

Centralized Logging with *syslog-ng* and SEC

Leon Towns-von Stauber, Occam's Razor
Seattle Area System Administrators
Guild, October 2007

<http://www.occam.com/>



Contents

Introduction.....	2
syslog-ng.....	10
Simple Event Correlator.....	31
Support Tools.....	84
Future Activities.....	97

Introduction

- This talk describes an infrastructure that provides:
 - Aggregation of system logs from many UNIX hosts and other network devices
 - Partially-automated analysis of logged events
- This system replaced a set of host-based scripts that used simple regular expression matching to perform log reduction, emailing the remainder to administrators
 - Local scripts hard to keep updated, especially when some had host-specific differences
 - There was no correlation of multi-host events
 - Environment-wide events (like network scans) would result in floods of emails from many hosts

Introduction

- The benefits of centralized log aggregation and analysis include:
 - Log reduction and correlation reduce workload associated with viewing logs, making regular review feasible
 - Regular review of logs gives sysadmins better feel for computing environment, can spot anomalies more readily
 - Automated analysis and reporting provides early warning of unusual and possibly problematic events
 - Relaying log messages to a secure loghost makes them immune to tampering by local intruder, permits later forensic analysis

Legal Notices

- This presentation Copyright © 2007 Leon Towns-von Stauber. All rights reserved.
- Trademark notices
 - Sun™ and Solaris™ are trademarks of Sun Microsystems. See <http://www.sun.com/suntrademarks/>.
 - syslog-ng™ is a trademark of BalaBit IT Security. See <http://www.balabit.com/trademarks/>.
 - Other trademarks are the property of their respective owners.

Introduction - Logging Environment

- Loghost
 - Sun Enterprise 250
 - Dual 400-MHz UltraSPARC II CPUs
 - 1 GB RAM
 - 5 10/100BASE-T Ethernet interfaces
 - Dual 4-GB mirrored boot disks
 - Dual 18-GB mirrored disks for log data (`/var/log`)
 - Solaris 10
 - `syslog-ng` 1.6.8, SEC 2.3.1

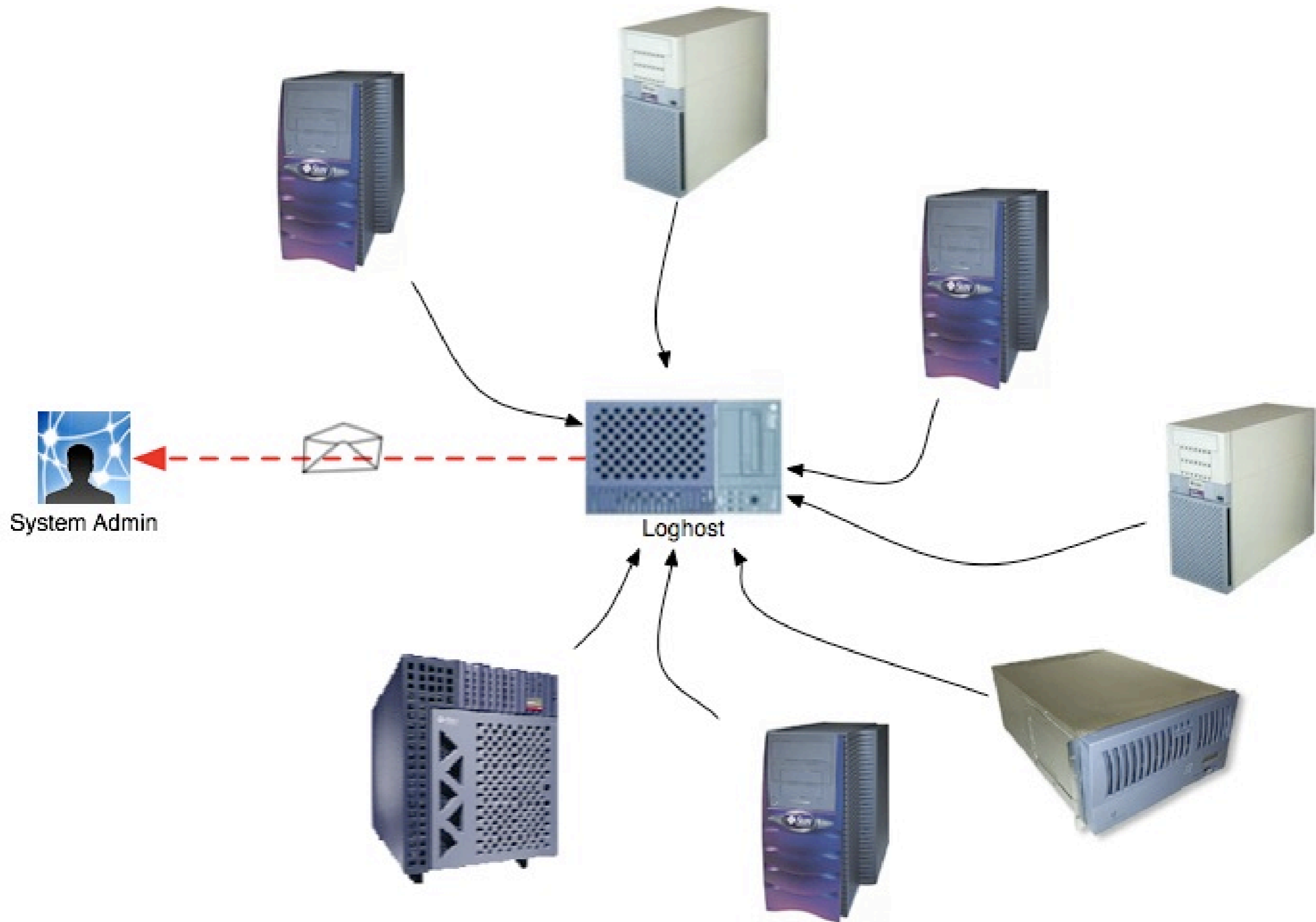
Introduction - Logging Environment

- Loghost
 - Networking
 - Interfaces placed on subnets with bulk of clients, to avoid dependency on router and/or firewall as much as possible
 - Placed in service September 2005
 - Dedicated only to logging, for security reasons

Introduction - Logging Environment

- Clients
 - About 70 now, started with closer to 100
 - Mostly Solaris (2.5.1 through 10)
 - Also 1 NetApp filer (started with 2), 2 Brocade switches
 - No new software for clients

Introduction - Logging Environment



Centralized Logging Environment

syslog-ng



syslog-ng - Intro

- syslog-ng is a replacement for UNIX `syslogd`, started by Balázs Scheider in 1998
- Now also offered in a commercial version by BalaBit
- <http://www.balabit.com/network-security/syslog-ng/>
- Syslog-ng for Solaris
 - http://www.softpanorama.org/Logs/Syslog_ng/
- Central Logging for Unix
 - <http://sial.org/talks/central-logging/>
- This talk is based on version 1.6.8
 - Current versions are 1.6.12 and 2.0.5

syslog-ng - Client Setup

- As mentioned earlier, clients continue to use stock `syslogd`
- They require two configuration changes
- `/etc/hosts`
 - Entry for `loghost` associated with `loghost`'s subnet-local IP address if one exists, or main address if not
 - Remember to remove the Solaris-default `loghost` alias to the host itself

syslog-ng - Client Setup

- `/etc/syslog.conf`
 - Send all logs to loghost
 - `*.debug @loghost`
 - Here's the full config file used on our Solaris hosts:

```
*.err;kern.notice;auth.notice           /dev/sysmsg
*.err;kern.debug;daemon.notice;mail.crit /var/adm/messages

*.alert;kern.err;daemon.err             operator
*.alert                                  root

*.emerg                                  *

*.debug                                  @loghost
```

syslog-ng - Server Setup

- I set up a dedicated account for use by syslog-ng
 - `syslog:x:514:514:Syslog-ng User:/var/log:`
 - Locked password
 - Group isn't used for anything
- After compiling, installed under `/opt/local/`
- This is on Solaris 10, so I created an SMF service to handle startup and shutdown
 - `/lib/svc/method/syslog-ng`
 - Startup script on next slide
 - `/var/svc/manifest/system/syslog-ng.xml`
 - Can provide SMF manifest upon request

syslog-ng - Server Setup

```
#!/sbin/sh

DAEMON=/opt/local/sbin/syslog-ng
USER=syslog
CONFFILE=/opt/local/etc/syslog-ng.conf
PIDFILE=/var/run/syslog-ng.pid

echo 'syslog-ng service starting.'

# Before syslogd starts, save any messages from previous crash dumps so that
# messages appear in chronological order.
/usr/bin/savecore -m
if [ -r /etc/dumpadm.conf ]; then
    . /etc/dumpadm.conf
    [ -n "$DUMPADM_DEVICE" -a "x$DUMPADM_DEVICE" != xswap ] && \
        /usr/bin/savecore -m -f $DUMPADM_DEVICE
fi

$DAEMON -u $USER -f $CONFFILE -p $PIDFILE
```

[/lib/svc/method/syslog-ng](#)

syslog-ng - Server Setup

- I compiled syslog-ng with support for TCP wrappers, and created an appropriate entry in `hosts.allow`, but it doesn't matter at this point
- The restrictions only apply to TCP connections, not UDP

syslog-ng - Server Setup

- All the log files are under `/var/log/`
 - The complete record for the day is `all`
 - The working file used by SEC for regular updates is `all_reduced.tmp`
 - This file goes away when a regular update is sent out
 - `syslog-ng-filtered` logs are in `byfac/` and `byapp/`
 - Some handy symlinks are in `bylnk/`, to help remember what the various `local` facilities (`local1`, `local2`, etc.) are used for
 - SEC-filtered logs are in `sec/`
 - Rotated log files are in `archive/`

syslog-ng - Config File

- Config file is `/opt/local/etc/syslog-ng.conf`
- The config file has 5 kinds of statements
 - General options
 - Sources and destinations
 - Filters
 - Log statements, where you direct messages from sources to destinations through filters
- I use this configuration for rough filtering and message routing, and to launch SEC processes for further, finer-grained parsing

syslog-ng - Config File

- Options

```
options {  
    group("is");  
    perm(0640);  
    create_dirs(yes);  
    dir_group("is");  
    dir_perm(0750);  
    chain_hostnames(no);  
    log_fifo_size(3000);  
};
```

- Setting group and permissions
- `create_dirs(yes)` - Create log directories on-the-fly
- `chain_hostnames(no)` - Record only the source host of a message
- `log_fifo_size(3000)` - Increase size of message buffer

syslog-ng - Config File

- Here's the first log statement

```
log { source(s_all); destination(d_all); };
```

- Every log message received by syslog-ng goes to the `d_all` destination (no filtering)

