

# 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

# Class Agenda

**6:30-6:45** Article of the Week

**6:45-7:15** NIDS Lab — Part II

**7:15-7:45** Host Based IDS / Tripwire Lab — Part I

**7:45-8:00** Break

**8:00-9:00** Host Based IDS / Tripwire Lab — Part II

**9:00-9:30** Analysis and Management Tools

## **Instructor**

Aron Trauring – Adjunct Professor, CUNY SPS / CEO, Zoteca

**Email:** [atrauring@zoteca.com](mailto:atrauring@zoteca.com)

**Personal Website:** <http://aronst.org/>

**Zoteca Corporate Website:** <http://www.zoteca.com/>

## NIDS Lab

### Snort

- GPL licensed CLI tool
- Commercial company SourceFire provides tested rules, support and products
- Purchased by Check Point

### Other FOSS IDS

- Bro
- Prelude

### Invoking Snort

- `snort <options> <expression>`
- Expression similar to `tcpdump` selectors

### Snort Sniffer Mode

- similar to `tcpdump` or `tethereal`
- `-v` — displays TCP/IP headers
- `-d` — displays application layer data
- `-e` — displays link layer headers

### Snort Logging Mode

- logs all packets sniffed
- `-l <logpath directory>` — logs into subdirectories by IP address
- `-h <home network (local IP range)>` — logs into subdirectories by non-local IP address
- `-b` — logs into binary file that can be analyzed by `ethereal` or `tcpdump`

### Snort IDS Mode

- logs only packets that are suspicious or warrant further attention
- `-c <config file>` — configuration file sets parameters for logging
- In Ubuntu and others: `/etc/snort/snort.conf` is default config file
- `/var/log/snort` is default logging directory
- can leave off `-vde` switches which slows down and may cause packets to drop

## Snort Alert Modes

- `-A full` — full alert information. Default when nothing is specified
- `-A fast` — logs only packet header and alert type (useful on fast networks)
- `-M <workstation>` — use smb to send to Windows pop-up service
- `-s` — send to Unix syslog
- send to database for later analysis

## Example Snort Session

- `snort -A full -c /etc/snort/snort.conf -b`
- in other window run `nmapfe`
- Control-C after a while
- `ls -al /var/log/snort`
- `ethtool -r /var/log/snort/<logfile>`

## Snort Configuration — Home Network

- `var HOME_NET <addresses>`
- Addresses comma-separated list of local network e.g. `192.168.1.0/24`
- `var HOME_NET $<interface>`
- Takes IP and mask from interface configuration e.g. `eth0`
- `EXTERNAL_NET` can also be defined
- default for both is `any`

## Snort Configuration — Internal Servers

- Define servers you have running on your network
- Can specify ports so only registers attacks on open ports
- Limits false positives

## Snort Configuration — Decoders and Pre-processors

- Run on traffic before it passes through rule sets
- For proper formatting or types of traffic easier to deal with as a class
- Example: decoder to reassemble fragmented packets so properly formatted
- Example: pre-processor for port scanning traffic which is high-volume and better dealt with en masse
- Only change configuration as gain more experience with the tool

### Snort Configuration — Output Modules

- Used for managing output
- Three modules: syslog, database, unified (binary format)
- `output <module name>: <configuration options>`
- See documentation for details

### Snort Configuration — Rule Sets

- Rule sets are in `rules` directory
- In config file can turn on/off whole rule sets via deleting/adding comment #

### Snort Configuration — Individual Rules

- Go to rules directory and edit rules files
- Can turn on/off individual rule via deleting/adding comment #
- Better to work on individual rules than whole set if relevant in anyway

Rule Classes	Descriptions
attack-responses rules	These are alerts for common response packets after an attack is successful. They should rarely report false positives and should be left on in most cases.
backdoor rules	These are common signs a backdoor or Trojan horse program is in use. They will rarely be false positive.
bad-traffic rules	These rules represent nonstandard network traffic that should not typically be seen on most networks.
chat rules	Look for standard sign-ons for many popular chat programs. If chat is allowed explicitly or implicitly, then these alerts should be turned off. Also, note that these are not silver bullets for chats and will not detect all types of chat traffic. Still, they can be helpful in ferreting out the worst offenders.
ddos rules	Look for standard distributed denial of service types of attacks. On a DMZ and WAN, these alerts don't serve much purpose, because if you are under a distributed denial of service you will probably know it right away. However, they can be very useful inside the LAN to see if you have zombie machine participating unknowingly in a DDOS attack on another network.

