

Build your own IDS with Logsurfer

by Boris Loza, PhD, CISSP

If you look up IDS in a computer dictionary, you probably won't find it. However, this abbreviation is well known among security specialists. IDS stands for Intrusion Detection System, and it's a relatively new term in the computer world.

An IDS is a passive-monitoring device that doesn't interfere in any way with the existing network traffic or existing applications. It's just a sophisticated data collection and analysis tool. IDS won't protect your company from being attacked, but it will continuously monitor, detect, and alert IT personnel to intrusion attempts.

Host-based vs. network-based IDS

One of the existing classifications divides IDS into network- and host-based systems. A network-based IDS analyzes network packets. It can detect intrusions but it can't determine if the intrusion has been successful. A host-based IDS analyzes security activity logs. This type of IDS can see successful intrusions but it might not necessarily see the unsuccessful attacks. A host-based IDS also detects intrusions even if network connections are encrypted or if attackers use dial-up connections or consoles.

Both approaches have their strengths and weaknesses. Therefore, hybrid IDS (a combined network- and host-based system) tries to reduce those limitations.

Internal breaches still make up a significant portion of attacks—44 percent—according to the Computer Security Institute/FBI survey. This statistic emphasizes the need for detecting intrusions on the machines inside the network as well as the perimeter. In this case, a host-based IDS can play a significant role. You can build such an IDS using a free log analyzer like Logsurfer.

Logsurfer

This application is designed to monitor any text-based log files on your system in real time. One of the most powerful features of Logsurfer is its ability to create new rules as needed. This functionality is especially useful when a specific, reoccurring message, such as filesystemfull, needs immediate attention by an administrator but only requires an alert to be sent once per hour. You can achieve this goal by sending mail to the administrator and including a new rule to ignore the same message for a specified time period.

With dynamic rules, you can redefine at run-time any events that are of interest. A good example might be a telnet connection from a given host. After the connection was denied, you might want to follow up with all other events related to this host, if they are considered unimportant at any other time or not.

Installation

You can download the latest version of Logsurfer via anonymous ftp from

<ftp://ftp.cert.dfn.de/pub/tools/audit/logsurfer/>

Logsurfer is written in C, and once compiled, can be distributed and installed on any other boxes with the same hardware architecture. Be sure to check a PGP signature of the distribution. The public key of the signer (Ruediger Riediger) is available at

<http://www.cert.dfn.de/~riediger/mykey.asc>

If you have a PGP package already installed, add this key to your public key ring

```
pgp -ka mykey.asc
```

where mykey.asc is a file that contains Ruediger Riediger's public key. To ensure that you have an unmodified version from the DFN-CERT, run the following command:

```
pgp logsurfer-1.5.tar.asc
```

If the signature doesn't match, the software has likely been compromised. Don't use this version.

Logsurfer installation is quite straightforward. After unpacking the package with tar you can then type configure. When the configuration is finished, you can type make and then type make install. We suggest defining the -DSENDMAIL_FLUSH option in the CFLAGS definition for the Makefile. This will prevent memory exhaustion when an unusual amount of activity causes many instances of sendmail.

The binary Logsurfer and the man pages are then placed into the correct system directories. The default directory for Logsurfer is /usr/local/. You can modify this default using several options when executing the configure script. (Read the readme and install files for more compilation options.)

Configuration

Logsurfer has a sophisticated configuration syntax. But if you play with the configuration facilities, you'll learn how to use them.

If you didn't specify another location for the logsurfer.conf (the Logsurfer configuration file) during installation, you need to create one in the /usr/local/etc directory. This file will contain the rules to analyze messages, as shown in **Table A**.

Table A: Logsurfer rules to analyze log messages

| Rule Name | Description |
|-----------|-------------|
|-----------|-------------|

| | |
|--------------------|--|
| <code>match</code> | Determines what lines match the rule (as defined within <code>egrep (1)</code>). You can use spaces, tabs, <code>"</code> , or <code>"</code> in order to determine a regular expression. |
|--------------------|--|

| | |
|------------------------|--|
| <code>not_match</code> | Excludes any line matched by <code>match</code> . Specify <code>not_match</code> as "-" if you don't use exceptions. |
| <code>stop</code> | If the <code>stop</code> isn't "-" and the line is matched, then the rule is being deleted from the list of active rules. |
| <code>not_stop</code> | If the line matches <code>stop</code> and it isn't specified as "-", the rule isn't deleted. |
| <code>timeout</code> | The number of seconds this rule should be active. If the timeout is 0, then the rule won't timeout. |
| <code>continue</code> | This rule is optional. If this keyword is given, then Logsurfer will continue to find another matching rule. Otherwise, the matching is stopped. |
| <code>action</code> | Action of a rule. Could be: ignore, exec program, <code>pipe</code> program, <code>open</code> context, <code>delete</code> context, <code>report</code> program context and <code>rule</code> (before behind top bottom). |

In the action field, `program` is an application that you can call. Execution is started in the background so that the execution of the Logsurfer process isn't impacted. Context is a set of messages that match certain regular expressions. All messages (even unimportant ones) are stored in structures called contexts for a certain period of time. This is useful if you're interested in these messages later on but the log file isn't accessible. You can control the content of each context with the rules shown in [Table B](#).

