Tripwire: Pioneering Integrity Scanning for Cybersecurity

Eugene H. Spafford
spaf@purdue.edu
Purdue University
West-Lafayette, IN, USA

CCS CONCEPTS

• **Security and privacy → File system security**; **Operating systems security**; **Intrusion detection systems**; *Symmetric cryptography and hash functions.*

ACM Reference Format:
Eugene H. Spafford. 2022. Tripwire: Pioneering Integrity Scanning for Cybersecurity. In *ACSAC 2022, Dec 5–9, 2022, Austin, TX*. ACM, New York, NY, USA, 4 pages. https://doi.org/XXXXXXX.XXXXXXX

1 BACKGROUND

In 1990 there was no Internet as we now know it. The proto-Internet (the NSFNet and regional networks) that existed at that time did not allow commercial traffic and was mostly composed of government and academic sites. The predominant operating system was UNIX, in its many variations. The majority of computers connected to the NSFnet and related were high-end workstations (e.g., Sun Microsystems and Vaxstations) and minicomputers, with some mainframes connected. What PC-type computers were in use ran primarily MS-DOS or classic Mac OS.

At this time there was no significant vendor community for security add-ons outside of some specialized mainframe systems. There was a growing market for some small anti-virus companies, but they were mostly focused on products for MS-DOS. There were some freeware tools in circulation for UNIX, such as the recently-released COPS package.[2]

However, there was growing concern over security. Computer virus incidents for PC-class computers were effectively doubling every year. There was a growing presence of intrusions into systems by various parties, known and unknown. Proof of concept viruses were known for UNIX systems[1, 11] and it was believed to be a matter of time before they appeared "in the wild." The Internet Worm and Wank Worm,[9, 16] along with the intrusions described in the *Cuckoo's Egg*[17] in prior years had also raised concern about network-based threats.

A series of stealthy intrusions occurred in early 1991, later documented in the book At Large[10], primarily targeting Sun computers. Spafford discovered that set of intrusions on his early honeypot systems. They were notable for not triggering any of the intrusion alarms then available. The intrusion and subsequent installation of a "backdoor" was done in a manner that circumvented the CRC check used in tools such as COPS and common to patch notices from vendors such as Sun Microsystems and the CERT/CC. The attacks were carried out through flaws in the software and then installed backdoor code in the shared libraries. Those files were altered so that the date/time of modification appeared unchanged, and the contents adjusted to match the previous CRC checksum.

ACSAC 22, Dec 5–9, 2022, Austin, TX
2022. ACM ISBN . . . $
https://doi.org/XXXXXXX.XXXXXXX

Spafford realized that a more reliable means of integrity verification was necessary. He sketched out the design of a system that used multiple message digest function values stored in a database, along with a record of i-node values. Initial functions were chosen from different hash families to avoid the possibility of common vulnerabilities. Kim was a talented undergrad student at the time who was seeking a project, and Spafford tasked him with writing the code.

The Tripwire tool was designed to monitor files and directories on a UNIX system for changes that could come from unauthorized modifications, software failures, malware, or intrusions. Over time, a number of other uses were also identified, including verifying updates and ensuring consistency with a baseline.

One inspiration for the design involved the placement of *trap or tripwire* files to expose snooping intruders.[1] This had proven highly effective in Spafford's firewall and bait systems.[2] If the system is properly configured, security administrators could learn when an intruder or local "snooping" user has accessed the trapped files, thus unavoidably updating the file's timestamp. The Tripwire program would report this.

2 THE ARTIFACT

On November 2, 1992, the initial Tripwire tool was released to one hundred beta test sites around the world, and therefter to a wider audience. Several bugs were identified, and four updates were released in 1993. In December 1993, the first formal release of Tripwire was made.

The original tool was written in C, with configuration being adaptable for different versions of Unix. Because of the heterogeneous nature of computer equipment at most sites, the design of Tripwire emphasized program and database portability. The code was written in the standard K&R C programming language,[5] adhering to POSIX standards wherever possible. The result was a program that compiled and ran on at least 28 BSD and System-V variants of UNIX, including Xenix and Unicos. All the early releases were developed in a cleanroom style,[13] with Gene Kim doing the development, and Spaf running the acceptance testing. The development environment at Purdue enabled the initial release to be adjusted to run under versions on UNIX on (at least) Sun 3, Sun 4, DEC VAX with BSD, AT&T 3B20, Ridge, Pyramid, and Sequent machines.[3]

