LogJAMM: Logs are Just Another Monitoring Mechanism

SAND2007-2691C

Centrally logging Linux, UNIX and Windows machines

June 11, 2007

Paul Sery

System Administrator

The acknowledgement statement MUST be used on the title slide of all presentation material distributed outside of Sandia.

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

National Nuclear Security Administration

Sandia National Laboratories

Taking a slightly different approach to centralized logging

- Do the normal things
  - transmit logs to central location
  - archive the logs
  - analyze the logs
  - use for forensics when necessary
- But add the system administrator to the mix
  - they're most familiar with their systems
  - our human intrusion detection system (IDS)
  - many eyes instead of a few
- Why do this?

Is the read bit set on /var/log files?

- We don't expect users to read their logs
- But system administrators should
- Do they?
- I don’t think I want to know the answer
- Example: our client SSL certificate expired:
  - client can't connect to server
  - it tries to reconnect every 10 seconds
  - thousands of log messages piled up on clients
  - my email address was in each message!
- I didn't get a single complaint or query

Is the read bit set for /var/log files?

- This is not really a surprising result
- reading logs is a boring, mind numbing process
- easy to forget
- easy to ignore
- easy to miss important events in the clutter

What's the solution?

- IMHO there is no solution
- There’s no silver bullet so we need to do what's possible:
  - gather data
  - provide better tools
  - encourage sys admins to read and analyze their data
  - incrementally improve out capabilities with time & experience
- Thus, a central logging system:
  - centralize log data collection
  - make it easier to read logs
  - make it easier to analyze logs
  - create automated analysis systems
  - research advanced techniques

Central Logging System overview

- The remaining slides describe the
  - system architecture overview
  - query and analysis system
  - future

System architecture

- Simple client-server configuration
  - similar to examples found at www.campin.net/newlogcheck.html
- Clients sends encrypted log data to server
  - optionally send unencrypted via TCP or UDP
- Server collects and stores data
- Analysis machine
  - downloads from server(s)
  - crunches data
  - makes available to system administrators

Query and Analysis System

- Designed for System Administrators
  - System administrators have lots of jobs to do and little time to perform them
  - our job is to make their job easier
  - help them to consistently and thoroughly review logs
- To do this:
  - remove need to log into individual machines
  - remove need to sift through repetitive log data
  - create tools for finding new and different events
- Use system administrators as our de-facto IDS

System Administrator tools

- Accessed through a web page
- Provides query/analysis tools
- Access limited to the designated sys admin

Raw and Normalized Logs

- Raw logs contain all information
- Homogenized logs have variable information stripped
  - replace numbers with a token
  - replace random identifiers with a token

Query Tools

- Search raw or normalized logs
- Use:
  - regular expressions
  - self-generated indexes (tokens)
  - Fourier Transform-based filters
  - to find differences and new, never before seen events
  - local versus remote log differences
- Optionally e-mail normalized logs, diffs, new stuff
- Optionally e-mail real-time events

Raw Logs

- Use the web page to look at your raw logs
  - don't have to interactively log into each machine,
  - or write your own scripts
  - display by machine, date, facility and priority
  - you can also do text searches
- One-stop-shopping makes reading logs easier

Raw log example

- Mar 26 04:10:01 xyz crond[13283]: (root) CMD (/usr/lib/sa/sa1 1 1)
- Mar 26 04:20:01 xyz crond[13295]: (root) CMD (/usr/lib/sa/sa1 1 1)
- Mar 26 04:22:01 xyz crond[13298]: (root) CMD (run-parts /etc/cron.weekly)
- Mar 26 04:24:45 xyz anacron[1766]: Updated timestamp for job `cron.weekly' to 2006-03-26
- Mar 26 04:30:01 xyz crond[1770]: (root) CMD (/usr/lib/sa/sa1 1 1)
- Mar 26 04:40:01 xyz crond[1774]: (root) CMD (/usr/lib/sa/sa1 1 1)
- Mar 26 04:50:01 xyz crond[1778]: (root) CMD (/usr/lib/sa/sa1 1 1)
- Mar 26 05:00:01 xyz crond[1858]: (root) CMD (/usr/lib/sa/sa1 1 1)
- Mar 26 05:01:01 xyz crond[1881]: (root) CMD (run-parts /etc/cron.hourly)
- Mar 26 05:10:01 xyz crond[1993]: (root) CMD (/usr/lib/sa/sa1 1 1)
- Mar 26 05:20:01 xyz crond[1997]: (root) CMD (/usr/lib/sa/sa1 1 1)
- Mar 26 05:30:01 xyz crond[2001]: (root) CMD (/usr/lib/sa/sa1 1 1)
- Mar 26 05:40:01 xyz crond[2006]: (root) CMD (/usr/lib/sa/sa1 1 1)
- Mar 26 05:50:01 xyz crond[2010]: (root) CMD (/usr/lib/sa/sa1 1 1)
- Mar 26 06:00:01 xyz crond[2024]: (root) CMD (/usr/lib/sa/sa1 1 1)
- Mar 26 06:01:01 xyz crond[2027]: (root) CMD (run-parts /etc/cron.hourly)
- Mar 26 06:10:01 xyz crond[2030]: (root) CMD (/usr/lib/sa/sa1 1 1)
- Mar 26 06:20:02 xyz crond[2034]: (root) CMD (/usr/lib/sa/sa1 1 1)