- Here's the source definition

```
source s_all {  
    internal();  
    sun-streams("/dev/log" door("/var/run/syslog_door"));  
    udp();  
};
```

- Messages are generated internally by syslog-ng, from the loghost itself, or from remote clients (via UDP)

syslog-ng - Config File

- Here's the first log statement

```
log { source(s_all); destination(d_all); };
```

- Here's the destination

```
destination d_all { file("/var/log/all"  
    template("$R_DATE $HOST $MSG\n")  
    template_escape(no));  
    program("/opt/local/bin/secStart main`"  
    template("$R_DATE $HOST $MSG\n")); };
```

- All messages are recorded in `/var/log/all`
- Messages are recorded with the time received, the source hostname, and the message content
- The timestamp included with the syslog message is rewritten, otherwise clients with inaccurate time or in different timezones confuse things

syslog-ng - Config File

- Here's the first log statement

```
log { source(s_all); destination(d_all); };
```

- Here's the destination

```
destination d_all { file("/var/log/all"  
    template("$R_DATE $HOST $MSG\n")  
    template_escape(no);  
    program("/opt/local/bin/secStart main`"  
    template("$R_DATE $HOST $MSG\n")); };
```

- In addition to specifying the log file, when this destination is set up it also runs the secStart script, which is used in spawning an SEC process to handle the same set of messages
- More on secStart later

syslog-ng - Config File

- The second `log` statement has a simple filter attached

```
log { source(s_all); filter(f_nona); destination(d_fac); };
```

- Here's the destination

```
destination d_fac      { file("/var/log/byfac/$FACILITY"  
                             template("$R_DATE $HOST $MSG\n")  
                             template_escape(no)); };
```

- Messages are automatically sorted into separate log files per syslog facility
- `/var/log/byfac/auth`, `/var/log/byfac/daemon`, `/var/log/byfac/kern`, `/var/log/byfac/local0`, etc.

syslog-ng - Config File

- The second `log` statement has a simple filter attached

```
log { source(s_all); filter(f_nona); destination(d_fac); };
```

- Here's the filter

```
filter f_nona { not(host("netapp1")); };
```

- Messages from NetApp filers are filtered out, because their use of facilities (and severities) is screwy

syslog-ng - Config File

- The third log statement

```
log { source(s_all); filter(f_emerg); destination(d_emerg); };
```

- Here's the destination

```
destination d_emerg { file("/var/log/byapp/emerg"  
    template("$R_DATE $HOST $MSG\n")  
    template_escape(no));  
    program("/opt/local/bin/secStart emerg`"  
    template("$R_DATE $HOST $MSG\n")); };
```

- Messages of **emerg** severity have dedicated SEC process to generate immediate email notifications

- Here's the filter

```
filter f_emerg { level(emerg) and not(host("netapp1")); };
```

syslog-ng - Config File

- The fourth `log` statement has a more complex filter, meant to funnel logged `su` and `sudo` commands into a separate file

```
log { source(s_all); filter(f_su); destination(d_su); };
```

- Here's the destination

```
destination d_su      { file("/var/log/byapp/su"
                          template("$R_DATE $HOST $MSG\n")
                          template_escape(no));
                      program("/opt/local/bin/secStart su`"
                              template("$R_DATE $HOST $MSG\n")); };
```

- Here's the filter

```
filter f_su          { (program("su") and not program("sudo")) or
                      (program("sudo") and
                       (match("sh$") or match("COMMAND=/*/su"))); };
```

- You can use keywords (such as `level` and `host` in the previous example, or `program` here) to construct

syslog-ng - Config File

- The fifth `log` statement introduces the `final` flag, meant to stop further processing if a message makes it through this filter, as the remaining `log` statements are for specific, non-overlapping purposes

```
log { source(s_all); filter(f_mem); destination(d_mem); flags(final); };
```

- The destination is similar to those before, logging to a file in `byapp/`, and generating a dedicated SEC process for immediate notifications (in this case, of memory errors)

```
destination d_mem { file("/var/log/byapp/memory"
    template("$R_DATE $HOST $MSG\n")
    template_escape(no));
    program("/opt/local/bin/secStart memory`"
    template("$R_DATE $HOST $MSG\n")); };
```

syslog-ng - Config File

- The fifth `log` statement introduces the `final` flag, meant to stop further processing if a message makes it through this filter, as the remaining `log` statements are for specific, non-overlapping purposes

```
log { source(s_all); filter(f_mem); destination(d_mem); flags(final); };
```

- Here's the filter

```
filter f_mem { program("SUNW,UltraSPARC") or  
  match("\\[AFT") or  
  (match("AFAR") and not match("SAFARI")) or  
  match("Fault_PC") or  
  match("Memory Module") or  
  match("Softerror") or  
  (program("unix") and match("remov") and match("age")); };
```

syslog-ng - Config File

- The remaining `log` statements are similar to those that came before

```
log { source(s_all); filter(f_scsi); destination(d_scsi); flags(final); };  
log { source(s_all); filter(f_md); destination(d_md); flags(final); };  
log { source(s_all); filter(f_netapp); destination(d_netapp); flags(final); };
```

- Two to catch SCSI and DiskSuite errors, and one to route NetApp logs to a separate file

syslog-ng - Performance

- Currently, this installation of syslog-ng handles about 150,000 log messages per weekday (~2 msgs/sec)
- It averaged less than half of that until 9/20/07, when one workstation started sending lots of print jobs to a server
 - Those print logs are now responsible for about 80,000 messages per day
- It used to average almost 300,000 messages per day, before some judicious changes to BIND logging
- At current rate, syslog-ng takes up less than 3 MB of RAM, using about 0.1% of CPU
- After logging a message to a file, syslog-ng hands off to SEC

Simple Event Correlator



- Simple Event Correlator (SEC) is written by Risto Vaarandi, and was first released in 2001
- <http://www.estpak.ee/~risto/sec/>
- Working with SEC
 - <http://sixshooter.v6.thrupoint.net/SEC-examples/article.html>
- It's essentially a 9000+-line Perl script used to automatically process log messages of any kind
 - Similar to Swatch, but much more sophisticated, and with that sophistication comes greater complexity
- This talk is based on version 2.3.1
 - Current version is 2.4.1

SEC - Intro

- Installed as `/opt/local/sbin/sec`
- Started by `syslog-ng` using `secStart` script
 - Argument to `secStart` specifies the SEC config file to use
 - `secStart` actually just prints the `sec` command line; `syslog-ng` executes it using backticks
 - `syslog-ng` directs log messages to `sec`'s standard input, which is accepted due to the `-input=-` argument

SEC - Intro

```
#!/bin/sh
#
# secStart - Print SEC command line with default options.
#

usage () {
    echo "usage:      $progname config

        'config' is the name of an SEC config file in /opt/local/
etc/sec/." >&2
    exit 2
}

progname=`basename $0`

[ $# -eq 1 ] || usage

echo "/opt/local/sbin/sec -conf=/opt/local/etc/sec/$1 -pid=/var/
run/sec-$1.pid -dump=/var/log/sec-$1.dump -debug=5 -syslog=local1
-intevents -input=-"
```

secStart

- `secStart` (cont'd.)
 - Why do this instead of running independent `sec` processes to monitor the log files themselves?
 - It was difficult to guarantee that messages wouldn't be missed, or wouldn't be parsed two or more times, when processes were restarted
 - This way, I know that every message received by `syslog-ng` is parsed exactly once each by the appropriate `sec` processes, and that all processes stop and start in sync

SEC - Configuration

- SEC config files are located in `/opt/local/etc/sec/`
- They can be located anywhere, as they're specified in the `sec` command line
- There's a `main` config (currently ~1400 lines) and several special-purpose configs (`emerg`, `md`, `memory`, `scsi`, and `su`, 20-80 lines apiece)
- Config files are kept under RCS

SEC - Configuration

- An SEC configuration is composed of multi-line stanzas, or rule definitions, with each line containing a key and value
- Keys include:
 - `type` - Type of rule (examples later)
 - `desc` - Textual description of rule
 - `ptype` - Type of pattern (typically `substr` or `regexp`)
 - `pattern` - String or Perl-style regular expression used to match log message
 - `context` - Apply rule only when named context in effect
 - `action` - What to do when rule is matched
 - `continue` - After this rule, continue (default) or stop

SEC - Configuration

- Rule types used in the examples
 - `suppress` - Simple rule to toss messages that match
 - `single` - If message matches, take immediate action
 - `singlewithsuppress` - If message matches, take immediate action, but then ignore similar messages for a time given by value of `window`
 - `singlewiththreshold` - Take action if the number of matching messages within a given `window` reaches a `threshold`
 - `pairwithwindow` - Specify 2 patterns; when 1st pattern matches, watch for 2nd pattern to appear within window; if it does, execute action; if not, execute different action

SEC - Configuration

- For each message, rules are processed one at a time, in order, until message is suppressed, matches a rule with `continue=dontcont`, or end-of-file is reached
- We'll start with a relatively simple configuration, one designed to alert on inappropriate uses of `su` and `sudo` (`/opt/local/etc/sec/su`)

SEC - Configuration Example: su

```
type=single
desc=Set log file and addressee list
ptype=substr
pattern=SEC_STARTUP
context=SEC_INTERNAL_EVENT
action=assign %f /var/log/sec/root_su;\
        assign %a sec-notify
```

- A rule like this appears at the top of each config file
- It matches internally-generated messages used by SEC to mark startup, and sets some variables for later use
 - %f - file in which to record parsed messages
 - %a - email address to which notifications are sent

SEC - Configuration Example: su

- The internal startup messages look like this:

```
Oct  5 00:01:39 loghost sec[13548]: Simple Event Correlator version 2.3.1
Oct  5 00:01:39 loghost sec[13548]: Reading configuration from /opt/local/etc/sec/su
Oct  5 00:01:39 loghost sec[13548]: Creating SEC internal context
'SEC_INTERNAL_EVENT'
Oct  5 00:01:39 loghost sec[13548]: Creating SEC internal event 'SEC_STARTUP'
Oct  5 00:01:39 loghost sec[13548]: Deleting SEC internal context
'SEC_INTERNAL_EVENT'
```

- More on contexts in a bit

SEC - Configuration Example: su

```
type=single
desc=su root by $4 $3 on $2 at $1
continue=dontcont
ptype=regex
pattern=(\w+\s+\d+\s+\d+:\d+:\d+) ([\w.-]+) su:.\+'su root\' (\w+) for (\w+)
action=pipe '' /bin/mailx -s "root su by $4 on $2" %a; write %f
```

