Lustre Log Analyzer

Kalpak Shah
DataDirect Networks
Lustre Log Analysis Requirements

► Need scripts to parse Lustre debug logs
  • Only way to effectively use the logs for debugging
► No structured approach to analyzing and understanding Lustre logs
► Event correlation between multiple clients and Lustre servers can be very difficult
► MDS / OSS RPC Tracing capabilities
► Event notifications based on configurable triggers
► High level of expertise and Lustre knowledge needed to use logs effectively
► Lustre logs can grow to millions of lines across servers – sometimes in GBs
Who needs this?

► Developers
  • A handy tool to understand and analyze Lustre logs would be most beneficial for new and experienced developers

► Support
  • Once logs are received from customer, support would like to identify suspicious issues quickly
  • Need ability to search and index large logs quickly and effectively

► Administrators
  • While debugging specific issues onsite, admins would like to monitor the logs (easily) and get alerts
Some options today

► **Ilanalyze**
  - Perl script to indent and color code logs
  - Extract specific subsystem log lines
  - Show RPC patterns
  - Limited use – and not maintained or improved anymore

► **Simple Event Correlator**
  - Useful for event log monitoring
  - Does not index the logs so searching is slow and ineffective

► **Splunk**
  - Good GUI for log management and analysis
  - Indexes log data so can be searched
  - Commercial tool – unlikely to be used by developers and support
Some specific log analysis needs

► Toggle subsystems on / off
  • mdc, mds, osc, ost, obdclass, obdfilter, llite, ptlrpc, portals, Ind, Idlm, lov

► Analyze logs across multiple servers in a single view

► RPC debugging

► Search for specific strings / errors / warnings

► View logs across nodes filtered between certain timestamps

► Ability to add custom logic / searches

► Email alerts based on certain conditions
Logstash

- Logstash is a tool for managing events and logs. You can use it to collect logs, parse them, and store them for later use.
- It helps if you take logs and other event data from your systems and move it into a central place.
- Using Elasticsearch as a backend datastore - Logstash acts as the workhorse, creating a powerful pipeline for storing, querying and analyzing your logs.
- It has built-in inputs, filters, codecs and outputs, you can harness some powerful functionality with a small amount of effort.
Logstash Components – Log Collection Pipeline

► Input

• Inputs are the mechanism for passing log data to Logstash.
  • file: reads from a file on the filesystem
  • syslog: listens on the well-known port 514 for syslog messages and parses it

► Filters

• Filters are used as intermediary processing devices in the Logstash chain. They are often combined with conditionals in order to perform a certain action on an event, if it matches particular criteria
  • drop: drop an event completely, for example, debug events
  • clone: make a copy of an event, possibly adding or removing fields.

► Output

• Outputs are the final phase of the Logstash pipeline
  • elasticsearch: save your data in an efficient, convenient and easily queryable format
  • file: writes event data to a file on disk.
Logstash Configuration for Lustre logs

input {
    file { path => "/tmp/lctl/debug_kernel" }
}

filter {
    lustre_debug { }
}

output {
    elasticsearch { host => elasticsearch-server }
}

Custom parsing logic to parse lctl debug_kernel logs. It is a Ruby file.
Elasticsearch

► Elasticsearch is a flexible and powerful open source, distributed, real-time search and analytics engine.

► Elasticsearch gives you the ability to move easily beyond simple full-text search

► Provides scalability to our log indexing and storage capabilities
We now have the data in Elasticsearch, but looking at JSON documents would be worse than Lustre logs.

Kibana is an open source (Apache Licensed), browser based analytics and search dashboard for ElasticSearch.

It has a very nice interface to build graphs, charts and much more based on data stored in an Elasticsearch index.

Let’s see how we can use it for our Lustre log analysis requirements.
Email alerts can be configured in Logstash using their conditionals configuration.

Use regular expression searches which will be used by Logstash while parsing logs to trigger alerts.
Searching for logs of a particular subsystem across nodes

In Logstash filtering, we can replace these subsystem codes with names.

Change Time Frame & auto-update window

Search Any Text
Graphical view for your text searches

► Can be useful to determine log spikes across timeline
View actual logs & select columns

Viewing actual logs from multiple text search & select columns to view
Multiple search terms

- Logs across various oss nodes with multiple search
Search by timestamp & text filter

Ex: search for all ldlm logs with LDLM_ENQUEUE event in last 3 mins on 2 specific nodes
Search for specific transaction ID across logs of multiple nodes
Custom graphs

- Custom graphs can be added and configured to suit requirements
- For example, below custom graph shows variance in number of logs across every 5 mins
- If variance is high, it indicates that number of log lines has jumped by 1366% across two 5 minute intervals - which could indicate a problem
Multiple searches with column filtering

 ► For example show me logs of debug_mark 0x80 which contains logs for function class_process*
Questions?
Installing Logstash

► Prerequisites
  • Java

► Installing Logstash
  • $ curl -O https://download.elasticsearch.org/logstash/logstash-1.4.0.tar.gz
  • $ tar zxvf logstash-1.4.0.tar.gz
  • $ cd logstash-1.4.0

► Lets try out
  • $ bin/logstash -e 'input { stdin { } } output { stdout {} }'
    o hello world
    o 2013-11-21T01:22:14.405+0000 0.0.0.0 hello world
Installing Elasticsearch

$ curl -O https://download.elasticsearch.org/elasticsearch/elasticsearch/elasticsearch-1.0.1.tar.gz

$ tar zxfv elasticsearch-1.0.1.tar.gz

$ cd elasticsearch-1.0.1/

. $ /bin/elasticsearch