Tripwire – A situation report, part 2

SAFETY FIRST!

KLAUS BOSAU

The second article in the series is concerned solely with the configuration of Tripwire, a special kind of monitoring tool. Using the example of the widely-used Academic Source Release (ASR), we explain the syntactical characteristics of the configuration file, and the important instrument of the selection mask.

The only configuration file is *tw.config*, the power unit of every *Tripwire* installation. For simple and rapid adaptation to platform-dependent specifics of the file system, the configuration file is in the form of a list. Each entry concerns only one object and follows the simple form:

```
[!|=] Object [Selection mask] [#comment]
```

As objects, entire directories or individual files are permitted. A directory represents its entire content. Be careful, as file system boundaries cannot be overstepped. For example if */usr* and */usr/lib* are mount points for two further partitions, and if the entire content of */usr* is to be monitored, both paths must be listed separately.

Bangs are good for objects which are constantly changing

Tripwire monitors each object found in *tw.config*, unless a preceding "!" (bang) expressly prohibits this. This exclusion marker is provided for non-critical objects like */dev*, whose monitoring would waste computing time. But beware, frequent use of the exclusion marker increases the risk of uninvited guests slipping in unnoticed!

For directories, therefore, there exists another option: "=" monitors the I-node of the directories, but not its content (the I-nodes and datazones of entries). This resource-saving long leash tightens

up in the event of an access to the content; but *Tripwire* shows neither the objects concerned, nor the type of modification itself. This is practical in the case of objects such as */tmp* or */var/spool/mail*, which are constantly changing in normal operation.

Select flags mark out more concrete properties

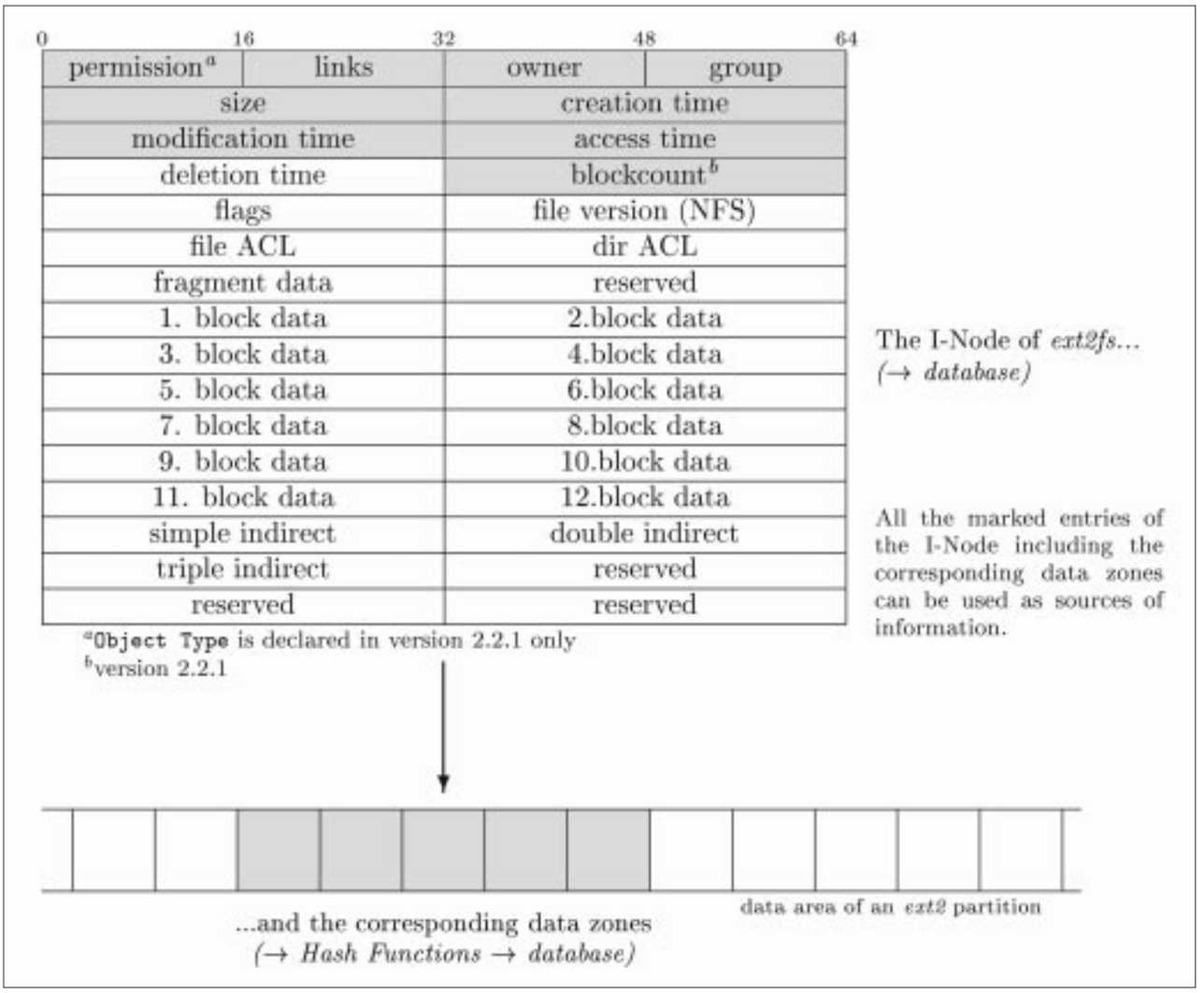
A far more refined synchronisation is possible with *select flags* . These – seventeen (!) – flags are represented by individual letters or numbers, each assigned to a concrete property of the object.