- This rule looks for instances of `su` to root
- Elements of the message (timestamp, hostname, username, etc.) are captured in Perl regex backreferences (`$1`, `$2`, etc.), and used to generate the description
- The *event description* (the value of `desc`) is then piped through a mail command, and written out to a log file
- The event description is used by the `pipe` and `write` actions unless explicitly given other input
- **Example output:** `su root by user succeeded on host at Oct 5 04:41:11`

SEC - Configuration Example: su

```
type=single
desc=sudo $4 as root by $3 disallowed on $2 at $1
continue=dontcont
ptype=regex
pattern=(\w+\s+\d+\s+\d+:\d+:\d+) ([\w.-]+) sudo:.\s+(\w+) : command not allowed.
+USER=root.\s+(\w*sh)$
action=pipe ' /bin/mailx -s "root shell by $3 on $2" %a; write %f
```

- This rule looks for disallowed attempts to use sudo to run a root shell
- Works pretty much the same as the last one

```
type=suppress
desc=sudosh is OK
ptype=regex
pattern=sudo:.\s+COMMAND=\s+/opt\s+/local\s+/bin\s+/sudosh$
```

- This rule ignores successful uses of sudosh to run a root shell, as sudosh includes its own audit log

SEC - Configuration Example: su

```
type=single
desc=sudo $4 as root by $3 allowed on $2 at $1
continue=dontcont
ptype=regexp
pattern=(\w+\s+\d+\s+\d+:\d+:\d+) ([\w.-]+) sudo:.\s+(\w+) :.\s+USER=root.\s+\/(\w*sh)$
action=pipe ' /bin/mailx -s "root shell by $3 on $2" %a; write %f
```

- Final rule looks for successful uses of sudo to run a root shell
- Works pretty much the same as the last one

SEC - Configuration Example: `scsi`

- Now we'll take a look at a configuration that makes use of contexts (`/opt/local/etc/sec/scsi`)
- A context is a named state that can be set by a rule, which affects the processing of other rules until the context lifetime runs out, it's deleted by another rule, or the SEC process dies
- A context can also store a set of related messages
- First, as in the other example, setting up a variable for the notification email address

```
type=single
desc=Set addressee list
ptype=substr
pattern=SEC_STARTUP
context=SEC_INTERNAL_EVENT
action=assign %a sec-notify
```

SEC - Configuration Example: scsi

```
type=single
desc=$1 $2 $3 $5
continue=dontcont
ptype=regexp
pattern=(\w+\s+\d+\s+\d+:\d+:\d+) ([\w.-]+) (.+)( \[ID \d+ \w+\.\w+\] )?(.*)
action=add $2
context=$2
```

```
type=single
desc=$1 $2 $3 $5
continue=dontcont
ptype=regexp
pattern=(\w+\s+\d+\s+\d+:\d+:\d+) ([\w.-]+) (\w+:) (\[ID \d+ \w+\.\w+\])?.*(WARNING:
\/.+)
action=create $2 60 report $2 /bin/mailx -s "SCSI errors on $2" %a;\
    add $2
```

- The first rule has no effect until a context is set, so we have to look at the second rule first to make sense of this
- The second rule processes SCSI-related messages (filtered by syslog-ng) for one with a `WARNING:` string, which indicates the start of a set of SCSI error messages

SEC - Configuration Example: scsi

```
type=single
desc=$1 $2 $3 $5
continue=dontcont
ptype=regexp
pattern=(\w+\s+\d+\s+\d+:\d+:\d+) ([\w.-]+) (.+)( \[ID \d+ \w+\.\w+\] )?(.*)
action=add $2
context=$2
```

```
type=single
desc=$1 $2 $3 $5
continue=dontcont
ptype=regexp
pattern=(\w+\s+\d+\s+\d+:\d+:\d+) ([\w.-]+) (\w+:) (\[ID \d+ \w+\.\w+\])?.*(WARNING:
\/.+)
action=create $2 60 report $2 /bin/mailx -s "SCSI errors on $2" %a;\
    add $2
```

- When the second rule matches, it creates a context named after the hostname (\$2) which lasts for 60 seconds
- At end of that time, whatever's in the context's event store is reported via email
- This also adds the matched message to the event store

SEC - Configuration Example: scsi

```
type=single
desc=$1 $2 $3 $5
continue=dontcont
ptype=regexp
pattern=(\w+\s+\d+\s+\d+:\d+:\d+) ([\w.-]+) (.+)( \[ID \d+ \w+\.\w+\] )?(.*)
action=add $2
context=$2
```

```
type=single
desc=$1 $2 $3 $5
continue=dontcont
ptype=regexp
pattern=(\w+\s+\d+\s+\d+:\d+:\d+) ([\w.-]+) (\w+:) (\[ID \d+ \w+\.\w+\])?.*(WARNING:
\/.+)
action=create $2 60 report $2 /bin/mailx -s "SCSI errors on $2" %a;\
    add $2
```

- For the next message processed from the same host, if it's within the context lifetime, it'll match the first rule and be added to the context's event store
- Note that these rules also strip the clutter of the syslog ID added in later Solaris releases, by not including \$4 in desc

SEC - Configuration Example: `scsi`

```
type=single
desc=$1 $2 $3 $5
continue=dontcont
ptype=regexp
pattern=(\w+\s+\d+\s+\d+:\d+:\d+) ([\w.-]+) (.+)( \[ID \d+ \w+\.\w+\] )?(.*)
action=add $2
context=$2
```

```
type=single
desc=$1 $2 $3 $5
continue=dontcont
ptype=regexp
pattern=(\w+\s+\d+\s+\d+:\d+:\d+) ([\w.-]+) (\w+:) (\[ID \d+ \w+\.\w+\])?.*(WARNING:
\/.+)
action=create $2 60 report $2 /bin/mailx -s "SCSI errors on $2" %a;\
    add $2
```

- After 60 seconds of collecting SCSI logs from the same host (enough to chronicle one event), email is sent out

SEC - Configuration Example: scsi

- Example email notification:

Date: Thu, 30 Aug 2007 03:17:25 -0700 (PDT)

To: sec-notify@example.com

Subject: SCSI errors on host

Aug 30 03:16:23 host unix: WARNING: /sbus@1f,0/SUNW,fas@e,8800000/sd@5,0 (sd5):

Aug 30 03:16:23 host unix: Error for Command: write Error Level:

Retryable

Aug 30 03:16:23 host unix: Requested Block: 44 Error Block: 44

Aug 30 03:16:24 host unix: Vendor: SEAGATE Serial Number:

9744M60149

Aug 30 03:16:24 host unix: Sense Key: Hardware Error

Aug 30 03:16:24 host unix: ASC: 0x3 (peripheral device write fault), ASCQ:

0x0, FRU: 0x10

SEC - Configuration Example: `main`

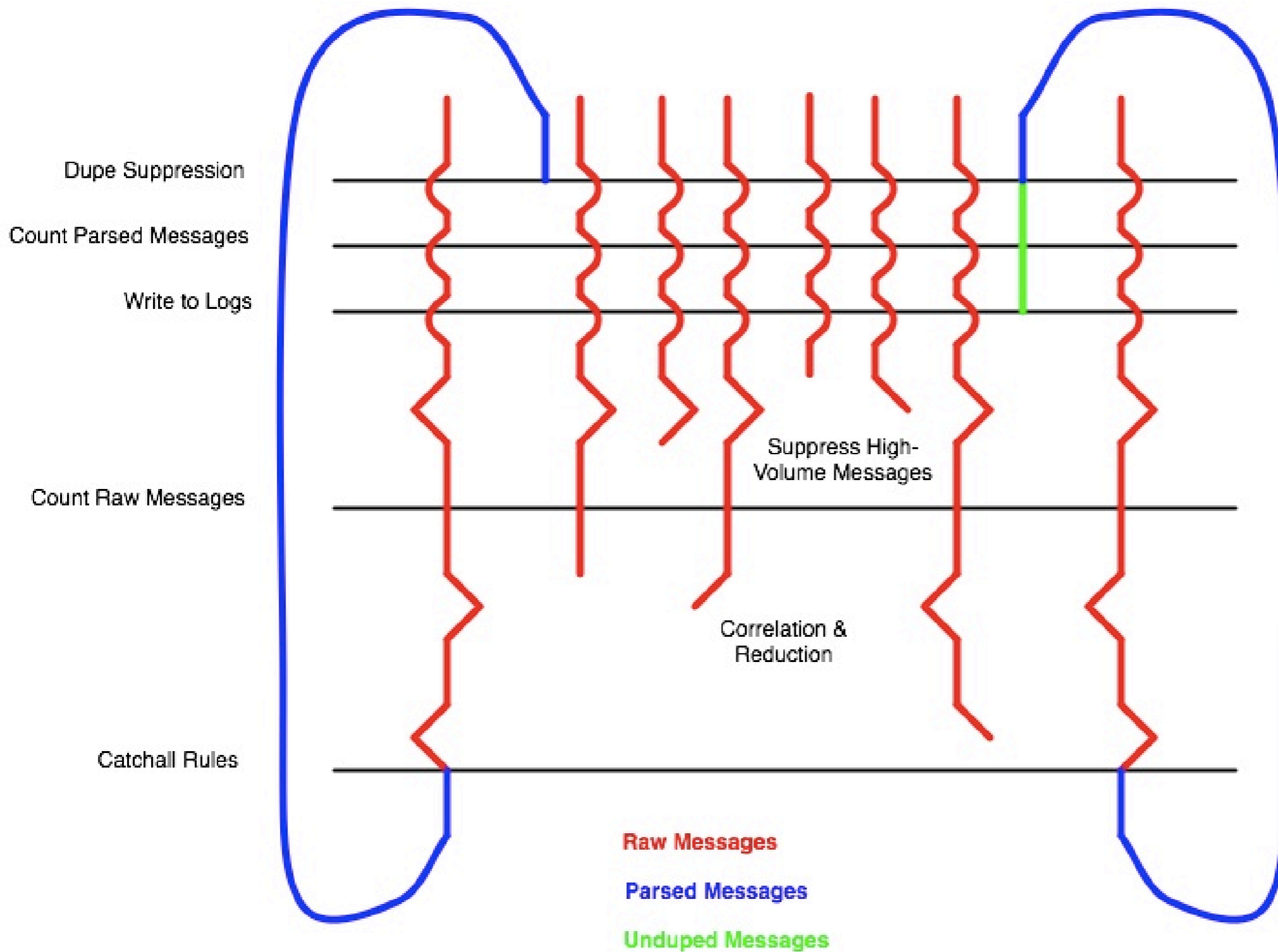
- Now onto the `main` configuration
 - This is a large file, so I'll choose a few excerpts
- First, some words about the overall structure
 - Most of the work of reducing and correlating logs is done in the vast middle of the file
 - At the end, any messages that ran the gauntlet are tagged (`PARSED:` is prepended to the message), and sent back through the rule set with an `event` action
 - This is done so that duplicate messages can be suppressed; near the beginning of the file is a rule that eliminates duplicates of `PARSED:` messages that show up within 5 minutes of each other