<b>Rule Classes</b>	<b>Descriptions</b>
dns rules	Look for some standard exploits against DNS servers. If you aren't running your own DNS, you can turn these off.
dos rules	Similar to the ddos.rule set above.
experimental rules	These are turned off by default. These are generally used only for testing new rules until they are moved into one of the other categories.
exploit rules	These are for standard exploit traffic and should always be enabled.
finger rules	These rules flag traffic having to do with finger servers. If you are not running finger anywhere, you could probably turn these off. However, finger servers often are running hidden from the system administrator, so you could leave these on as they shouldn't generate false positives if you don't have any.
ftp rules	Same as finger rules but looking for FTP exploits. Again, there is no harm in leaving them enabled even if you don't have FTP servers since it will alert you to any rogue FTP servers you may have.
icmp-info rules	These rules track the use of ICMP messages crossing your network, for example, pings. These are often the cause of false positives, and you may want to disable the whole lot unless you want to keep a close eye on ICMP traffic on your network. Another class for known bad ICMP traffic, icmp rules catches ports scans and the like.
icmp rules	Cover bad or suspicious ICMP traffic such as port scans, and are less likely to generate false positives. However, it is possible they will be triggered often on a busy network with lots of diagnostic services running.
imap rules	Rules regarding the use of Internet Message Access Protocol (IMAP) on your network.
info rules	Trap miscellaneous error messages on your network from Web, FTP, and other servers.
local rules	You add your own custom signatures for your network in this file. This file is empty by default. See the documentation for information on writing a custom Snort rule.
misc rules	Rules that don't fit under one of the other categories or don't warrant their own sections are in this file. An example would be older alerts like Gopher server exploits.

<b>Rule Classes</b>	<b>Descriptions</b>
multimedia rules	Track usage of streaming video type software. If you allow streaming video applications or use video conferencing on your network, then you will want to disable these rules.
mysql rules	Watch for administrator access and other important files in a MySQL database. If you don't run MySQL, then you can probably disable these alerts. Also, if your MySQL database is under development, these might trigger a lot of false positives.
Netbios rules	This class of rules alerts you to various NetBIOS activity on your LAN. Some of them are obvious exploits. However, others, such as the NULL session alerts, may happen normally on a Windows LAN. You will have to play with this section to figure out the rules that are appropriate for your LAN.
nntp rules	News server-related rules. If you don't run network news on your servers, you can probably turn these off.
oracle rules	Oracle database server rules. Again, if you don't run it, turn it off.
other-ids rules	These rules are related to exploits on other IDS manufacturers' boxes. Chances are that you don't have any NIDS on your LAN, but if you do, leave these on.
p2p rules	Rules governing peer-to-peer file sharing software use. These rules will create alerts during normal use of these products, so if you allow this software then you will need to turn these off.
policy rules	This file contains various alerts relating to allowed activity on the LAN, such as Go-to-my-pc and other programs. You should review these and enable only the ones that apply to your internal policies.
pop2 / pop3 rules	Both files to mail servers. Most companies, if using POP, will be using a POP3 server. If you have either of these types of servers, leave these rules on; if not, disable them.