Table A: *Parameters to control the context behavior*

| Field | Description |
|--------------------------|--|
| <code>match</code> | Determines what line matches the context. |
| <code>not_match</code> | Excludes any line matched by <code>match</code> . Specify <code>not_match</code> as "-" if you don't use exceptions. |
| <code>line_limit</code> | Defines the maximum number of lines that are to be stored within the context. It's always useful to define a limit to avoid denial of service attacks. |
| <code>timeout_max</code> | Specifies how long this context should be active before the default action is executed. |
| <code>timeout</code> | In addition to the absolute timeout, the default action is executed if a relative timeout (in seconds) is defined and no new log message is added to the context within that time period. |
| <code>action</code> | The specified action is carried out whenever the maximum number of lines, the <code>timeout_max</code> , or <code>timeout</code> is reached. All actions except <code>open</code> , <code>delete</code> , and <code>rule</code> are available. |

When you create a new rule, the first word following the keyword, rule, must be one of the following:

- `before`—The new rule is inserted before the current rule.
- `behind`—The new rule is inserted behind the current rule.
- `top`—The new rule is inserted at the top of all rules.
- `bottom`—The new rule is appended at the end of all existing rules.

After you specify the position of a new rule, you can create it using the same syntax as we explained for all rules. For more details, see the man pages `logsurfer(1)` and `logsurfer.conf(4)`.

Preparing the system

Because we're going to build a host-based IDS, we need to prepare the system to help us collect as much information as possible. Add the following line into the `/etc/syslog.conf` file:

```
* /var/adm/logsurfer.log
```

Based on this configuration, the `syslogd` daemon will send all event data into a `/var/adm/logsurfer.log` file (see `syslog.conf(4)`).

If you decide to trace the incoming TCP connections, restart the `inetd` daemon with the `-t` option. This will log the client's IP address and TCP port number, along with the name of the service.

The Logsurfer configuration also allows you to start external programs in response to events. For this reason, running Logsurfer as root is dangerous. Create a non-privileged user account that will be responsible for Logsurfer. Make all log files readable by the group that this user belongs to. Logsurfer can be started with this account.

You can send the `logsurfer.log` files from all your servers to another one that will act as a logfile collector. In this case, if the system is compromised, an intruder can't alter the `logsurfer.log` file. For such a configuration, you'll need to run Logsurfer only on that machine, but keep in mind the possibility of network problems.

Security policies and rules

IDS security policies detect attacks or suspicious behavior by identifying an attack signature. Such a signature is a series of events or audit log messages that occur during a specific attack. Policies contain rules that detect and respond to conditions or events on your UNIX box.

Let's suppose you decide to monitor the usage of the `su` command. In case of an unsuccessful `su` attempt, a security administrator is notified by email. The `/var/adm/logsurfer.log` file will keep records of usage of the `su` command. The following entry from this file shows a failed `su` from user `mark` to `root`:

```
Jan 21 14:18:21 sunserver20 su: 'su root'  
=>failed for mark on /dev/pts/2
```

In this case, your entry for the Logsurfer configuration file will look like this:

```
"\su root\' failed " - - - 0 pipe
"/usr/local/bin/start-mail adm@foo.com
\"Failed SU ROOT\" \"$0"
```

This rule matches the failed su attempts (the " \su root\' failed ") without exceptions (the first "-"). There are no self-destroying rules (the second "-"). It has no not_stop rules (the third "-") and no timeout (the "0"). For each message, a user Adm@foo.com will get email with a subject "Failed SU ROOT" and a message body

```
Jan 21 14:18:21 sunserver20 su: 'su root' failed for mark on /dev/pts/2
```

The Start-mail application is a part of the Logsurfer distribution (subdirectory contrib), and allows you to specify the subject of and recipient for email messages.

You can run this policy from the command line:

```
$logsurfer /var/adm/logsurfer.log
```

Additionally, Logsurfer can accept input via stdin. This is useful if you decide, for example, to monitor unusual logins—let's say after 00:00 A.M. on Saturday. You may create the following rules:

```
'Mon|Tue|Wed|Thu|Fri' - - - 0 ignore
'user1|user2|user3|user4' - - - 0 ignore
'.*' - - - 0 pipe "/usr/local/bin/
=>start-mail Adm@foo.com \"Unusual
=>logins\" \"$0"
```

These rules will ignore all logins from Monday through Friday and notify about any other logins from Saturday until Sunday except for user1, user2, user3, and user4. You can invoke this policy from the crontab:

```
59 0-5 * * 6-0 last | /usr/local/bin/logsurfer
=>-c /usr/local/etc/Check_logins.conf
```

In this case, Logsurfer won't use the default configuration file /usr/local/etc/logsurfer.conf, but another one that's specified with a -c option.