SEC - Configuration Example: main

- Structure (cont'd.)
 - The same rule that suppresses duplicates retags the remaining messages (prepending `UNDUPED:` to the message)
 - The tag is necessary so that log messages don't match the following rules their first time through
 - `UNDUPED:` messages are then counted in sliding time windows of 10 minutes; if the number reaches a threshold (currently 20), an email is sent immediately, as that logging volume may indicate a problem
 - A separate rule counts raw log messages (prior to most parsing), with a higher threshold (800 within 2 minutes)

SEC - Configuration Example: main

- Structure (cont'd.)
 - Finally, `UNDUPED`: messages are written out to log files (without the `UNDUPED`: tag)
 - Most Sendmail messages are written to `sec/misdirected_email`; their volume is so high, and their actionability so low, that they're written to a separate file and not counted as described earlier
 - All other messages go to `sec/all_reduced` and to `all_reduced.tmp` (used for periodic email reports)
 - Log messages written to these files are in standard `syslog` format, in case further processing is desired
- We'll see what these rules look like in a bit



Flow of Log Messages Through main Configuration

SEC - Configuration Example: main

- Here's how the file breaks down
 - **Setup rule** (1 rule, 15 lines)
 - Set variables for log pathnames, notification email
 - **Output rules** (10 rules, 89 lines)
 - Duplicate suppression, count of parsed messages, writing to log files
 - **Counting rules** (9 rules, 63 lines)
 - Suppress very high-volume messages (Samba file open/close, print logs, mailing list barrages), count the rest
 - **Heartbeat rule** (1 rule, 14 lines)
 - Keep track of hosts that send heartbeat log messages

SEC - Configuration Example: main

- Breakdown (cont'd.)
 - **Suppression rules** (99 rules, 487 lines)
 - Lots of basic suppress rules
 - **Correlation rules** (27 rules, 273 lines)
 - Correlate multiple messages from Brocade switches, Sendmail, OpenSSH, Samba, xntpd
 - **Boot rules** (20 rules, 176 lines)
 - Correlate messages from system boots
 - **Scan rules** (34 rules, 216 lines)
 - Correlate messages from network security scans

SEC - Configuration Example: main

- Breakdown (cont'd.)
 - **Password catch rules** (2 rules, 19 lines)
 - Look for cleartext passwords in logs
 - **Catchall rules** (5 rules, 39 lines)
 - Tag remaining messages and send them back through

SEC - Configuration Example: main

- For a detailed look, we'll start at the end of the file
- Here are some of the catchall rules to tag parsed messages

```
type=single
desc=Log messages w/o annoying message ID and priority, and w/o PID
continue=dontcont
ptype=regexp
pattern=(\w+\s+\d+\s+\d+:\d+:\d+ [\w.-]+ .+?)(\[\d+\])?: \[ID \d+ \w+\.\w+\] (.+)
action=event 0 PARSED:$1: $3
```

```
type=single
desc=Log messages w/o (usually) unhelpful PID
continue=dontcont
ptype=regexp
pattern=(\w+\s+\d+\s+\d+:\d+:\d+ [\w.-]+ .+?)\[\d+\]: (.+)
action=event 0 PARSED:$1: $2
```

```
type=single
desc=Log all remaining messages
continue=dontcont
ptype=regexp
pattern=.+
action=event 0 PARSED:$0
```

SEC - Configuration Example: main

- The catchall rules remove some elements that aren't usually helpful in evaluating importance (syslog message ID, process ID) with the use of Perl backrefs
- Then `event 0` puts the parsed and tagged message back into the message queue without delay
- Back toward the beginning of the file, the `PARSED:` messages are matched by the duplicate suppression rules

SEC - Configuration Example: main

```
# Suppress duplicates (those differing only in timestamp), then pass on
type=singlewithsuppress
desc=$2
continue=dontcont
ptype=regexp
pattern=^PARSED:(\w+\s+\d+\s+\d+:\d+:\d+) (.+)
action=event 0 UNDUPED:$1 $2
window=300
```

```
# In case something somehow gets to here...
type=singlewithsuppress
desc=Malformed message $1
continue=dontcont
ptype=regexp
pattern=^PARSED:(.+)
action=event 0 UNDUPED:%s
window=300
```

- The `singlewithsuppress` rule uses the value of `desc` to determine whether messages are “similar”
- Since the timestamp isn’t included in the event description, messages are compared only on content
- These rules prevent flooding by lots of similar messages

SEC - Configuration Example: main

- After de-duplication, the remaining messages are counted:

```
# Count parsed messages.
type=singlewiththreshold
desc=Over 20 interesting log messages received in the last 10 minutes.
continue=takenext
ptype=regexp
pattern=^UNDUPED:
action=create WARNED_OF_EXCESSIVE_INTERESTING_LOGS 1800;\
    pipe ' /bin/mailx -s "Excessive logging detected at %t" %a
context=!WARNED_OF_EXCESSIVE_INTERESTING_LOGS
window=600
thresh=20
```

- If at least 20 messages are counted in any 10-minute (600-second) period, an email is sent out
- A context is used to prevent such emails from being sent more than twice an hour
- When the threshold is tripped, the context prevents this rule from operating for 30 minutes (1800 seconds)

SEC - Configuration Example: main

- The next section is used to count raw messages, as they're first coming in

```
#####
```

```
# COUNTING RULES #
```

```
#####
```

```
# Get rid of the most verbose things early, both to prevent them from getting  
# processed by too many rules, and to keep them from skewing count of raw logs.
```

```
type=suppress
```

```
desc=Boring Samba stuff
```

```
ptype=regexp
```

```
pattern=smbserver smbd\[ \d+ \]: \[ID \d+ daemon\.notice\] \w+ (opened|closed) file
```

```
type=suppress
```

```
desc=Boring print stuff
```

```
ptype=regexp
```

```
pattern=\[ID \d+ lpr\.debug\]
```

```
...
```

SEC - Configuration Example: main

- The next section is used to count raw messages, as they're first coming in

...

```
# Count raw messages.
type=singlewiththreshold
desc=Over 800 log messages received in the last 2 minutes.
continue=takenext
ptype=regexp
pattern=.+
action=create WARNED_OF_EXCESSIVE_RAW_LOGS 600;\
    pipe ' /bin/mailx -s "Excessive logging detected at %t" %a
context=!WARNED_OF_EXCESSIVE_RAW_LOGS
window=120
thresh=800
```

- The idea is that heavy volume may indicate a problem (unauthorized scan, broken software, etc.), but be reflected in log messages that you would typically pay no attention to
- The email can spur you to investigate the raw logs

SEC - Configuration Example: main

- Syslog heartbeat
 - Every client has a cron job like this:

```
01,11,21,31,41,51 * * * * /bin/logger -p debug checking in...
```

- The following rule looks for those messages

```
# Heartbeat messages should be sent from each host every 10 minutes. If miss 2
# of them, notify.
type=single
desc=Haven't heard from $1 in over 20 minutes.
continue=dontcont
ptype=regexp
pattern=\w+\s+\d+\s+\d+:\d+:\d+ ([\w.-]+) (syslog:|root:|root: \[ID \d+ user\.debug
\]) checking in\.\.\.
action=create HEARTBEAT_$1 1260 pipe '' /bin/mailx -s "No contact from $1" %a
```

- Every time a heartbeat message is received, a 21-minute timer is started for that host

SEC - Configuration Example: main

- Syslog heartbeat (cont'd.)
 - The host-specific context timer is renewed each time a message is received
 - If the 21-minute timer expires, then at least two heartbeat messages in a row have been missed
 - 21 minutes allows for some variance on either side of a 20-minute window
 - When that happens, send an email

SEC - Configuration Example: main

- Example correlation, for Sendmail

```
type=pairwithwindow
desc=Invalid mail address
continue=dontcont
ptype=regexp
pattern=(\w+\s+\d+\s+\d+:\d+:\d+ mailhost sendmail)\[\d+\]: (\w+:) (<.+>?)\.\.\.
(User unknown|Invalid route address|Invalid host name|User address required|
Unbalanced \'<>')
action=event 0 PARSED:$1: Invalid address $3
desc2=Invalid mail address, extended
ptype2=regexp
pattern2=mailhost sendmail\[\d+\]: $2 (from=<.+?>).+(relay=.)+
action2=event 0 PARSED:%1: Invalid address %3 $1, $2
window=90
```

- Turns these raw sendmail logs:

```
Oct 11 14:02:46 mailhost sendmail[9535]: 19BL2jUH009535: <Joe.User>... User unknown
Oct 11 14:02:47 mailhost sendmail[9535]: 19BL2jUH009535:
from=<JANE.USER@EXAMPLE.COM>, size=78721, class=0, nrcpts=1,
msgid=<200710112102.19BL2jUH009535@example.com>, proto=SMTP, daemon=MTA,
relay=winserver [192.168.18.71]
```

SEC - Configuration Example: main

- Example correlation, for Sendmail

```
type=pairwithwindow
desc=Invalid mail address
continue=dontcont
ptype=regexp
pattern=(\w+\s+\d+\s+\d+:\d+:\d+ mailhost sendmail)\[\d+\]: (\w+:) (<.+>?)\.\.\.
(User unknown|Invalid route address|Invalid host name|User address required|
Unbalanced \'<')
action=event 0 PARSED:$1: Invalid address $3
desc2=Invalid mail address, extended
ptype2=regexp
pattern2=mailhost sendmail\[\d+\]: $2 (from=<.+?>).+(relay=.)
action2=event 0 PARSED:%1: Invalid address %3 $1, $2
window=90
```

- Into this:

```
Oct 11 14:02:46 mailhost sendmail: Invalid address <Joe.User>
from=<JANE.USER@EXAMPLE.COM>, relay=winserver [192.168.18.71]
```

SEC - Configuration Example: main

- Several other Sendmail correlations of that sort, followed by this:

```
# Suppress this after reducing mail errors, otherwise we can miss second message# of pair when there are multiple addressees and some are successful
type=suppress
desc=Successful email
ptype=regexp
pattern=sendmail.+msgid=
```

- We don't really care about successful emails, so those logs are suppressed
- But not until after the error messages have been correlated, otherwise this rule would have suppressed second log message in previous example