- and on, and on, page after page

Homogenized Logs

- Log messages contains static and variable information
  - static: program name, message header, etc.
  - variable: IP addresses, PIDS, etc.
- Removing variables reduces visual “noise”
  - normalized logs are much easier to scan
  - unusual events tend to stand out
- Normal computer logs can be tens to millions of lines per day
- Homogenizing drastically reduces total number of messages
- Several orders of magnitude in some cases
- Many messages are exactly alike once variables stripped

Homogenized Log Example 1

- Count normalized message
- 1 anacron: Updated timestamp for job cron.daily to ...-...-...
- 171 crond(pam_unix)[...]: session closed for user root
- 170 crond(pam_unix)[...]: session opened for user root by (uid=...)
- 144 crond: (root) CMD (/usr/lib/sa/sa... ... ...)
- 1 crond: (root) CMD (run-parts /etc/cron.daily)
- 23 crond: (root) CMD (run-parts /etc/cron.hourly)
- 341 crond: pam_krb...[...]: ccache dir: /tmp
- 341 crond: pam_krb...[...]: configured realm dce.sandia.gov
- 170 crond: pam_krb...[...]: no v... creds for user root, skipping session cleanup
- 170 crond: pam_krb...[...]: no v... creds for user root, skipping session setup
- 170 crond: pam_krb...[...]: pam_close_session returning ... (Success)
- 170 crond: pam_krb...[...]: pam_open_session returning ... (Success)
- 341 crond: pam_krb...[...]: renewable lifetime: ...
- 341 crond: pam_krb...[...]: ticket lifetime: ...
- 341 crond: pam_krb...[...]:...: /etc/v...srvtab
- 341 crond: pam_krb...[...]:...: forwardable not proxiable
- 341 crond: pam_krb...[...]:...: Kerberos ...
- 341 crond: pam_krb...[...]:...: no ignore_afs
- 341 crond: pam_krb...[...]:...: no krb..._convert
- 341 crond: pam_krb...[...]:...: user_check
- 341 crond: pam_krb...[...]:...: validate
- 341 crond: pam_krb...[...]:...: warn
- 5 rhnsd: running program /usr/sbin/rhn_check
- 1 sendmail:...: to=root, ctladdr=root (.../...), delay=...:...:..., xdelay=...:...:..., mailer=local, pri=..., dsn=....., stat=Sent
- 14 ssl: ... connected from .....:...
- 9 ssl: VERIFY OK: depth=...,
- 14 syslog-ng[...]: Connection broken to AF_INET(.....:...), reopening in ... seconds

Homogenized versus Raw Logs

- Example 1 summarized over 4K messages
- But contained very little interesting information
- Many cron, sendmail, etc. jobs but not much else
- Must sift through a lot of clutter to determine that the logs aren't interesting
- How do you see a tree for the forest?

Normalized log example 2

- 185 crond(pam_unix)[...]: session closed for user root
- 184 crond(pam_unix)[...]: session opened for user root by (uid=...)
- 131 crond: (root) CMD (/usr/lib/sa/sa... ... ...)
- 1 crond: (root) CMD (/usr/local/sbin/CRONJOBS/normalize_messages.pl)
- 1 crond: (root) CMD (/usr/local/sbin/CRONJOBS/xfer_logs....sh)
- 21 crond: (root) CMD (/usr/sbin/ntpdate -s IPADDR > /dev/null)
- 3 crond: (root) CMD (/usr/sbin/up...date -u > /dev/null)
- 22 crond: (root) CMD (run-parts /etc/cron.hourly)
- 150 crond: pam_krb...[...]: ccache dir: /tmp
- 149 crond: pam_krb...[...]: configured realm dce.sandia.gov
- 184 crond: pam_krb...[...]: no v... creds for user root, skipping session cleanup
- 184 crond: pam_krb...[...]: no v... creds for user root, skipping session setup
- 74 crond: pam_krb...[...]: pam_close_session returning ... (Success)
- 74 crond: pam_krb...[...]: pam_open_session returning ... (Success)
- ...
- 1 httpd: [error] [client IP_ADDRESS] script not found or unable to stat: /var/www/cgi-bin/bash
- 1 httpd: [error] [client IP_ADDRESS] script not found or unable to stat: /var/www/cgi-bin/csh
- 1 httpd: [error] [client IP_ADDRESS] script not found or unable to stat: /var/www/cgi-bin/finger
- 1 httpd: [error] [client IP_ADDRESS] script not found or unable to stat: /var/www/cgi-bin/hamcards.cgi
- 1 httpd: [error] [client IP_ADDRESS] script not found or unable to stat: /var/www/cgi-bin/maillist.pl
- 1 httpd: [error] [client IP_ADDRESS] script not found or unable to stat: /var/www/cgi-bin/noN-exIstAnt_scriPt.lp
- 1 httpd: [error] [client IP_ADDRESS] script not found or unable to stat: /var/www/cgi-bin/nph-test-cgi
- 1 httpd: [error] [client IP_ADDRESS] script not found or unable to stat: /var/www/cgi-bin/postcard.pl