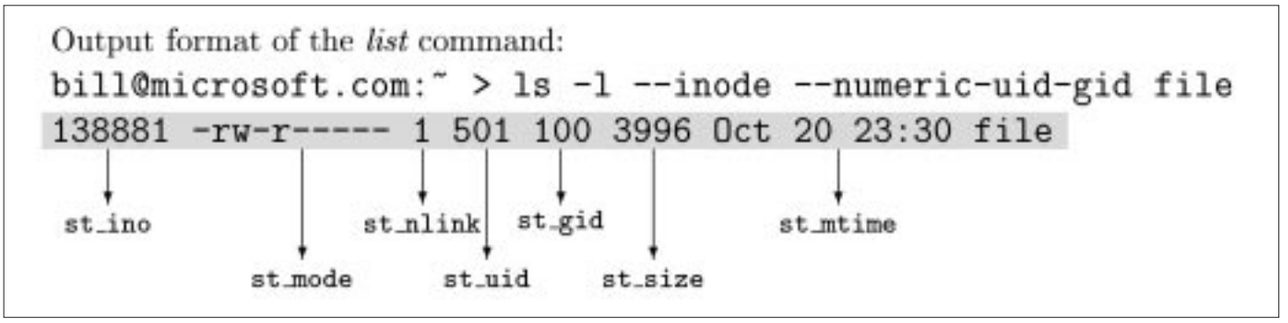
The spectrum of properties which can be selected is derived primarily from the range of data, and thus from the structure of the related file system. For the Linux platform the relationships are clear, because the ext2 file system, defined back in 1995 by Remy Card, has established itself as more or less the standard (for now).

The 128-byte I-node offers nine ext2-specific properties, which are commented on request by the ASR in the reference database. Version 2.2.1 in fact

[top]
Figure 1: Indices

[above]
Figure 2: The output format of the list command





knows eleven properties. Figure 1 shows that this means all the main fields of the I-node are captured.

The wallflowers *flags* and *file/dir ACL*, which until now have had no practical benefit, are proposed as interesting candidates for future expansions. All *ext2*-specific *select-flags* of a *Tripwire* protective shield are summarised in Table 1, specifying their respective meaning. Figure 2 shows the relationship to the output format of the *list* command. Version 2.2.1 has five further *ext2-select-flags* (24 with Windows NT); their usefulness is however limited, as they are almost identical to the well-known *select-flags* of the ASR.

The proposed indirect object characteristics are mainly suited for early detection of unintentional modifications to the file system or those induced as the result of incorrect functions.

Targeted subtle attacks are only to be warded off to the extent of preventing the intruder from attaining root-privileges – and thus access to the data zones. Insiders will know, or guess, that this endeavour will still be keeping zealous administrators of UNIX-type operating systems busy a hundred years from now. No practically-usable file system can ever really achieve this.

Ambitious attempts at a solution collide at this point with the limits imposed by resource hunger of cryptographic methods. A "high security operating system" with the performance of a pocket calculator is hardly acceptable.

Certainty the integrity of an object can be achieved only through the direct "survey" of the data zones by an effective signature function. Algorithms such as *SHA* and *Haval* (see below) are not deceived even if an intruder were to have full access to the object and unlimited time to cover up. In the ASR there are eight common signature functions to choose from for this. In Version 2.2.1. there are four.

