

Host-Based Intrusion Detection / Tripwire Lab (IDS)

Open Source Security Tools for Information Technology Professionals

School of Professional Studies (SPS)

The City University of New York (CUNY)

Aron Trauring
Adjunct Professor
CEO, Zoteca

Class 11 December 5th, 2005

What is Host-Based IDS?

- Focus on individual host (vs. network)
- File integrity checker — checks for altered files (similar to `chkrootkit`)

What is Checked?

- Changes to password file
- Changes to system configuration files
- Changes to file permissions

Advantages

- Fewer false positives
- No signature updates (activities are checked)
- Less prone to being tricked
- Less tending and tuning needed

Disadvantages

- Have to load and manage software on every host to be effective
- Alert comes after attack successful

Tripwire

- Originally FOSS project
- Went commercial in 2000
- Code released under GPL continues to be actively developed
- Commercial version supports Windows, management tools and GUI

How Tripwire Works — Overview

- Create a baseline database
- Contains attributes for important system files
- Encrypts and signs its own files
- Compare current state to baseline
- Can see if have been trojanized
- Can see intruder tracks in forensic analysis

How Tripwire Works — Components

- Policy — files and directories to snapshot with rules indicating violations
- Database — encrypted snapshot created by evaluating filesystem against the policies
- Can compare filesystem to snapshot at any time

Installing Tripwire

- Need to create site key passphrase to protect policy and configuration files from intruder
- Need to create local key passphrase to protect database file and generated reports file from intruder
- Need to configure policies
- Need to initialize snapshot database
- Assumption system is clean — need to install `tripwire` right after install system

Tripwire Update

- When you legitimately modify a file be sure to immediately re-initialize snapshot database
- Danger may think its a legitimate modification if don't

Policy Configuration File — General

- `twpol.txt` — main policy file (text format)
- Converted to encrypted version
- Backup and delete after configuring and testing

Policy Configuration File — Content

- Global settings
- Policy variables
- Policy directives

Policy Configuration File — Policy Directives

- Format: `file/directory name --> property mask ;`
- `file/directory name` — file or directory being tracked
- `property mask` — properties of file that are being tracked

Property Masks

Code Letters	Attributes Tracked
a	Last access time
b	Blocks allocated
c	Create/modify time
d	Device ID on which the inode resides
g	Group ID of file owner
i	inode number
l	Whether the file is allowed to grow
m	Modification timestamp
n	inode reference count (number of links)
p	Read/write/execute permissions
r	ID of device pointed to by inode
s	File size
t	Type size
u	User ID of file owner
C	CRC32 hash
H	Haval hash

Code Letters	Attributes Tracked
M	MD5 hash
S	SHA hash

Policy Directive Example

- `/etc/myfile --> +amcpsu`
- + or - indicates whether to track or ignore
- a — notify if last access time is modified
- m — notify if modification timestamp is modified
- c — notify if create/modify time is modified
- p — notify if file permissions are modified
- s — notify if file size is modified
- u — notify if user id of owner is modified

Predefined Template Property Masks

Variable	Property Mask
<code>\$(ReadOnly)</code>	<code>+pinugtsdbmCM-rlasSH</code>
<code>\$(Dynamic)</code>	<code>+pinugtd-srlbamcCM SH</code>
<code>\$(Growing)</code>	<code>+pinugtdl-srbamcCM SH</code>
<code>\$(Device)</code>	<code>+pugsdr-intlbamcCM SH</code>
<code>\$(IgnoreAll)</code>	<code>-pinugtsdrlbamcCM SH</code>
<code>\$(IgnoreNone)</code>	<code>+pinugtsdrlbamcCM SH</code>

Predefined Template Property Masks — Example Uses

- `$(ReadOnly)` — key configuration files access date can change but size or content can't
- `$(Growing)` — e.g. for log files
- `$(IgnoreNone) -SHa` — quickly define new mask

Predefined Variables in Policy File

- Allow for quickly setting rules
- `SEC_CRIT = $(IgnoreNone) -SHa ; # Critical files that cannot change`
- `SEC_SUID = $(IgnoreNone) -SHa ; # Binaries with the SUID or SGID flags set`
- `SEC_BIN = $(ReadOnly) ; # Binaries that should not change`
- `SEC_CONFIG = $(Dynamic) ; # Config files that are changed infrequently but accessed often`
- `SEC_LOG = $(Growing) ; # Files that grow, but that should never change ownership`