SEC - Configuration Example: main

- Example correlation, for OpenSSH

```
# Usually this is the result of a scan, but the first message comes right before  
# the message where the scanning host is identified, and thus before the scan  
# context is created. So what you often get is the setsockopt or aborted cxn  
# error, then the notification from the scan correlation rules.  
#
```

```
# Delay the SSH error for a few minutes, and avoid reprocessing on the way back  
# through the rules by making the context lifetime longer than the delay. By  
# then, either the scan context will have been entered and the message will be  
# dumped, or it won't and the message will be printed.
```

```
type=single
```

```
desc=Pre-scan SSH error
```

```
continue=dontcont
```

```
ptype=regexp
```

```
pattern=([\w.-]+) sshd.+error: (setsockopt SO_KEEPALIVE: Invalid argument|accept:  
Software caused connection abort)
```

```
action=create SAW_SSH_ERR_$1 210; event 200 $0
```

```
context=!SAW_SSH_ERR_$1
```

- This should get caught by the scan correlation rules, but often it shows up before the scan context is entered, since that doesn't happen until an authorized scanner is identified by a hostname or IP address in a message

SEC - Configuration Example: main

- Example correlation, for OpenSSH

```
# Usually this is the result of a scan, but the first message comes right before  
# the message where the scanning host is identified, and thus before the scan  
# context is created. So what you often get is the setsockopt or aborted cxn  
# error, then the notification from the scan correlation rules.  
#
```

```
# Delay the SSH error for a few minutes, and avoid reprocessing on the way back  
# through the rules by making the context lifetime longer than the delay. By  
# then, either the scan context will have been entered and the message will be  
# dumped, or it won't and the message will be printed.
```

```
type=single
```

```
desc=Pre-scan SSH error
```

```
continue=dontcont
```

```
ptype=regexp
```

```
pattern=([\w.-]+) sshd.+error: (setsockopt SO_KEEPALIVE: Invalid argument|accept:  
Software caused connection abort)
```

```
action=create SAW_SSH_ERR_$1 210; event 200 $0
```

```
context=!SAW_SSH_ERR_$1
```

- So we delay processing

- First stop further processing with `continue=dontcont`
- The event action sends it back through for re-processing

SEC - Configuration Example: main

- Example correlation, for OpenSSH

```
# Usually this is the result of a scan, but the first message comes right before  
# the message where the scanning host is identified, and thus before the scan  
# context is created. So what you often get is the setsockopt or aborted cxn  
# error, then the notification from the scan correlation rules.
```

```
#  
# Delay the SSH error for a few minutes, and avoid reprocessing on the way back  
# through the rules by making the context lifetime longer than the delay. By  
# then, either the scan context will have been entered and the message will be  
# dumped, or it won't and the message will be printed.
```

```
type=single
```

```
desc=Pre-scan SSH error
```

```
continue=dontcont
```

```
ptype=regexp
```

```
pattern=([\\w.-]+) sshd.+error: (setsockopt SO_KEEPALIVE: Invalid argument|accept:  
Software caused connection abort)
```

```
action=create SAW_SSH_ERR_$1 210; event 200 $0
```

```
context=!SAW_SSH_ERR_$1
```

- When this message hits this rule again 200 seconds later, it'll bypass it since the context lifetime of 210 seconds won't have run out yet

SEC - Configuration Example: main

- Example correlation, for OpenSSH

```
# Usually this is the result of a scan, but the first message comes right before  
# the message where the scanning host is identified, and thus before the scan  
# context is created. So what you often get is the setsockopt or aborted cxn  
# error, then the notification from the scan correlation rules.  
#
```

```
# Delay the SSH error for a few minutes, and avoid reprocessing on the way back  
# through the rules by making the context lifetime longer than the delay. By  
# then, either the scan context will have been entered and the message will be  
# dumped, or it won't and the message will be printed.
```

```
type=single
```

```
desc=Pre-scan SSH error
```

```
continue=dontcont
```

```
ptype=regex
```

```
pattern=( [\w.-]+ ) sshd.+error: (setsockopt SO_KEEPALIVE: Invalid argument|accept:  
Software caused connection abort)
```

```
action=create SAW_SSH_ERR_$1 210; event 200 $0
```

```
context=!SAW_SSH_ERR_$1
```

- By that time, either the scan context will be in effect and this message will be suppressed by the scan rules later in the file, or there is no scan and this message will make it to the reduced log file

SEC - Configuration Example: main

- Another example correlation, for OpenSSH

```
type=suppress
desc=SSH restart
ptype=regexp
pattern=\w+\s+\d+\s+\d+:\d+:\d+ ([\w.-]+) sshd\[\d+\]:
context=SSH_RESTART_$1

type=pairwithwindow
desc=SSH shutdown
continue=dontcont
ptype=regexp
pattern=(\w+\s+\d+\s+\d+:\d+:\d+) ([\w.-]+) sshd\[\d+\]:.+Received signal 15\;
terminating\.
action=event 0 PARSED:$1 $2 sshd: exiting on signal 15
desc2=SSH startup
ptype2=regexp
pattern2=\w+\s+\d+\s+\d+:\d+:\d+ $2 sshd\[\d+\]:.+Server listening on \S+ port 22\.
action2=create SSH_RESTART_%2 10; event 0 PARSED:%1 %2 sshd: restarted
window=10
```

- A two-rule correlation to reduce several sshd restart messages into one

SEC - Configuration Example: main

- Another example correlation, for OpenSSH
 - It turns these:

```
Jul 17 16:11:14 host sshd[302]: [ID 800047 auth.info] Received signal 15;  
terminating.
```

```
Jul 17 16:11:17 host sshd[317]: [ID 800047 auth.info] Server listening on :: port  
22.
```

```
Jul 17 16:11:17 host sshd[317]: [ID 800047 auth.info] Server listening on 0.0.0.0  
port 22.
```

- Into this:

```
Jul 17 16:11:14 host sshd: restarted
```

SEC - Configuration Example: main

- Correlation of boot logs

```
#####  
# BOOT RULES #  
#####  
  
type=single  
desc=Create bootup context  
continue=dontcont  
ptype=regexp  
pattern=(\w+\s+\d+\s+\d+:\d+:\d+) ([\w.-]+) (gen)?unix:..+(SunOS Release|Copyright  
1983-200\d Sun Microsystems)  
action=create BOOT_$2 600; event 0 PARSED:$1 $2 starting up...  
context=!BOOT_$2  
  
type=suppress  
desc=Bootup messages  
continue=dontcont  
ptype=regexp  
pattern=\w+\s+\d+\s+\d+:\d+:\d+ ([\w.-]+) \w+: \[ID \d+ kern\.(info|notice)\]  
context=BOOT_$1  
  
...
```