The release included a comprehensive self-test suite to ensure that it was properly generating message digests, capturing i-node information, and reporting changes. The test suite was also portable

[1] Hence the original motivation for the name "Tripwire."
[2] There was never a formal publication about these systems, but I did blog about them several years later: https://www.cerias.purdue.edu/site/blog/2011/07/
[3] The UNIX environment was rich with different implementations and approaches at the time, which provided many benefits but also posed significant issues when attempting to write portable software—much more so than today.

```
# file/dir      selection-mask
/etc            R       # all files under /etc
@@ifhost solaria.cs.purdue.edu
 !/etc/lp               #       except for SVR4 printer logs
@@endif
/etc/passwd     R+12    # you can't be too careful
/etc/mtab       L       # dynamic files
/etc/motd       L
/etc/utmp       L
=/var/tmp       R       # only the directory, not its contents
```

Figure 1: An excerpt from an example `tw.config` file

to all of the supported systems. A set of template configuration files were also included, as was extensive documentation.

Tripwire was operated by initially generating a database based on directives in a configuration file. The configuration (`tw.config`) was a structured ASCII text file. This file was designed so it could be edited by the security administrator to indicate files and directories to be monitored, and the type of monitoring to conduct. Flags could be set to indicate whether items were allowed to be deleted or altered, and directories were marked as to whether new files could be created within them without sounding an alarm. There was an included macro language that allowed users to create template files that were expanded into the full configuration files, thus simplifying the installation at some sites.[4]

The `tw.config` file contained the names of files and directories with an associated selection-mask. A selection-mask might look like: `+pinugsm12-a`. Flags were added ("+") or deleted ("-") from the set of items to be examined. Tripwire interpreted this as, "Report changes in permission and modes, inode number, number of links, user id, group id, size of the file, modification timestamp, and signatures 1 and 2. Disregard changes to access timestamp." (See Figure 1.)

A flag existed for every distinct field stored in an inode.

To support the various typical files on target systems, the distribution had some standard macros for use in configuration files:

read-only files Only the access timestamp is ignored.
log files Changes to the file size, access and modification timestamp, and signatures are ignored.
growing log files Changes to the access and modification timestamp, and signatures are ignored. Increasing file sizes are ignored.
ignore nothing self-explanatory
ignore everything self-explanatory

When the program was run for the first time it would create a database of every scanned object, with the discovered (or calculated) values as indicated in the configuration file. This included digital hash values of files specified in the configuration, file permissions, modification dates, and existence. Information was also collected about directories and their properties.

Thereafter, the program could be run in check mode to compare current values against the archived values. Files that had changed in any way were reported. Directories that had changed attributes or contents were also reported. In this manner any alteration—whether benign or not—was flagged for operator investigation.

The database file was also a human-readable text file. This not only provided an alternate means of checking the database for potential tampering (e.g., comparison against a printed copy), but it allowed users to verify properties of individual files. The Tripwire distribution included a stand-alone program, `siggen`, that generated signatures for the files specified on the command line. This tool provides a convenient means of generating any of the included signatures for any file.

To allow the possible use of Tripwire at sites consisting of a large number of machines, the design allowed for configuration and database files not residing on the monitored machines. Input could be read from file descriptors, open at the time of Tripwire invocation. The file descriptors could be connected to UNIX pipes or network connections. Thus, a remote server or a local program could supply the files. Furthermore, the use of UNIX-style pipes also allowed for outside programs to supply encryption and compression services, should the administrator wish.

The advice given with the documentation was for system administrators to consider putting the configuration file, generated database, and executable program all on read-only media, or on media only mounted during single-user operation. These approaches would prevent any intruder from modifying any of those files to circumvent the scans. We were told that these restrictive uses were employed in some highly-sensitive government applications. However, most users chose to store the files on the regular file system, albeit with highest protection, sometimes including ACLs. This seemed to work well, as we received no credible reports of attack circumvention until well after 2000.[5]

Versions of Tripwire were released that used the MD2–MD5[14, 15] family of message digest algorithms, and the 4-pass SNEFRU[12] digest. Later, the 128-bit HAVAL[18] and NIST SHA algorithms were added. The original defaults were to use MD5 and SNEFRU. Tripwire also included POSIX 1003.2 compliant CRC-32 and CCITT compliant CRC-16 signatures. The intent was to provide multiple algorithms from different "families" to avoid there being any possibility of a weakness found in a single algorithm that would defeat the check. This was configurable by the users to trade-off CPU usage vs. greater confidence in the result.