<b>Rule Classes</b>	<b>Descriptions</b>
porn rules	These are some rudimentary traps for pornography-related Web surfing. These are by no means a replacement for a good content-filtering system, but can catch some of the more egregious violators.
rpc rules	This class handles remote procedure call (RPC) alerts. Even though you may not think you are running any of these services, they often run as part of other programs, so it is important to be aware when this is happening on your LAN. RPC can enable remote code execution and is often used in Trojans and exploits.
rservices rules	Track use of various remote services programs, such as rlogin and rsh. These are insecure services in general, but if you have to use them, they can be tracked closely with this rule set.
scan rules	Alert you to use of port scanning programs. Ports scans are a good indication of illicit activity. If you use port scanners, you will want to either turn off Snort during those times or disable the particular rule for your scanner machine.
shellcode rules	This class looks for packets containing assembly code, low-level commands also known as shell code. These commands are often integral to many exploits such as buffer overflows. Catching a chunk of shell code flying by is often a pretty good indication that an attack is underway.
smtp rules	Govern alerts for mail server use on the LAN. This section will need some fine-tuning, as many normal mail server activities will set off rules in this section.
sql rules	Rules for various SQL database programs. If you don't run any databases you can turn these off, but it's not a bad idea to leave them on just in case there are SQL databases running that you don't know about.
telnet rules	Track Telnet use on the network. Telnet is often used on routers or other command line devices, so it is a good thing to track even if you don't run Telnet on your servers.

Rule Classes	Descriptions
tftp rules	TFTP (trivial FTP) is an alternate FTP server often run on routers. It can be used to upload new configurations and therefore is worth keeping an eye on.
virus rules	Contain signatures of some common worms and viruses. This list is not complete and is not maintained regularly. It is not a replacement for virus scanning software but can catch some network-aware worms.
web-attacks rules web-cgi rules web-client rules web-coldfusion rules web-frontpage rules web-iis rules web-php rules	All these classes refer to various kinds of suspicious Web activity. Some are generic, such as the web-attacks class. Others, like web-iis and web-frontpage, are specific to a particular Web server platform. However, even if you don't think you run a Microsoft Web server or use PHP, it is worth leaving them all running to uncover any of this kind of activity on your LAN you may be unaware of. You will have to do some fine-tuning of the rule sets, especially if your Web servers are in active development.
X11 rules	Track the use of the X11 graphical environment on your network.

### Webmin Snort

## Host-Based Intrusion Detection / Tripwire Lab

### 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
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-srlbamcCMSH</code>
<code>\$(Growing)</code>	<code>+pinugtdl-srbamcCMSH</code>
<code>\$(Device)</code>	<code>+pugsdr-intlbamcCMSH</code>
<code>\$(IgnoreAll)</code>	<code>-pinugtsdrlbamcCMSH</code>
<code>\$(IgnoreNone)</code>	<code>+pinugtsdrlbamcCMSH</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-local.key`
- `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/site.key /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

## Analysis and Management Tools

### Analyzing Information

- Too much information leads to “analysis paralysis”
- Sysadmins often ignore available information for “lack of time”
- Need tools to quickly analyze and manage output of tools

### System Logs

- `/var/log/messages`
- `/var/log/syslog`
- Failed logins precursor to attack
- Servers rebooting at strange time
- Syslog Data Mining Tools [http://www.syslog.org/index.php?name=Web\\_Links&req=viewlink&cid=4&min=0&orderby=titleA&show=10](http://www.syslog.org/index.php?name=Web_Links&req=viewlink&cid=4&min=0&orderby=titleA&show=10)

### Swatch

- How To Configure Swatch <http://sial.org/howto/logging/swatch/>
- Looks for some string in logfile
- Takes notification action (e.g. send email) if match is found

### SEC — Simple Event Correlator

- Triggering Actions Using SEC <http://sial.org/howto/logging/sec.pl/>
- Looks for some string in logfile
- Takes real action (e.g. generate firewall rule) if match is found

### Network Monitoring Systems

- Nagios Screenshots <http://www.nagios.org/about/screenshots.php>
- Pluggable architecture
- Works with backend database
- Nagmin — Webmin management plugin
- Nagmin Screenshots [http://nagmin.sourceforge.net/screen\\_shots.htmNa](http://nagmin.sourceforge.net/screen_shots.htmNa)
- OpenNMS

**Analysis Console for Intrusion Databases (ACID)**

- Works with `snort` and `syslog`
- Port all IDS data into a database
- Sort and organize from database
- ACID Screenshots <http://www.cert.org/kb/acid/>

**SGUIL - The Analyst Console for Network Security Monitoring**

- GUI-based vs. ACID web-based
- Claims deeper and quicker analysis
- SGUIL Screenshots <http://sguil.sourceforge.net/index.php?page=screenshots>
- SGUIL Flash Demo <http://sguil.sourceforge.net/index.php?page=flashdemo>