- `SEC_INVARIANT = +tpug ; # Directories that should never change permission or ownership`
- `SIG_LOW = 33 ; # Non-critical files that are of minimal security impact`
- `SIG_MED = 66 ; # Non-critical files that are of significant security impact`
- `SIG_HI = 100 ; # Critical files that are significant points of vulnerability`

Structure of Policy Rules

- `rulename` — referenced in report
- `severity` — how critical
- `rules` — policy directives, as above, using variables for most part

Make Changes to `twcfg.txt`

- `DBFILE =/var/lib/tripwire/$(HOSTNAME).twd`
- To avoid getting rooted change `/var/lib/tripwire` to a location on a write-protected/removable medium (e.g. USB stick, floppy, CD-R)
- Check other lines in the `twcfg.txt` and eventually make more changes (not necessary, unless you want to change e.g. the default location of some files).

Generate Keys and tw.cfg

- `twadmin --generate-keys -v -L /etc/tripwire/hostname-loc`
- `twadmin --generate-keys -v -S /etc/tripwire/site.key`
- After changing the configuration file, a binary version of that file has to be generated
- `twadmin --create-cfgfile --site-keyfile /etc/tripwire/site.key /etc/tripwire/twcfg.txt`
- Remove the clear-text config file afterwards (so intruder can't see)

Retrieve, Edit and Re-generate `tw.cfg`

- If necessary to make changes to the `tripwire` configuration
- `twadmin --print-cfgfile > twcfg.txt`
- After changing the configuration file, a binary version of that file has to be generated
- `twadmin --create-cfgfile --site-keyfile /etc/tripwire/si /etc/tripwire/twcfg.txt`
- Remove the clear-text config file afterwards (so intruder can't see)

Write Policy and Generate `tw.pol`

- `tw.pol` encrypted file (by default generated from `twpol.txt`)
- `man twpolicy` before changing `twpol.txt`.
- `twadmin --create-polfile /etc/tripwire/twpol.txt`
- Remove `twpol.txt`

Initialize Database and First Checks

- The database, located and named as specified in tw.conf, needs to be initialized
- `tripwire --init -v`
- If get missing file errors or other errors re-configure policies

Retrieve, Edit and Re-generate `tw.pol`

- If necessary to make changes to the `tripwire` policy
- `twadmin --print-polfile > twpol.txt`
- Edit `twpol.txt`
- `tripwire --update-policy /etc/tripwire/twpol.txt`

Checking for Problems

- `tripwire --check -I -n`
- `-I` — open the `tripwire` report in the editor (as specified in `tw.cfg`)
- `-n` — stop `tripwire` from showing output in the shell as well
- Change e.g. `/etc/passwd`
- Tripwire opens a report in the editor.
- All inconsistencies or changes that were found in the system are displayed.
- Next to each of the monitored changes there is a checkbox, which is checked (with `x`) by default.

- When the report is closed, all inconsistencies that are still checked will be updated in the database (and won't be mentioned again next time).

Regular Check

- When database needs to be readjusted after making changes to the system (e.g. install new software, kernel,...).
- If new software has been installed, `tripwire` will probably show a lot of inconsistencies in its report.
- If run `tripwire` immediately before and after making changes to the system, it is very unlikely that it has been compromised in the mean time.
- So just accept `tripwire` to make all changes to the database.
- The first few times may need to fine-tune your policy file as you will probably get a lot of 'noise' in the `tripwire` reports.
- Remove or lower restrictions on files/directories that are mentioned by `tripwire` each time (while you are sure there is nothing wrong) and update the policy.

Periodic Checks

- When there is no intention to change the database.
- It is still possible to allow acceptable changes to go in the database.
- Can make a daily cron job
- Tripwire won't keep your system safe but, when used correctly, at least tells you when you should be worried and take serious action.

Protecting Tripwire from Being Rooted

- Store binaries (`/usr/sbin/tripwire`) on read only media
- Store database(`/var/lib/tripwire`) on read-only or removable media