- First rule sets up boot context, rest are suppressions

SEC - Configuration Example: main

• Correlation of boot logs can turn this:

```
Oct 5 14:10:18 host sshd[12238]: [ID 800047 auth.info] Received signal 15; terminating.
Oct 5 14:10:20 host pseudo: [ID 129642 kern.info] pseudo-device: tod0
Oct 5 14:10:20 host genunix: [ID 936769 kern.info] tod0 is /pseudo/tod#0
Oct 5 14:10:21 host lpsched[321]: [ID 286429 lpr.debug] dispatch 8_SHUTDOWN
Oct 5 14:10:21 host lpsched[321]: [ID 889611 lpr.debug] s_shutdown(1)
Oct 5 14:10:21 host syslogd: going down on signal 15
Oct 5 14:12:45 host xntpd[3518]: [ID 866926 daemon.notice] xntpd exiting on signal 15
Oct 5 14:12:45 host genunix: [ID 672855 kern.notice] syncing file systems...
Oct 5 14:12:45 host genunix: [ID 904073 kern.notice] done
Oct 5 14:12:45 host genunix: [ID 540533 kern.notice] SunOS Release 5.8 Version Generic_117350-46 64-bit
Oct 5 14:12:45 host genunix: [ID 913632 kern.notice] Copyright 1983-2003 Sun Microsystems, Inc. All rights reserved.
Oct 5 14:12:45 host genunix: [ID 678236 kern.info] Ethernet address = 0:3:ba:44:a0:ae
Oct 5 14:12:45 host krtld: [ID 469452 kern.info] NOTICE: socall: 64-bit driver module not found
Oct 5 14:12:45 host unix: [ID 189951 kern.info] mem = 4194304K (0x10000000)
Oct 5 14:12:45 host unix: [ID 930857 kern.info] avail mem = 4117422080
Oct 5 14:12:45 host rootnex: [ID 466748 kern.info] root nexus = Sun Blade 2000/1000 (UltraSPARC-III+)
Oct 5 14:12:45 host rootnex: [ID 349649 kern.info] pcisch0 at root: SAFARI 0x8 0x700000
Oct 5 14:12:45 host genunix: [ID 936769 kern.info] pcisch0 is /pci#8,700000
Oct 5 14:12:45 host pcisch: [ID 370704 kern.info] PCI-device: ebus#5, ebus#0
Oct 5 14:12:45 host scsi: [ID 521012 kern.info] <SUNW2g cyl 14087 alt 2 hd 24 sec 42>
Oct 5 14:12:45 host genunix: [ID 936769 kern.info] xcalppm0 is /pci#8,700000/ebus#5/ppm#1,e
Oct 5 14:12:45 host rootnex: [ID 349649 kern.info] schppm0 at root: SAFARI 0x8 0x410050 ...
Oct 5 14:12:45 host rootnex: [ID 349649 kern.info] pcisch1 at root: SAFARI 0x8 0x600000
Oct 5 14:12:45 host genunix: [ID 936769 kern.info] pcisch1 is /pci#8,600000
Oct 5 14:12:45 host qlc: [ID 486054 kern.info] QLogic FCA Driver v0.40.5 (0): F/W version 2.01.112
Oct 5 14:12:45 host qlc: [ID 686097 kern.info] NOTICE: qlcic(0): loop ONLINE
Oct 5 14:12:45 host pcisch: [ID 370704 kern.info] PCI-device: SUNW,qlc#4, qlc#0
Oct 5 14:12:45 host genunix: [ID 936769 kern.info] qlc0 is /pci#8,600000/SUNW,qlc#4
Oct 5 14:12:45 host genunix: [ID 936769 kern.info] fp0 is /pci#8,600000/SUNW,qlc#4/fp#0,0
Oct 5 14:12:45 host scsi: [ID 799468 kern.info] ssd0 at fp0: name w2100000c504a0cc3,0, bus address e8
Oct 5 14:12:45 host genunix: [ID 936769 kern.info] ssd0 is /pci#8,600000/SUNW,qlc#4/fp#0,0/sad#w2100000c504a0cc3,0
Oct 5 14:12:45 host scsi: [ID 365881 kern.info] <SUNW2g cyl 14087 alt 2 hd 24 sec 42>
Oct 5 14:12:45 host genunix: [ID 408114 kern.info] /pci#8,600000/SUNW,qlc#4/fp#0,0/sad#w2100000c504a0cc3,0 (ssd0) online
Oct 5 14:12:45 host swageneric: [ID 308332 kern.info] root on /pci#8,600000/SUNW,qlc#4/fp#0,0/disk#w2100000c504a0cc3,0:a fstype ufs
Oct 5 14:12:45 host ebus: [ID 521012 kern.info] toddsl2870 at ebus0: offset 1,300070
Oct 5 14:12:45 host genunix: [ID 936769 kern.info] toddsl2870 is /pci#8,700000/ebus#5/rtc#1,300070
Oct 5 14:12:45 host rootnex: [ID 349649 kern.info] mc-us30 at root: SAFARI 0x0 0x400000 ...
Oct 5 14:12:45 host genunix: [ID 936769 kern.info] mc-us30 is /memory-controller#0,400000
Oct 5 14:12:45 host ebus: [ID 521012 kern.info] bbc_beeep0 at ebus0: offset 1,32
Oct 5 14:12:45 host genunix: [ID 936769 kern.info] bbc_beeep0 is /pci#8,700000/ebus#5/beeep#1,32
Oct 5 14:12:45 host pcisch: [ID 370704 kern.info] PCI-device: usb#5,3, ohci0
Oct 5 14:12:45 host genunix: [ID 936769 kern.info] ohci0 is /pci#8,700000/usb#5,3
Oct 5 14:12:45 host usba: [ID 464422 kern.warning] WARNING: /pci#8,700000/usb#5,3 (ohci0): connecting device on port 1 failed
Oct 5 14:12:45 host usba: [ID 950233 kern.info] USB-device: keyboard#4, hid2 at bus address 2
Oct 5 14:12:45 host genunix: [ID 936769 kern.info] hid2 is /pci#8,700000/usb#5,3/keyboard#4
Oct 5 14:12:45 host genunix: [ID 408114 kern.info] /pci#8,700000/usb#5,3/keyboard#4 (hid2) online
Oct 5 14:12:45 host pcisch: [ID 370704 kern.info] PCI-device: SUNW,XVR-100#1, pfb0
Oct 5 14:12:45 host genunix: [ID 936769 kern.info] pfb0 is /pci#8,700000/SUNW,XVR-100#1
Oct 5 14:12:45 host pfb: [ID 604756 kern.info] pfb#0: 1920x1200, rev 5159.0
Oct 5 14:12:45 host unix: [ID 270823 kern.info] opus: UltraSPARC-III+ (portid 0 impl 0x15 ver 0xb0 clock 1200 MHz)
Oct 5 14:12:45 host ebus: [ID 521012 kern.info] se0 at ebus0: offset 1,400000
Oct 5 14:12:45 host genunix: [ID 936769 kern.info] se0 is /pci#8,700000/ebus#5/serial#1,400000
Oct 5 14:12:45 host pcisch: [ID 370704 kern.info] PCI-device: firewall#5,2, hcil3940
Oct 5 14:12:45 host genunix: [ID 936769 kern.info] hcil3940 is /pci#8,700000/firewire#5,2
Oct 5 14:12:45 host pcisch: [ID 370704 kern.info] PCI-device: network#5,1, eri0
Oct 5 14:12:45 host genunix: [ID 936769 kern.info] eri0 is /pci#8,700000/network#5,1
Oct 5 14:12:45 host genunix: [ID 454863 kern.info] dump on /dev/dsk/c0t1d0s1 size 20004 MB
Oct 5 14:12:45 host eri: [ID 517527 kern.info] SUNW,eri0 : 100 Mbps half duplex link up
Oct 5 14:12:45 host pseudo: [ID 129642 kern.info] pseudo-device: devinfo0
Oct 5 14:12:45 host genunix: [ID 936769 kern.info] devinfo0 is /pseudo/devinfo#0
Oct 5 14:12:45 host scsi: [ID 365881 kern.info] /pci#8,700000/scsi#6 (glm0):
Oct 5 14:12:45 host scsi: [ID 365881 kern.info] /pci#8,700000/scsi#6 (glm0):
Oct 5 14:12:45 host 7 Symbios 53c875 found.
Oct 5 14:12:45 host pcisch: [ID 370704 kern.info] PCI-device: scsi#6, glm0
Oct 5 14:12:45 host genunix: [ID 936769 kern.info] glm0 is /pci#8,700000/scsi#6
Oct 5 14:12:45 host scsi: [ID 365881 kern.info] /pci#8,700000/scsi#6,1 (glm1):
Oct 5 14:12:45 host supports power management.
Oct 5 14:12:45 host scsi: [ID 365881 kern.info] /pci#8,700000/scsi#6,1 (glm1):
Oct 5 14:12:45 host 7 Symbios 53c875 found.
Oct 5 14:12:45 host pcisch: [ID 370704 kern.info] PCI-device: scsi#6,1, glm1
Oct 5 14:12:45 host genunix: [ID 936769 kern.info] glm1 is /pci#8,700000/scsi#6,1
Oct 5 14:12:45 host ebus: [ID 521012 kern.info] scm1z0 at ebus0: offset 0,40
Oct 5 14:12:45 host genunix: [ID 936769 kern.info] scm1z0 is /pci#8,700000/ebus#5/i2c#1,30/card-reader#0,40
Oct 5 14:12:45 host pseudo: [ID 129642 kern.info] pseudo-device: fssnap0
Oct 5 14:12:45 host genunix: [ID 936769 kern.info] fssnap0 is /pseudo/fssnap#0
Oct 5 14:12:45 host pseudo: [ID 129642 kern.info] pseudo-device: winlock0
Oct 5 14:12:45 host genunix: [ID 936769 kern.info] winlock0 is /pseudo/winlock#0
Oct 5 14:12:45 host pseudo: [ID 129642 kern.info] pseudo-device: lockstat0
Oct 5 14:12:45 host genunix: [ID 936769 kern.info] lockstat0 is /pseudo/lockstat#0
Oct 5 14:12:45 host pseudo: [ID 129642 kern.info] pseudo-device: vol0
Oct 5 14:12:45 host genunix: [ID 936769 kern.info] vol0 is /pseudo/vol#0
Oct 5 14:12:45 host rootnex: [ID 349649 kern.info] upa64#0 at root: SAFARI 0x8 0x480000
Oct 5 14:12:45 host pseudo: [ID 129642 kern.info] pseudo-device: llc10
Oct 5 14:12:45 host genunix: [ID 936769 kern.info] llc10 is /pseudo/llc1#0
Oct 5 14:12:45 host ebus: [ID 521012 kern.info] audioc#0 at ebus0: offset 1,200000
Oct 5 14:12:45 host genunix: [ID 936769 kern.info] audioc#0 is /pci#8,700000/ebus#5/audio#1,200000
Oct 5 14:12:45 host pseudo: [ID 129642 kern.info] pseudo-device: pm0
Oct 5 14:12:45 host genunix: [ID 936769 kern.info] pm0 is /pseudo/pm#0
Oct 5 14:12:45 host pseudo: [ID 129642 kern.info] pseudo-device: tod0
Oct 5 14:12:45 host genunix: [ID 936769 kern.info] tod0 is /pseudo/tod#0
Oct 5 14:12:45 host pseudo: [ID 129642 kern.info] pseudo-device: lofi0
Oct 5 14:12:45 host pseudo: [ID 129642 kern.info] pseudo-device: fcp0
Oct 5 14:12:45 host genunix: [ID 936769 kern.info] fcp0 is /pseudo/fcp#0
Oct 5 14:12:45 host ebus: [ID 521012 kern.info] gpio_873170 at ebus0: offset 1,300600
Oct 5 14:12:45 host genunix: [ID 936769 kern.info] gpio_873170 is /pci#8,700000/ebus#5/gpio#1,300600
Oct 5 14:12:45 host pseudo: [ID 129642 kern.info] pseudo-device: rsm0
Oct 5 14:12:45 host genunix: [ID 936769 kern.info] rsm0 is /pseudo/rsm#0
Oct 5 14:12:45 host fcip: [ID 962561 kern.info] fcip attach for port instance (0x0) successful
Oct 5 14:12:45 host ebus: [ID 521012 kern.info] gpio_873170 at ebus0: offset 1,300600
Oct 5 14:12:45 host genunix: [ID 936769 kern.info] gpio_873170 is /pci#8,700000/ebus#5/gpio#1,300600
Oct 5 14:12:45 host pseudo: [ID 129642 kern.info] pseudo-device: ram0
Oct 5 14:12:45 host genunix: [ID 936769 kern.info] ram0 is /pseudo/ram#0
Oct 5 14:12:45 host ebus: [ID 521012 kern.info] ecpp0 at ebus0: offset 1,300278
Oct 5 14:12:45 host genunix: [ID 936769 kern.info] ecpp0 is /pci#8,700000/ebus#5/parallel#1,300278
Oct 5 14:12:45 host devfsadm[58]: [ID 742014 daemon.error] RCM notification failed: error_code: 14
Oct 5 14:12:45 host su: [ID 366847 auth.info] 'su sys' succeeded for root on /dev/console
Oct 5 14:12:45 host eri: [ID 517527 kern.info] SUNW,eri0 : No response from Ethernet network : Link down -- cable problem?
Oct 5 14:12:45 host eri: [ID 517527 kern.info] SUNW,eri0 : 100 Mbps full duplex link up
Oct 5 14:12:45 host /usr/lib/nfs/lockd[201]: [ID 599441 daemon.info] Number of servers not specified. Using default of 20.
Oct 5 14:12:45 host ntpdate[227]: [ID 558275 daemon.notice] adjust time server 192.168.50.155 offset -0.327033 sec
Oct 5 14:12:45 host lpsched[249]: [ID 851279 lpr.debug] file descriptor resource limit is 4096 (-2042 printers)
Oct 5 14:12:46 host printd[265]: [ID 979083 lpr.debug] get_lock(/var/spool/print/.printd.lock, 1)
Oct 5 14:12:46 host printd[265]: [ID 854461 lpr.debug] get_lock(/var/spool/print/.printd.lock, 1) - have lock
Oct 5 14:12:46 host printd[265]: [ID 166668 lpr.debug] job_list_append(0x0, NULL, NULL)
Oct 5 14:12:46 host printd[265]: [ID 697681 lpr.debug] daemon exiting...
Oct 5 14:12:46 host lpsched[254]: [ID 467870 lpr.debug] schedule(EV_ALARM)
Oct 5 14:12:46 host lpsched[254]: [ID 467870 lpr.debug] schedule(EV_INTERRUPT)
Oct 5 14:12:46 host lpsched[254]: [ID 467870 lpr.debug] schedule(EV_MOTIFY)
Oct 5 14:12:46 host lpsched[254]: [ID 467870 lpr.debug] schedule(EV_SLOWP)
Oct 5 14:12:48 host xntpd[325]: [ID 702911 daemon.notice] xntpd 3-5.93c Mon Sep 20 15:47:11 PDT 1999 (1)
Oct 5 14:12:48 host xntpd[325]: [ID 301315 daemon.notice] tickadj = 5, tick = 10000, tvu_maxalew = 495, est. hz = 100
Oct 5 14:12:48 host xntpd[325]: [ID 182907 daemon.info] precision = 8 usec
Oct 5 14:12:48 host xntpd[325]: [ID 798731 daemon.notice] using kernel phase-lock loop 0041
Oct 5 14:12:48 host xntpd[325]: [ID 677929 daemon.info] read drift of 11.803 from /var/ntp/ntp.drift
Oct 5 14:12:48 host xntpd[325]: [ID 798731 daemon.notice] using kernel phase-lock loop 0041
Oct 5 14:12:48 host xntpd[325]: [ID 988144 daemon.debug] signal_no_reset: signal 18 had flags 20000
Oct 5 14:12:48 host lpstat[329]: [ID 761841 lpr.debug] database: printers, services: user files nis nisplus xfn
Oct 5 14:12:48 host lpstat[329]: [ID 761841 lpr.debug] database: printers, services: user files nis nisplus xfn
Oct 5 14:12:48 host lpstat[329]: [ID 761841 lpr.debug] database: printers, services: user files nis nisplus xfn
Oct 5 14:12:48 host lpstat[329]: [ID 761841 lpr.debug] database: printers, services: user files nis nisplus xfn
Oct 5 14:12:49 host lpget[325]: [ID 761841 lpr.debug] database: printers, services: user files nis nisplus xfn
Oct 5 14:12:49 host lpget[325]: [ID 761841 lpr.debug] database: printers, services: user files nis nisplus xfn
Oct 5 14:12:49 host sshd[335]: [ID 800047 auth.info] Server listening on :: port 22.
Oct 5 14:12:49 host sshd[335]: [ID 800047 auth.info] Server listening on 0.0.0.0 port 22.
Oct 5 14:12:50 host lpget[345]: [ID 761841 lpr.debug] database: printers, services: user files nis nisplus xfn
Oct 5 14:12:50 host lpget[345]: [ID 761841 lpr.debug] database: printers, services: user files nis nisplus xfn
Oct 5 14:12:50 host lpget[352]: [ID 761841 lpr.debug] database: printers, services: user files nis nisplus xfn
Oct 5 14:12:50 host lpget[355]: [ID 761841 lpr.debug] database: printers, services: user files nis nisplus xfn
Oct 5 14:12:50 host lpget[358]: [ID 761841 lpr.debug] database: printers, services: user files nis nisplus xfn
Oct 5 14:12:51 host pcisch: [ID 370704 kern.info] PCI-device: SUNW,XVR-100#1, pfb0
Oct 5 14:12:51 host genunix: [ID 936769 kern.info] pfb0 is /pci#8,700000/SUNW,XVR-100#1
Oct 5 14:12:51 host pfb: [ID 604756 kern.info] pfb#0: 1920x1200, rev 5159.0
Oct 5 14:13:47 host pseudo: [ID 129642 kern.info] pseudo-device: devinfo0
Oct 5 14:13:47 host genunix: [ID 936769 kern.info] devinfo0 is /pseudo/devinfo#0
Oct 5 14:14:00 host ebus: [ID 521012 kern.info] scm1z0 at ebus0: offset 0,40
Oct 5 14:14:00 host genunix: [ID 936769 kern.info] scm1z0 is /pci#8,700000/ebus#5/i2c#1,30/card-reader#0,40
Oct 5 14:14:00 host pseudo: [ID 129642 kern.info] pseudo-device: fssnap0
Oct 5 14:14:00 host genunix: [ID 936769 kern.info] fssnap0 is /pseudo/fssnap#0
Oct 5 14:14:00 host pseudo: [ID 129642 kern.info] pseudo-device: lockstat0
Oct 5 14:14:00 host genunix: [ID 936769 kern.info] lockstat0 is /pseudo/lockstat#0
Oct 5 14:14:00 host pseudo: [ID 129642 kern.info] pseudo-device: vol0
Oct 5 14:14:00 host genunix: [ID 936769 kern.info] vol0 is /pseudo/vol#0
Oct 5 14:14:00 host pseudo: [ID 129642 kern.info] pseudo-device: llc10
Oct 5 14:14:00 host genunix: [ID 936769 kern.info] llc10 is /pseudo/llc1#0
Oct 5 14:14:01 host ebus: [ID 521012 kern.info] audioc#0 at ebus0: offset 1,200000
Oct 5 14:14:01 host genunix: [ID 936769 kern.info] audioc#0 is /pci#8,700000/ebus#5/audio#1,200000
Oct 5 14:14:01 host pseudo: [ID 129642 kern.info] pseudo-device: tod0
Oct 5 14:14:01 host genunix: [ID 936769 kern.info] tod0 is /pseudo/tod#0
Oct 5 14:14:01 host pseudo: [ID 129642 kern.info] pseudo-device: lofi0
Oct 5 14:14:01 host genunix: [ID 936769 kern.info] lofi0 is /pseudo/lofi#0
Oct 5 14:14:01 host fcip: [ID 962561 kern.info] fcip attach for port instance (0x0) successful
Oct 5 14:14:01 host ebus: [ID 521012 kern.info] gpio_873170 at ebus0: offset 1,300600
Oct 5 14:14:01 host genunix: [ID 936769 kern.info] gpio_873170 is /pci#8,700000/ebus#5/gpio#1,300600
Oct 5 14:14:01 host pseudo: [ID 129642 kern.info] pseudo-device: ram0
Oct 5 14:14:01 host genunix: [ID 936769 kern.info] ram0 is /pseudo/rsm#0
Oct 5 14:14:01 host ebus: [ID 521012 kern.info] ecpp0 at ebus0: offset 1,300278
Oct 5 14:14:01 host genunix: [ID 936769 kern.info] ecpp0 is /pci#8,700000/ebus#5/parallel#1,300278
Oct 5 14:14:02 host pseudo: [ID 129642 kern.info] pseudo-device: devinfo0
Oct 5 14:14:02 host genunix: [ID 936769 kern.info] devinfo0 is /pseudo/devinfo#0
Oct 5 14:17:38 host xntpd[325]: [ID 854739 daemon.info] synchronized to 149.59.50.155, stratum=2
Oct 5 14:18:08 host pcisch: [ID 370704 kern.info] PCI-device: SUNW,XVR-100#1, pfb0
Oct 5 14:18:08 host genunix: [ID 936769 kern.info] pfb0 is /pci#8,700000/SUNW,XVR-100#1
Oct 5 14:18:08 host pfb: [ID 604756 kern.info] pfb#0: 1920x1200, rev 5159.0
```

