and Incident Handling: 4.2 Computer and Network Hacker Exploits, Part 1", SANS, 2003, p13.

[6] CERT/CC, "Advisory CA-2002-28 Trojan Horse Sendmail Distribution", Carnegie Mellon University, October 8, 2002, URL: http://www.cert.org/advisories/CA-2002-28.html (January 14, 2003).

[7] CERT/CC, "Incident Note IN-2001-06, Verification of Downloaded Software", Carnegie Mellon University, June 8, 2001, URL: http://www.cert.org/incident_notes/IN-2001-06.html (January 14, 2003).

[8] Orlowski, Andrew, "Debian Security breach", The Register, November 24 2003, URL: http://www.securityfocus.com/news/7513 (January 14, 2003).

[9] Halflife, "Bypassing Integrity Checking Systems", *Phrack Magazine Volume 7*, Issue 51, September 1, 1997, URL: http://packetstormsecurity.nl/mag/phrack/phrack51.zip (January 14, 2003).

[10] Kevin Poulsen, "Linux kernel backdoor blocked", The Register, November 7, 2003, URL: http://theregister.co.uk/content/55/33855.html (January 14, 2003).

[11] Automatiseringsgids, "Politie kijkt geen porno meer", ten Hagen & Stam Uitgevers, September 12, 2003, URL: http://www.automatiseringsgids.nl/news/default.asp?nwsId=23834 (January 14, 2003).

[12] NSRL, "Homepage National Software Reference Library", NIST, August 20, 2003, URL: http://www.nsrl.nist.gov/ (January 14, 2003).

[13] Reijnders, Maarten, RIAA: geripte en gedownloade mp3 herkenbaar, Webwereld, Donderdag, August 28, 2003, URL: http://www.webwereld.nl/nieuws/16074.phtml (January 14, 2003).

[14] WinZip Computing, Inc, "Homepage Winzip", WinZip Computing, Inc, 2003, URL: http://www.winzip.com (January 14, 2003).

[15] RSA Laboratories, "RSA Laboratories' Frequently Asked Questions About Today's Cryptography", Version 4.1, RSA Security Inc., 2000 , URL: http://www.rsasecurity.com/rsalabs/faq/3-6-6.html (January 14, 2003).

© SANS Institute 2004, As part of the Information Security Reading Room Author retains full rights.

[16] Bosau, Klaus, “Safety First, Tripwire– a situation report, part 2”, *Linux magazine issue 6*, 2001, URL: http://www.linux-magazine.com/issue/06/TripwirePart2.pdf (January 14, 2003).

[17] van Oorschot Paul and Wiener Mike, “Parallel collision search with applications to hash functions and discrete logarithms”, ACM Press, 1994, URL: http://www3.sympatico.ca/wienerfamily/Michael/MichaelPapers/parallelcoll.pdf (January 14, 2003).

[18] Rivest R.L., “The MD2 Message Digest Algorithm (RFC1319)”, IETF, April 1992, URL: http://www.ietf.org/rfc/rfc1319.txt?number=1319 (January 14, 2003).

[19] Rivest R.L., “The MD4 Message Digest Algorithm (RFC1320)”, IETF, April 1992, URL: http://www.ietf.org/rfc/rfc1320.txt?number=1320 (January 14, 2003).

[20] Rivest R.L., “The MD5 Message Digest Algorithm (RFC1321)”, IETF, April 1992, URL: http://www.ietf.org/rfc/rfc1321.txt?number=1321 (January 14, 2003).

[21] Boland Tim and Fisher Gary, “Selection of Hash Algorithms”, June 30, 2000, URL: http://www.nsrl.nist.gov/documents/hash-selection.doc (January 14, 2003).

[22] NIST, “Secure Hash Standard (Federal Information Processing Standards Publication 180-1)”, U.S. Department of Commerce, April 1995, URL: http://www.itl.nist.gov/fipspubs/fip180-1.htm (January 14, 2003).