As each function has been granted its own *select flag*, the administrator can react very flexibly to special requirements when configuring. These

Table 1: The *ext2-select-flags* of ASR and what they mean

select-flag	report	meaning
p	st_mode	Access rights and modes of execution (SUID-, SGID-Bit (!) and "text"-Bit)
i	st_ino	Number of I-node: The I-node number of an object is not altered by normal write/read operations. If such an inconsistency is found in the integrity report, this suggests that the object concerned has been deleted and replaced by a forgery with the same name
n	st_nlink	Number of hard links and/or sub directories: A special field of the I-node, the so-called "links count", specifies in the case of a directory the number of associated sub directories, and in the case of a file, the number of links associated with the I-node. In the latter case the counter goes up whenever a hard link is produced on the associated data zones. If, using *ln /etc/passwd /home/hacky* a hard link to */etc/passwd* is created, then the corresponding counter in the I-node of */etc/passwd* will increase by one. In the next integrity test the file would thus be shown as "changed".
u	st_uid	User-ID: User and group ID do of course act as superb targets for attacks of all kinds
g	st_gid	Group-ID
s	st_size	file size: a fully usable indicator, since it may not always be easy, so modify a configuration file in such a way that the file size is retained, and yet the desired effect is achieved
a	st_atime	date of last access: Just reading in a file is enough to update this sensitive entry in the associated I-node. Deploying the relevant *select-flag* in combination with a signature monitoring therefore makes little sense as processing the signature means the relevant file obviously has to be read. The *access timestamp* can be made visible using *ls -l —time=atime* ....
m	st_mtime	Time of last modification: This field is only updated when the relevant file has been modified and backed up again. The *modification timestamp* is something every Linux user is familiar with from content directories, which have been created using *dir* or *vdir*
c	st_ctime	Date of last status change, i.e. of last write access to the I-node: A status change occurs e.g. when changing the access rights of a file. The *inode timestamp* can be fetched with *ls -l —time=ctime* ....
t (2.2.1)	Object Type	File type (file, directory, symbolic link)
d (2.2.1)	Device Number	Partition type: Partitions are provided with a special identification number on installation, which gives information about the type of formatting. Magic number: The respective *select-flag* ensures that apart from other characteristics the partition's identification number is also commented in the reference databank, from which the I-node of the respective object stems.
l (2.2.1)	Size	"Logfile": Indicates that the size of the respective file in regular operation can only get bigger. Unlike s, which queries any change in the size of the file, a message is only issued here if a decrease is detected. (A typical candidate for example would be */var/log/messages*. The ASR only makes this functionality available in connection with other *select-flags* as *template* (">").
r (2.2.1)	File Device Number	Main device number: This property is declared only for device files and in this case designates the number of the device driver which belongs to the associated I-node. If the */dev* directory is listed using *ls -l /dev*, instead of the file size, the main device number (and any existing sub-device numbers) are shown
b (2.2.1)	Blocks	blockcount: Number of datablocks which are occupied by the zone pointer of the I-node. The size of an ext2fs-block is specified when the partition is installed (typically 1024 bits)

arise from the importance of the object, the available computing power and the individual requirement for system security. Table 2 provides an aid to decision-making. This lists the most important characteristics of the individual candidates and recent findings from the domain of cryptography.

Optimal Mixture is in demand

The selection mask, i.e. a complete description of all the interesting properties of an object, comes about in its simplest form through grouping the *select-flags* into character strings such as "+ug-a". In this example in fact user and group identification of the owner, but not the time of last access, are being monitored.

In fact the example also includes all other properties, because the ASR basically treats undefined matter as selected. Equivalent notations for "+ug-a" accordingly are "+pinugsmc123456789-a" and "-a". If it is really only the user and group identification of the