SEC - Configuration Example: main

- Into this:

```
Oct  5 14:10:21 host syslogd: going down on signal 15
Oct  5 14:10:18 host sshd: exiting on signal 15
Oct  5 14:12:45 host xntpd: xntpd exiting on signal 15
Oct  5 14:12:45 host genunix: syncing file systems...
Oct  5 14:12:45 host genunix:  done
Oct  5 14:12:45 host starting up...
Oct  5 14:12:45 host usba: WARNING: /pci@8,700000/usb@5,3 (ohci0): connecting device
on port 1 failed
Oct  5 14:12:45 host devfsadmd: RCM notification failed: error_code: 14
```

- :-)

SEC - Configuration Example: main

- Correlation of scan logs

```
# Following are several pairs of rules. The 2nd rule sets up the scanning
# contexts; one host-specific, one generic. The host-specific context is used to
# extend the context lifetimes and to suppress host-specific messages (via the
# 1st rule), and the generic context is used to suppress non-host-specific
# messages (via the rules above).
```

```
type=single
desc=$1 scanhost extending scan context lifetime by 25 minutes...
continue=dontcont
ptype=regexp
pattern=(\w+\s+\d+\s+\d+:\d+:\d+) .+(sendmail|sshd|named|inet|rpcbind).+(scanhost|
192\.168\.73\.28)
action=set SCAN_scanhost 1500; set SCAN 1500; event 0 PARSED:%s
context=SCAN_scanhost
```

```
type=single
desc=$1 scanhost scan in progress...
continue=dontcont
ptype=regexp
pattern=(\w+\s+\d+\s+\d+:\d+:\d+) .+(sendmail|sshd|named|inet|rpcbind).+(scanhost|
192\.168\.73\.28)
action=create SCAN_scanhost 1500; create SCAN 1500; event 0 PARSED:%s
```

- Appearance of authorized scanner sets off context creation

SEC - Configuration Example: main

- Correlation of scan logs
 - Examples of suppress rules, which come before host-specific rules shown in previous slide

```
type=suppress
ptype=regex
pattern=(sendmail|sshd|inetd).+error: accept: Protocol error
context=SCAN
```

```
type=suppress
ptype=regex
pattern=(sendmail|sshd|smbd).+accept: Software caused connection abort
context=SCAN
```

```
type=suppress
ptype=regex
pattern=sendmail.+accepting connections again for daemon MTA
context=SCAN
```

```
type=suppress
ptype=regex
pattern=sshd.+error: setsockopt SO_KEEPALIVE: (Invalid argument|Connection reset by
peer|Broken pipe)
context=SCAN
```

SEC - Configuration Example: main

- Correlation of scan logs
 - This goes after scan rules, as seeing these can help identify when a scan is taking place, but otherwise we don't care about them

```
# Suppress after catching scan (or Samba startup), otherwise we can miss it.  
type=suppress  
desc=Boring Samba stuff  
ptype=regex  
pattern=(smbd|nmbd|winbindd)\[\d+\]: \[ID \d+ daemon\.notice\]
```

SEC - Conclusion

- By far, the bulk of the setup work is creating the log filters
 - The process is iterative
 - Let logs through, figure out what you don't care to see, create filters to suppress or correlate those messages
 - Repeat until volume is bearable
 - Here's a record of changes to the `main` config
 - Averaged 9.6 changes per month in first year (including very big changes)
 - Averaged 1.6 changes per month in second year
- Missing important log messages is bad
 - But so is having so many to look at that you ignore them

SEC - Conclusion

- How effective is the log reduction and correlation at highlighting anomalous events?
- Let's look at how many messages make it to the regular reports from `all_reduced`
- Earlier this year (6/07)
 - Averaged 480,000 total messages per week
 - Reduced to 480 messages per week (0.1% of total)
- Current (with extra volume from print logs)
 - Averaging 980,000 total messages per week
 - Reduced to 700 messages per week (0.07% of total)