Logsurfer uses the GNU regex library (same as the UNIX utility egrep). For a great explanation of regular expressions, you can read the documentation in the GNU regex library that comes with the Logsurfer distribution. Please take one suggestion based on our experience: make the rules as simple as possible!

Building IDS

After you understand the Logsurfer syntax, you can build an IDS. First, create a configuration file like that shown in [Listing A](#)

Listing A: *A simple configuration file for Logsurfer*

```

# This is a default logsurfer configuration file /usr/local/etc/logsurfer.conf
#
# Check failed login attempts:
`login: REPEATED LOGIN FAILURES ON ` - - -
 0 pipe "/usr/local/bin/start-mail Adm@foo.com
=>\"Failed LOGIN ATTEMPTS\" \" \"$0\"

# Possible nmap scanning
`sendmail\[[0-9]*\]: NOQUEUE: SYSERR: putoutmsg|accept: Protocol error|Broken pipe' -
- - 0 pipe
=>"/usr/local/bin/start-mail Adm@foo.com \"Possible NMAP Scanning\" \" \"$0\"

# TCP Wrapper rules
`refused connect from' - - - 0 pipe "/usr/local/bin/start-mail Adm@foo.com \"Refused Connect\" \" \" \"$0\"

# System problems
`file system full|address not found|unix: sorry' - - -
 0 pipe "/usr/local/bin/start-mail Adm@foo.com \"System Problem\" \" \"$0\"

# Look for unexpected system reboots
`reboot' - - - 0 pipe "/usr/local/bin/start-mail Adm@foo.com \"System Reboot\" \" \"$0\"

# Look for unexpected successful su commands
`su: \'su root\' succeeded" user1|user2|user3 - -
 0 pipe "/usr/local/bin/start-mail Adm@foo.com
=>\"SU Root\" \" \"$0\"

# Look for failed `su root'
`su: \'su root\' failed " - - - 0 pipe "/usr/local/bin/start-mail Adm@foo.com \"Failed SU Root\" \" \" \"$0\"

# Ignore the rest
`.*' - - - 0 IGNORE

```

You can start Logsurfer in the background from the command line or during boot time from the `/etc/init.d/` directory. To start from the command line, use the following command:

```

$logsurfer -p /var/adm/logsurfer.pid -f /
=>var/adm/logsurfer.log &

```

The `-f` option indicates the special follow mode if the end of file has been reached. Logsurfer will sleep for one second before trying to read a new line. This is similar to the `-f` option of the `tail(1)` command. The `-p` option writes the Logsurfer process ID into the given filename. We'll use this feature in a crontab to send a HUP signal after shifting the `logsurfer.log` file:

```

59 23 * * * > /var/adm/logsurf.log & kill
=>-HUP `cat /var/adm/logsurfer.pid`

```

This will clean (shift) the `/var/adm/logsurfer.log` file daily at 23:59.

In addition to the `/var/adm/logsurfer.log` file, you'll want to monitor the log files shown in [Table C](#).

Table C: Log files generated by Logsurfer

| Log file | Description |
|--------------------------------|--|
| <code>/var/adm/pacct</code> | Records the commands run by all users. You must turn process accounting on before this file is generated. Use the <code>lastcomm</code> command to audit commands run by a specific user during a specified time period. |
| <code>/var/adm/aculog</code> | Keeps track of dial-out modems. Look for records of dialing out that conflict with your policy for use of dial-out modems. Also look for unauthorized use of the dial-out modems. |
| <code>/var/adm/sudo.log</code> | Records sudo activities (if you use <code>sudo</code>). |

You can create more than one configuration file and run multiple Logsurfer instances with the `-c` option

```
$logsurfer /var/adm/logsurfer.log &
$logsurfer -c /usr/local/etc/aculog.conf /
=>var/adm/aculog &
$logsurfer -c /usr/local/etc/sudo.conf /
=>var/adm/sudo.log &
```

where `aculog.conf` and `sudo.conf` are different logsurfer configuration files.

To start Logsurfer automatically during a system startup, create a script in the `/etc/init.d/` directory. Be sure to make hard links for the logsurfer start-up file (see [init \(1M\)](#) and `init.d`) to control changes that occur whenever the system status changes.

Conclusion

A commercial IDS would probably give you a nice GUI interface and predefined security policies and rules. But such applications are usually not cheap. Therefore, if you're on a tight budget or want to start monitoring your system immediately for suspicious fingerprints, Logsurfer is for you.