Table 2: The Arsenal

select-flag	Algo-rithm1	Throughput in MB/s (on P/200)	Estimated security	Special features
1	MD5	7.2	*****	The Message-Digest 5 algorithm developed by the Crypto-Pope Ronald Rivest corrects weaknesses in MD4. Odd numbers (four instead of the former three) and the quantity of additive constants (one each for the 64 part steps) are altered. This greatly protects the algorithm against analytically supported attacks, but at the expense of processing speed. The euphoric evaluations by leading cryptographers in the past appear to be in need of revision in the light of more recent findings. So far it has not been possible to erode the effectiveness of the hash function, but collisions – as previously with MD4 – for the Compression Function (an essential partial structure of the hash function) have been found – to be dealt with at length in a later instalment. MD5 is currently the most used hash algorithm, yet its future looks bleak. Leading cryptographers are now declaring that future attacks will have good chances aof success!
2	Snefru (R)	1.4	****	The ideal pyramid was eventually built by Snefru's successor, Khufu, and the first the Great Pyramid at Giza – was the finest and most successful. The algorithm conceived by Ralf Merkle at the Xerox Palo Alto Research Center (PARC) did not quite match up to the high esteem enjoyed by its famed namesake. By April 1990 a keen student managed to dethrone the previously popular two-step version and to pocket a prize of 1000 dollars as a result. PARC is now recommending the 8-step variant. Since to date every attempt to defeat the 4-step version used here with 128-bit signature format has failed, security performance may well still be within acceptable limits. But one very real drawback is the comparatively low data throughput.x
3	CRC-32 (also 2.2.1)	9.3	**	Refer to explanation of Cycle redundancy check (CRC-16).
4	CRC-16	16.2	*	Both of these robust and fast CRC algorithms are actually intended to identify transmission errors caused by hardware. The simplest variant of such a checksum function is realised by successive XOR linking of all the words in a message. Even the signature size of 16 and 32 bits prohibits any use in large or important files. Since a forged file must, however, come with not just the appropriate signature but also the corresponding functionality to be of any use, it's certainly worth the risk of using it for less critical objects.
5	MD4	14.4	***	This was introduced in 1990 and was very popular because of its rapidity on RISC processors. In 1998 came the sobering-up period: A slightly modified version proved to be reversible. MD4 is now seen as defeated and should therefore no longer be used for the protection of more important objects. Collisions for MD4 can be created artificially on an ordinary commercial PC in a few seconds! This impressively clarifies the relevance of this consideration.
6	MD2	0.3	****	Unusually slow, designed solely for old-fashioned 8-bit processors, while MD4 and MD5 can exhaust fully 32 bits, thus the capacity of most current processors! Although MD2 is the oldest of the three Message-Digest-Algorithms from RSA, there has until now never been any question of its effectiveness. The only finding of a cryptanalytical nature concerns a slightly modified version. Collisions could in fact only be created artificially when, in the so-called Padding (which will be dealt with at length in a later instalment) the insertion of the message length was omitted.
7	SHA (also 2.2.1)	5.4	*****	The Secure-Hash-Algorithm of NIST is, like most hash algorithms, structurally similar to MD4. In 1994 it was superseded on the grounds of an undocumented weak point by SHA-1. There are persistent conjectures that the National Security Agency (NSA) has made possible an access mechanism to external data material. This would obviously only fork as long as the weak point also remained secret and is not disclosed by over-zealous cryptographers. This hypothesis is not one to which the author of this article wishes to subscribe in view of the paltry supply of information. TSS seems to share this view, since SHA is in the current version 2.2.1 in unaltered form.
8	Haval (also 2.2.1)	10.7	****	The large 160-bit signature nevertheless makes SHA a good choice – even for security-critical objects. Even NASA prefers this algorithm in their Tripwire installation. This was created in 1992 at the University of Wollongong by Yuliang Zheng. Haval is the only one to display both a variable signature size (128, 160, 192, 224, or 256 bit), as well as a variable number of work steps (three, four, or five). The message is split at this point into blocks of 1024 bits, which are then processed in three, four or five cycles respectively by the Compression Function. This means there are a total of 15 different variants of the algorithm available for practical applications. In the Academic-Source-Release the four-step variant with 128-bit signature format is used. My evaluation with respect to security may have to be revised upwards. The unconventional structure is a lucky fluke, because this makes the algorithm immune to ordinary attacks, which are based almost without exception on MD4-methods.

Figure 3: An example for the configuration file tw.config

```
#
# Tripwire config-file
#

/                       R       # All objects under `/' are monitored.
/usr                    R       # Entry necessary if second hard drive assigned.
/boot                   R       # Ditto, as own partition.
!/dev                           # Not interesting!
=/tmp                           # Monitor directory only, but not content.
=/proc                          # Also sufficient in process file system.
=/home                          # Private!
/etc/ppp/pap-secrets    R-m     # Timestamp not important as frequent access.
/var/log                L       # Log files.
/var/log/messages       >       # Steadily growing file.

# "@@include" inserts external text into "tw.config" at run time. All
# host-specific properties could be described in a separate file.
@@include /root/tw.host-special

# Here a variable selection mask "@@var" comes into use, whose respective # importance
can be specified using the command line option "-Dvar=...".
# In the integrity test or update the same option must always be selected
# as at initialisation. The counterpart to "-D" also exists.
# With "-Uvar" a definition formulated in "tw.config" can be cancelled.
# (If "@@var" has not been specified in the command line,
# "E" is immediately placed here.):
@@ifndef var
@@ define var           E
@@endif
/opt                    @@var

# The macro "@@ifhost" represents what is certainly the easiest tool for
# adaptation to different computer architectures. In the example, what has been
# achieved is that one and the same area of the filesystem, depending on the
# computer, is dealt with differently by "Tripwire". (But to do so the
# environmental variable HOSTNAME, which is evaluated during run time, must be
# correctly set.):
@@ifhost babyboy.mamabear.org || babygirl.mamabear.org
@@ define TEMPLATE_S    N
@@else
@@ define TEMPLATE_S    E
@@endif
/var/Honeypot           @@TEMPLATE_S
# Naturally only relevant for "Bear cubs"!

# The content can also be structured with "@@define". Complex configura-
# tion files can be made much more clear with this:
@@define private        E
@@define critical       R-12+78
@@define secret         N-a
/home/Helga             @@private
/home/Axel              @@private
/root                   @@critical
/sbin                   @@critical
/etc/inetd.conf         @@critical
/etc/hosts.allow        @@critical
/root/banking-details   @@secret
```