SEC - Conclusion

- What is the drain on system resources imposed by SEC?
- As stated earlier, current volume is ~150k msgs/day
 - Each message is processed at least once by SEC, often multiple times
 - Many messages are held in memory due to contexts, pairwithwindow rules, etc.
- At current rate
 - Smaller processes (emerg, md, memory, scsi, su), plus their associated secStart processes, take about 8 MB of RAM and less than 0.2% of CPU
 - The main process takes about 9 MB of RAM and around 1% - 1.5% of CPU

Support Tools



Support Tools - Custom Scripts

- `secStart` (described earlier)
- I use this alias as a convenient way to restart `syslog-ng` (and all the SEC processes) in SMF
 - `alias relog sudo svcadm restart syslog-ng`

Support Tools - Custom Scripts

- `sendLogs`
 - cron calls `sendLogs` to issue regular reports of anomalous events (i.e., the contents of `all_reduced`)

```
0 0,6-18 * * 1-5 /opt/local/bin/sendLogs
0 0,6,12,18 * * 0,6 /opt/local/bin/sendLogs
# Temporary holiday schedule
#0 0,6,12,18 * * * /opt/local/bin/sendLogs
```

- Hourly during work hours (6 AM - 6 PM weekdays), every six hours otherwise
- Example email

```
Date: Wed, 13 Jun 2007 08:00:01 -0700 (PDT)
To: log-report@example.com
Subject: Interesting logs Wed Jun 13 08:00:00 PDT 2007
```

```
Jun 13 07:36:44 host xntpd: too many recvbufs allocated (30)
Jun 13 07:50:00 smbserver dfullcheck: /home is 98% full
```

Support Tools - Custom Scripts

```
#!/bin/sh
#
# sendLogs - Email accumulated reduced logs to admins.
#

PATH=/usr/bin

LOG_FILE=/var/log/all_reduced
ADDRESSEES="log-report"

if [ -s ${LOG_FILE}.tmp ]; then
    mv ${LOG_FILE}.tmp ${LOG_FILE}.$$
    grep -v sudo: ${LOG_FILE}.$$ 1>/dev/null && grep -v sudo:
${LOG_FILE}.$$ \
        | mailx -s "Interesting logs `date`" $ADDRESSEES
    grep sudo: ${LOG_FILE}.$$ 1>/dev/null && grep sudo: $
{LOG_FILE}.$$ \
        | mailx -s "Non-admin sudo `date`" $ADDRESSEES
    rm ${LOG_FILE}.$$
fi
```

sendLogs

Support Tools - Custom Scripts

- `wbRestart`
 - I use this rule in the `main` config to notice when `winbindd` (the Samba daemon that binds to Active Directory to provide authentication) has lost its connection to AD

```
# Create flag file to restart winbindd on atlantis when AD cxn is lost.
type=singlewiththreshold
desc=Samba winbind connection to Active Directory lost.
continue=takenext
ptype=regexp
pattern=smbserver winbindd\[d+\]: \[ID d+ daemon\.error\] cli_nt_setup_creds:
request challenge failed
action=write %w %s
window=600
thresh=2
```

- When the signature message is matched (at least twice in 10 minutes), the rule creates a file that serves as a semaphore (`/var/log/sec/restart_winbindd`)

Support Tools - Custom Scripts

- `wbRestart`

- This cron job runs on an admin server with root SSH access to other machines

```
13,33,53 * * * * /opt/local/bin/wbRestart >/dev/null
```

- The `wbRestart` script checks for the existence of the semaphore file on the loghost, and if it's there, restarts `winbindd` on the Samba server to make it reestablish its AD connection

Support Tools - Custom Scripts

```
#!/bin/sh
#
# wbRestart - Restart winbindd when AD cxn lost.
#

PATH=/usr/bin

SSH="/opt/local/bin/ssh -q -o ConnectTimeout=10"
FLAG_FILE=/var/log/sec/restart_winbindd
SAMBA_SERVER=smbserver
SAMBA_ROOT=/opt/local/samba

if $SSH loghost ls $FLAG_FILE 1>/dev/null 2>&1; then
    $SSH $SAMBA_SERVER $SAMBA_ROOT/bin/smbcontrol winbindd
shutdown
    sleep 5
    $SSH $SAMBA_SERVER $SAMBA_ROOT/sbin/winbindd
    logger -p daemon.error -t wbRestart Restarted winbindd on
$SAMBA_SERVER
    $SSH loghost rm $FLAG_FILE
fi
```

wbRestart

Support Tools - logadm

- I was just about to install `logrotate`, when I discovered that `logadm` has been included with Solaris since version 9
- Not quite as flexible as `logrotate`, and the config file is a little harder to understand, but certainly good enough
- Configured in `/etc/logadm.conf`
- Can be manually edited, or via `logadm` commands

Support Tools - logadm

- Key to example `logadm.conf` lines
 - `-c` - Retain this many old copies (0 for unlimited)
 - `-N` - Don't complain about missing log files
 - `-c` - Rotate by copying file then truncating to zero length
 - `-p` - Rotate this often
 - `-P` - Time of last rotation, in UTC (automatically updated)
 - `-t` - Name of rotated file (including macros)
 - `-z` - Compress rotated files with `gzip`, keeping this many uncompressed (doesn't seem to work properly)
 - `-a` - Execute this command after rotation

Support Tools - logadm

- Here's what I added directly to `logadm.conf`

```
/var/log/sec/* -C 0 -N -p 1w -t '/var/log/archive/sec/%Y/$basename.%F' -z 0
/var/log/byapp/* -C 30 -N -c -p 1w -t '/var/log/archive/byapp/$basename.%F' -z 0
/var/log/byfac/* -C 5 -N -c -p 1w -t '/var/log/archive/byfac/$basename.%F' -z 0
/var/log/all -C 0 -P 'Thu Oct 11 07:01:01 2007' -a '/usr/sbin/svcadm restart syslog-
ng' -p 1d -t /var/log/archive/all.%Y-%m/all.%F -z 0
```

- All log files in `sec/`, `byapp/`, and `byfac/` are rotated weekly to `archive/`, and gzipped
- Files from `sec/` are rotated to `archive/sec/YYYY/` `filename.YYYY-MM-DD.gz`, and are never removed
- Files from `byapp/` are rotated to `archive/byapp/` `filename.YYYY-MM-DD.gz`; 30 old files are retained
- Files from `byfac/` are rotated to `archive/byfac/` `filename.YYYY-MM-DD.gz`; 5 old files are retained

Support Tools - logadm

- Here's what I added directly to `logadm.conf`

```
/var/log/sec/* -C 0 -N -p 1w -t '/var/log/archive/sec/%Y/$basename.%F' -z 0
/var/log/byapp/* -C 30 -N -c -p 1w -t '/var/log/archive/byapp/$basename.%F' -z 0
/var/log/byfac/* -C 5 -N -c -p 1w -t '/var/log/archive/byfac/$basename.%F' -z 0
/var/log/all -C 0 -P 'Thu Oct 11 07:01:01 2007' -a '/usr/sbin/svcadm restart syslog-
ng' -p 1d -t /var/log/archive/all.%Y-%m/all.%F -z 0
```

- Finally, `/var/log/all` is rotated daily to `archive/all.YYYY-MM/all.YYYY-MM-DD.gz`, and retained indefinitely
 - After that (the last log file to be rotated), `syslog-ng` is restarted, along with the SEC processes
- After more than two years saving every log message from every system, archived logs take up just over 1 GB of disk
- With 15 GB more available, at this rate there's room for another couple of decades before cleanup is necessary

Support Tools - logadm

- logadm keeps track of when to next rotate a log file by making changes to logadm.conf
- Here's what logadm dynamically added to logadm.conf

```
/var/log/sec/all_reduced -P 'Fri Oct  5 07:01:00 2007'  
/var/log/sec/mem_errors -P 'Fri Oct  5 07:01:00 2007'  
/var/log/sec/misdirected_email -P 'Fri Oct  5 07:01:00 2007'  
/var/log/sec/root_su -P 'Fri Oct  5 07:01:00 2007'  
/var/log/byapp/disksuite -P 'Fri Oct  5 07:01:00 2007'  
/var/log/byapp/memory -P 'Fri Oct  5 07:01:00 2007'  
/var/log/byapp/netapp -P 'Fri Oct  5 07:01:00 2007'  
/var/log/byapp/scsi -P 'Fri Oct  5 07:01:00 2007'  
/var/log/byapp/su -P 'Fri Oct  5 07:01:00 2007'  
/var/log/byfac/auth -P 'Fri Oct  5 07:01:00 2007'  
/var/log/byfac/b -P 'Fri Oct  5 07:01:00 2007'  
/var/log/byfac/daemon -P 'Fri Oct  5 07:01:00 2007'  
/var/log/byfac/kern -P 'Fri Oct  5 07:01:00 2007'  
/var/log/byfac/local0 -P 'Fri Oct  5 07:01:00 2007'  
/var/log/byfac/local1 -P 'Fri Oct  5 07:01:00 2007'  
/var/log/byfac/local2 -P 'Fri Oct  5 07:01:00 2007'  
/var/log/byfac/local3 -P 'Fri Oct  5 07:01:00 2007'  
/var/log/byfac/local4 -P 'Fri Oct  5 07:01:00 2007'  
/var/log/byfac/local6 -P 'Fri Oct  5 07:01:00 2007'  
...
```

Support Tools - logadm

- Gotcha: logadm **always** works in UTC (unlike cron)
- Ignores time zones
- Timestamps generated for rotated files are in UTC
- Example: Tried running logadm cron job at 23:58, to divide logs easily by whole days
- However, rotated logs for 3/8/2006 would be named with a timestamp of 2006-03-09, since logadm thought it was 07:58 of the next day
- Eventually rescheduled cron job for 12:01 AM, and just keep in mind that archived log files are dated a day late
- Notice how the rotation times on the previous slide are at 7:01 AM (PDT being 7 hours behind UTC)?

Future Activities



Future

- Ideas for future activities
- Accept logs from other platforms (Linux, HP-UX, AIX)
 - Server will handle it
 - Requires up-front work of trimming logs for normal events; a few weeks of daily time commitment
- Change transport protocol from UDP to TCP
 - Could enable SSL/TLS encryption and TCP wrappers
 - Server load would increase by unknown amount
 - Requires replacing `syslogd` with `syslog-ng` on all clients
- Script to generate SEC configs from simpler templates

Centralized Logging with *syslog-ng* and SEC

Leon Towns-von Stauber, Occam's Razor
Seattle Area System Administrators
Guild, October 2007

<http://www.occam.com/>