[4]This was a user-contributed feature developed during the beta test phase.

[5]This is anecdotal evidence; successful attacks may have occurred earlier that were not reported to us.

3 RELEASE

Tripwire was published in source code form to several Usenet security-related newsgroups and also announced to several security mailing lists. (This was before the WWW was developed.). The code was under a modified BSD software license that allowed free use for non-commercial purposes.

The source code was provided in the Usenet postings and was available for download from the host `ftp.cs.purdue.edu`. It was later hosted at the COAST[6] ftp site, and a copy was available via the CERIAS[7] archive of COAST at `http://ftp.cerias.purdue.edu/pub/tools/unix/ids/tripwire/` for nearly 20 years, although it was not maintained after about the year 2000.

There were bug fixes and enhancements made to the tool set from 1992-1996, with subsequent releases made in the same manner as the original. The tool was supported by Gene Kim these years after its release. In 1997, Gene Kim and Wyatt Starnes formed the company Tripwire to develop and market an expanded, commercial version of the program that included a version for Windows.

In 2012, the Tripwire company donated a version of the then-current program to the open source community, where it may still be found (`https://github.com/Tripwire/tripwire-open-source`). The version on the Purdue ftp sites was withdrawn as it was no longer current or being supported, and visitors were directed to the open source version hosted by the Tripwire company.

Tripwire was extensively described in several published papers in 1994, included at ACM CCS[6], SANS[7], and the Usenix Application Developers conference[8]. It was also described in the first and second editions of [3, 4].

4 IMPACT

Tripwire was (and is) incredibly widely used. As the first free publicly-available intrusion detection tool, and the first integrity monitoring tool, it enjoyed great success and interest. We saw download numbers in the thousands over the first year after its release, and based on sharing in other venues it was likely used by tens of thousands. Subsequent releases saw even larger adoption. For several years it was recommended by major CIRT teams, several vendors, and present in all the common repositories.

Users from around the world contributed extensions and porting information for other versions of Unix. The README and documentation were translated into several languages other than English. At one time in the mid 1990s we counted over two dozen alternate hosting sites.

The three published papers (including the Purdue tech report versions) have been cited 1155 times according to Google Scholar.

After Gene Kim graduated and went to grad school, he continued to get requests for technical support of Tripwire. One commercial user paid Purdue University a license fee and a fee to GeneK so they could use it to certify machines on Wall Street as not being modified between trading days. This was proposed to the federal regulator as a means of showing immutability of the running code, and the regulator accepted it—and then recommended it to other trading firms.

We had reports of Tripwire finding otherwise-unnoticed intrusions, as we expected. We also received reports of unexpected uses:

- Tripwire flagged multiple instances of file changes caused by hardware failures at sites (e.g., disk issues) and software glitches. These were otherwise unnoticed by the system administrators.
- In later years, network shares for PC-class machines were monitored by Tripwire, and this resulted in discovery of malware on those shares.
- The most common report we received was discovery of unauthorized or unannounced system changes occurring in environments where more than one person had ability to make changes.
- The portability of the configuration and database files, and the ability to use them from remote shares or network connections, resulted in Tripwire being used to validate installations of software on multiple machines.
- We heard several stories where *tripwire* files were established, as per our original intent, which were used to detect insiders snooping into files and directories where they had no reason to explore. At least one such use was related to us as a trigger that uncovered insider-perpetrated fraud.

This author (Spafford) recalls visiting high-sensitivity government sites and corporations in the early 1990s where Tripwire was a *primary* security tool being used on critical servers. This included machines at the CIA, the Executive Office of the President, STRATCOM, and several major corporations. These uses were not documented in the open literature, to our knowledge, but correspondence with the operators in these locations—and some on-site visits—were part of our personal history with the tool.

In 1997, Gene Kim and partner Wyatt Starnes licensed the Tripwire brand and software from Purdue and formed the Tripwire company in Portland, Oregon. The company has gone on to become one of the leading security application vendors in the world, with many tens (or hundreds) of thousands of paying customers. As a nod to the origins of Tripwire, the company donated a refined rewrite of the code to the open source community where it is maintained and still available.

In the mid 1990s, Mark Pollit of the FBI Laboratory consulted with Spafford about the use of the message digests in Tripwire. This resulted in the creation of a national reference database of signatures for known child pornography. This database is still in use as a quick, relatively less-intrusive method of searching file systems of confiscated computers for prohibited content.