owner which are to be scanned, this should be displayed by "+ug-pinsamc123456789" or "-pinsamc123456789". In the manpage of *tw.config* a corresponding indication has simply been omitted.

For users who are less obsessed with detail *Tripwire* provides pre-defined selection masks, so-called *templates*. Table 3 contains these standard cases. And combinations of *templates* and *select-flags* such as

"N-a" or "E+7" are permitted.

So the cryptic-looking character strings are markedly simplified with a *template* ; our example "User and group identification" is thus reduced to "E+ug". The selection mask can also be left out completely. Then the standard *template* "R" for "read-only" comes into play. But beware: the important *access-timestamp* is thereby excluded from the check!

The optimal combination of individual elements is produced from the function of the respective object and the general requirement for system security.

The resource use can, despite deliberate optimisation of the source code, turn out to be critically high. Assembler inlays were out of the question in *Tripwire* on grounds of portability. If *Tripwire* is running as a background process, this does not usually matter – on computers with sparse resources, though, it becomes a burden.

In this case the optimisation has to be weighed against less computing-intensive signature algorithms. I would recommend replacing the (now out of date) *template* "R" by a self-defined selection mask. A good compromise with respect to security and data throughput is "R-12+8".

A central configuration file on the Net

Professional users evaluated the feature of using just one configuration file on several computers of varying architecture at the same time. *Tripwire* has a single-stage preprocessor for this purpose, which interprets special keywords such as *@@include*, *@@ifhost* and *@@define*.

This effectively alleviates the use of *Tripwire* in large heterogeneous environments. In such a network for example it is conceivable that the configuration file could be reserved for a single computer and available to the other computers only on request.

Existing configuration files could be merged into a single one, with the respectively valid variants then being determined by the enquiring computer at run time. In corporate networks with ten or more computers this saves a lot of work for the administrator! Of course, this only makes sense if there can be no manipulation of the environmental variables of the enquiring computer!

An example clarifies the grey theory

Enough abstraction! Figure 3 shows a (made-up) example for *tw.config*, which presents, for better understanding, selected elements from the fund of the options sketched in this article.

I hope this little introduction to configuration may have sparked some interest in the inner life of the Filesystem Integrity Checker. The next in the series will have the same ambition: it offers a fascinating look into the unfathomable depths of the signature function. Also, interesting new features in Version 2.2.1 will be presented. ■

Info

[1] The ext2 filesystem overview: http://ftp.iis.com.br/pub/Linux/system/filesystem/ext2/Ext2fs-overview-0.1.ps.gz
[2] Snefru and accessories (Xerox): ftp://arisia.xerox.com/pub/hash
[3] National Institute of Standards and Technology: http://www.first.org
[4] Tripwire site of NASA: http://lheawww.gsfc.nasa.gov/~srr/tripwire.html
[5] Yuliang Zheng's Homepage: http://www.stcloudstate.edu/~bulletin/ee/index.html

■

Table 3: The templates of the ASR

template	Definition	Application
R	+pinugsm12-ac3456789	(R)ead-only: files which although generally accessible, can only be read (Standard)
L	+pinug-sacm123456789	(L)og file: User directories and files which are subject to constant modification
N	+pinugsamc123456789	ignore (N)othing: Full program. This selection mask is also ideal as a starting point for users' own definitions
E	-pinugsamc123456789	ignore (E)verything: For inventory. Only added or deleted objects are shown
>	+pinug-samc123456789	growing file: files which constantly grow in size but are not allowed to shrink
Device (2.2.1)	+pugsdr-intlbamcCMSH	Files which *Tripwire* must not open in the integrity test (these include all device files)