[23] Dobbertin Hans, Bosselaers Antoon, Preneel Bart, “RIPEMD-160: A Strengthened Version of RIPEMD”, April 18, 1996, URL: http://www.esat.kuleuven.ac.be/~cosicart/pdf/AB-9601/AB-9601.pdf (January 14, 2003).

[24] Krawczyk H., Bellare M., Canetti R., “HMAC: Keyed-Hashing for Message Authentication (RFC 2104)”, IETF, February 1997, URL: http://www.ietf.org/rfc/rfc2104.txt?number=2104 (January 14, 2003).

[25] Lagerweij, Bart, “Bart's PE Builder - Build your own Bart Preinstalled Environment (BartPE) bootable CD/DVD”, 2003, URL: http://www.nu2.nu/pebuilder (January 14, 2003).

[26] Messmer, Ellen, “HP, IBM, Sun back Tripwire on 'file-signature' database”, *Network World Fusion*, August 6, 2003, URL: http://www.idg.com.hk/cw/readstory.asp?aid=20030806003 (January 14, 2003).

[27] Tripwire, “Tripwire File Signature Database (FSDB) Announcement; General Q&A and Fact Sheet”, Tripwire, August 5, 2003, URL: http://www.tripwire.com/fsdb/faqs.cfm (January 14, 2003).

[28] Leyden John, “AV vendors sell ‘blunt razor blades’”, The Register, March 26, 2002, URL: http://www.theregister.co.uk/content/archive/24596.html (January 14, 2003).

[29] Rosenberger, Rob, Better antivirus software is worse than a virus?, Vmyths.com, November 6, 2003, URL: http://www.vmyths.com/rant.cfm?id=605&page=4 (January 14, 2003).

[30] McClure Stuard, Scambray Joel, Kurtz George, *Hacking Exposed*, 4th edition (Dutch), Academic Service, 2003.

[31] Kemp, John, “Beyond the Basics of Windows Security: A Guide to Protecting System Integrity”, University of Oregon, fall-2003, URL: http://cc.uoregon.edu/cnews/fall2003/sysintegrity.html (January 14, 2003).

© SANS Institute 2004, As part of the Information Security Reading Room Author retains full rights.

[32] Floydman, “A poor-man Tripwire-like system on Windows 9x/NT”, August 2, 2000, URL: http://www.geocities.com/floydian_99/poormantripwire.html (January 14, 2003)

[33] Jeremy Rauch, Basic File Integrity Checking, August 14, 2000, URL: http://www.securityfocus.com/infocus/1408 (January 14, 2003).

[34] Mel M.X., Baker Doris, *Cryptography Decrypted*, Addison Wesley, 2001, 127 – 156.

[35] Myers, Marc, “Intrusion Detection Preliminaries: Sanitizing Your E-Commerce Web Servers”, March 31, 2000, URL: http://www.securityfocus.com/infocus/1219 (January 14, 2003).

© SANS Institute 2004, Author retains full rights.

© SANS Institute 2004, As part of the Information Security Reading Room Author retains full rights.

13. Appendix A: Integrity Verification Tools

File Integrity Verification Tools

DigestIT
Name: DigestIT 2004
Author: Ken Ballard
Location: http://free.deluxnetwork.com/~clarkeco/ (January 14, 2003)
File: digestIT 2004.zip
MD5: 2c2f422b87621eae1b6e0ffa26039841
SHA-1 278508c563c67dd36d7a4b906467625487605abe

SummerProperties
Name: SummerProperties 1.1
Author: Earthmagic
Location: http://www.earthmagic.org/?software (January 14, 2003)
File: SummerProperties 1.1 Setup.exe
MD5: d373ae4b6fe1ce07d1593b513b660f2e
SHA-1: b54a4fa27564b2312a5105e07b8188dfe660964e