In 2004, Wyatt Starnes left Tripwire and founded Signacert. The company was based around the core concept of Tripwire: the database of message digests. Signacert was collecting digests of all authorized releases and patches of widely-used software, to be used in a product to scan systems for known and unknown software. Signacert was acquired by Harris corporation in 2010. In 2014, Wyatt died suddenly and Signacert was sold to private investors; it is not currently operating.

[6] COAST was the Computer Operations, Audit, and Security Technologies lab at Purdue, run by Spafford, 1992–1998.
[7] CERIAS is the successor to COAST, also started by Spafford in 1998, and about to celebrate its 25th anniversary.

Several modern IDS systems and malware detection systems have adopted the message digest concept pioneered in Tripwire for their own products.

Gene Kim left Tripwire in 2010 and has been an award-winning author and advocate for Visible Ops and DevSecOps methodologies over the last dozen years. Spafford has continued as a professor at Purdue University.

Tripwire was, to our knowledge, the first integrity monitoring tool for general-purpose computing systems. It remains a canonical example of that functionality.

REFERENCES

[1] Tom Duff. 1989. Experiences with Viruses on UNIX Systems. *Computing Systems* 2, 2 (spring 1989), 155–171.

[2] Daniel Farmer and Eugene H. Spafford. 1990. The COPS Security Checker System. In *Proceedings of the Summer Conference*. Usenix Association, Usenix Association, Berkeley, CA, 165–190.

[3] Simson Garfinkel and Gene Spafford. 1991. *Practical Unix Security*. O'Reilly & Associates, Inc., Sebastopol, CA.

[4] Simson Garfinkel and Gene Spafford. 1996. *Practical Unix & Internet Security*. O'Reilly & Associates, Inc., Sebastopol, CA.

[5] Brian W. Kernighan and Dennis M. Ritchie. 1978. *The C Programming Language*. Prentice-Hall, Englewood Cliffs, NJ.

[6] Gene H. Kim and Eugene H. Spafford. 1994. The Design and Implementation of Tripwire: A File System Integrity Checker. In *Proceedings of the 2nd ACM Conference on Computer and Communications Security*. ACM, ACM Press, NYC, NY, 18–29.

[7] Gene H. Kim and Eugene H. Spafford. 1994. Experiences with Tripwire: Using Integrity Checkers for Intrusion Detection. In *Systems Administration, Networking and Security Conference III*. Usenix Association, Berkeley, CA, 7 pages.

[8] Gene H. Kim and Eugene H. Spafford. 1994. Writing, Supporting, and Evaluating Tripwire: A Publicly Available Security Tool. In *Proceedings of the Usenix Applications Development Symposium*. Usenix Association, Berkeley, CA, 89–97.

[9] Thomas A. Longstaff and E. Eugene Schultz. 1993. Beyond preliminary analysis of the WANK and OILZ worms: a case study of malicious code. *Computers & Security* 12, 1 (1993), 61–77. https://doi.org/10.1016/0167-4048(93)90013-U

[10] Charles C. Mann and David H. Freedman. 1997. *At Large: the Strange Case of the World's Biggest Internet Invasion*. Simon & Schuster, NYC, NY.

[11] M. Douglas McIlroy. 1989. Virology 101. *Computing Systems* 2, 2 (spring 1989), 155–181.

[12] Ralph C. Merkle. 1990. A fast software one-way hash function. *Journal of Cryptology* 3, 1 (1990), 43–58.

[13] H. D. Mills, M. Dyer, and R. C. Linger. 1987. Cleanroom Software Engineering. *IEEE Softw.* 4, 5 (sep 1987), 19–25. https://doi.org/10.1109/MS.1987.231413

[14] R. L. Rivest. 1991. The MD4 message digest algorithm. In *Advances in Cryptology — Crypto '90*. Springer Berlin, Heidelberg, 303–311.

[15] R. L. Rivest. 1992. *RFC 1321: The MD5 Message-Digest Algorithm*. Technical Report. Internet Activities Board.

[16] E. H. Spafford. 1989. Crisis and Aftermath. *Commun. ACM* 32, 6 (jun 1989), 678–687. https://doi.org/10.1145/63526.63527

[17] C Stoll. 1989. *The Cuckoo's Egg*. Doubleday, NYC, US.

[18] Yuliang Zheng, Josef Pieprzyk, and Jennifer Seberry. 1992. HAVAL—a one-way hashing algorithm with variable length of output. In *International workshop on the theory and application of cryptographic techniques*. Springer Berlin, Heidelberg, 81–104.