Normalized log example 2

- Example 2 clearly shows some interesting events
- Notice the script execution attempts!
- This is “script kiddie” stuff commonly seen by Internet-facing web servers
- But this web server isn't facing the Internet!
- Turned out to be a standard security scan
- Spotting such events against the noisy background of “raw” log events is difficult

Self-generated index searches

- Problem: how to find similar messages
  - traditional solution: string or regex search
  - but, to some extent, you first need to know what you're looking for
  - how do you search for “something like xyz”?
- Possible solution: use self-generated index searches
  - create numeric indexes (tokens) from the log message
  - convert words and phrases into numbers
  - search for phrases numerically “close” to each other

Self-generated index searches

- Convert each letter into its ASCII equivalent
- For instance, abc = 304 and xyz = 373
  - “I know my abc's by heart” = 2119
  - “I know my xyz's by heart” = 2188
  - The sentences differ numerically by about 3%
- Use the indexes to search for similar sentences
- Using regular expressions to do this is harder
- Normalized messages generated by the same program tend to group together
  - for instance, failed ssh logins look like:
  - **sshd: Failed password for bob from ::ffff: IPADDR port ... ssh...**
  - use the index concept to search for similar events
  - varying the index by 5% reveals messages like:
  - **sshd: Failed password for bobz from ::ffff: IPADDR port ... ssh...**

Self-generated index searches

- Generate numeric indexes for the program name
  - the first word in the message
  - and the remaining words
- sshd: Failed password for bob from ::ffff: IP_ADDRESS port ... ssh...
  - *sshd:* is the program index
  - *Failed* is the first word index
  - *password for bob ...* is the remainder index
  - resulting indexes: 492, 581, 4813
- Using numeric indexes helps find both exact and similar events in the database
- Self-generated index searches
- Note: index searches of complex messages start to break down
  - “I know my abc's by heart” = 2119
  - “I don't know my xyz's very well” = 2835
  - you might need to correlate such messages, but they differ by over 30%
  - so far, it appears that 20% is the effective limit

Fourier Transform searches

- Think of a log message as a string of letters
  - Convert the letters to their ASCII values
  - Message can now be treated as a one-dimensional array (signal)
  - Transform from pattern space to frequency domain using Fast Fourier Transform (FFT)
  - Each message has a unique FFT signature
  - Filter coefficients to search for similar messages
- Next step, use 2-d FFTs
  - 1st dimension: Convert each word in a message into an index (see previous slides)
  - 2nd dimension: one client per day
  - We'll get one signature per machine per day
  - Possible alerts for unusual machine aggregate activity
- What else?
  - Investigate other time periods, combinations
  - Use super computers
  - Investigate other transforms
- Truly useful? Stay tuned

Daily homogenized log e-mails

- Schedule daily e-mails of normalized logs
- Have your cup of coffee while reviewing yesterday's “summarized” logs
- Quickly see what happened on your machines
- Simple, easily performed job
- We've been using it for over a year
- We find it to be a very effective tool

Real-time alerts

- Use the Simple Event Correlator (SEC) system
- SEC provides real-time syslog pattern matching that:
  - searches individual events for patterns
  - searches sequence of events for patterns
  - specifies action to take when pattern matched
- To configure SEC:
  - enter patterns for your machines via web page
  - interactively search results or receive e-mails

Local versus Remote Log Comparisons

- Local and remote logs should match up
- Four reasons they won't:
  - network trouble – logs don't get to the server
  - race conditions – SSL latency drops initial messages
  - server trouble – no server to save to
  - hacker wipes logs!
- All reasons are interesting, but the last is what we're looking for

Simple log comparison

- Checksum yesterday's local logs
- Transmit checksum to server via syslog-ng
- Server compares to its copy vs. client checksums
- Works but with a fair number of mismatches
  - reasons #1-3 (from previous slide) can cause mismatch
- Right now too many false-positives
- Perhaps better syslog configuration will work better
  - filter out SSL messages
  - filter out non-locally logged messages such as rhnsd

Direct Log Comparisons

- Copy yesterday's logs to server
  - server directly compares local vs remote
  - differences stored in db and mailed to sysadmin
  - small problems like SSL start-up latency are obvious
  - so are big problems
- More difficult to setup than checksum method
  - need cron job or daemon to transmit logs
  - use more network resources
  - most useful for secure servers
  - voluntary for now

New, never before seen events

- Use normalized log db and self-generated indexes
- Compare today's events to all previous ones
- Record, display e-mail new events
- Hopefully, interesting trends and interesting events will be evident

Future

- Anomaly detection
  - use the Sisyphus machine-learning system
  - use virus/spyware detection model (Cyber Defenders)
  - obtain syslog intrusion database
  - create normalized signatures
  - compare to incoming logs in real-time
- Want to increase the number of clients
  - the more data, the better
- Investigate other analysis tools