System Integrity Verification Tools

Windows File Protection
Name: File Signature Verification (Windows 2000 version)
Author: Microsoft
Location: %systemroot%\system32\
File: Sigverif.exe
MD5: 47470c509fdf00db1e0cc3949a4262eb
SHA-1: 781b6acf4434f9b4b05ed4263a7638a344c5cc57

Name: File Signature Verification (Windows XP version)
MD5: 9E198D0A2366DC09A64FB8EE7233AD8E
SHA-1: 763B60DBD41D25652120233407DC6782C8B7915C

MD5Summer
Name: MD5Summer version 1.2.0.5
Author: Luke Pascoe
Location: http://www.md5summer.org (January 14, 2003)
File: md5v12005.zip
MD5: 3486baf3e090108427f3a58ffbc3560d
SHA-1: 24a03f441e70e2a50d11aa73e3c51e4be2261969

Fsum
Name: Fsum
Author: Slavasoft
Location: http://www.slavasoft.com/fsum/overview.htm (January 14, 2003)
File: fsum.zip
MD5: AB6C626408EC8271015463A9005F8ACE
SHA-1: 529F332627C1235701FA437D71F27FFAEA2B1C9B

© SANS Institute 2004, As part of the Information Security Reading Room Author retains full rights.

GFI LANguard System Integrity Monitor

Name:	LANguard System Integrity Monitor version 3
Author:	GFI
Location:	http://www.gfi.com/lansim/ (January 14, 2003)
File:	lansim.exe
MD5:	2734E5CDE7035FF6A778EE5FFB14836A
SHA-1:	3970BCEFE305B2D7B0B4C32E7AC430984A2BDA43

Osiris

Name:	Osiris version 2.4.0
Author:	Brian
Location:	http://osiris.shmoo.com/ (January 14, 2003)
File:	osiris-2.4.0-win32.exe
MD5:	7257DBCEFB9901177E1D316EB58B305E
SHA-1:	67F59B6C50A4912C58463E5F339D8FE1C48E86A8

© SANS Institute 2004, Author retains full rights.

© SANS Institute 2004, As part of the Information Security Reading Room Author retains full rights.

Last Updated: May 22nd, 2013

Upcoming SANS Training

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

SANS Malaysia @ MCMC 2013	**Cyberjaya, MY**	**Jun 03, 2013 - Jun 08, 2013**	**Live Event**
SANS Pen Test Berlin 2013	**Berlin, DE**	**Jun 03, 2013 - Jun 08, 2013**	**Live Event**
Industial Control Systems Security Training - Houston	**Houston, TXUS**	**Jun 10, 2013 - Jun 15, 2013**	**Live Event**
Security Impact of IPv6 Summit 2013	**Washington, DCUS**	**Jun 14, 2013 - Jun 16, 2013**	**Live Event**
SANSFIRE 2013	**Washington, DCUS**	**Jun 14, 2013 - Jun 22, 2013**	**Live Event**
SANS Canberra 2013	**Canberra, AU**	**Jul 01, 2013 - Jul 13, 2013**	**Live Event**
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**
SANS Mumbai 2013	**Mumbai, IN**	**Jul 22, 2013 - Jul 27, 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 San Francisco 2013	**San Francisco, CAUS**	**Jul 29, 2013 - Aug 03, 2013**	**Live Event**
SANS SEC 560: Network Penetration Testing @ Bangalore 2013	**Bangalore, IN**	**Aug 05, 2013 - Aug 10, 2013**	**Live Event**
SANS Boston 2013	**Boston, MAUS**	**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 Thailand 2013	**Bangkok, TH**	**Aug 19, 2013 - Aug 31, 2013**	**Live Event**
SANS Virginia Beach 2013	**Virginia Beach, VAUS**	**Aug 19, 2013 - Aug 30, 2013**	**Live Event**
Mobile Device Security Summit 2013	**OnlineCAUS**	**May 30, 2013 - Jun 06, 2013**	**Live Event**
SANS OnDemand	**Books & MP3s OnlyUS**	**Anytime**	**Self Paced**