GrayLog for Java developers

Track Monitoring & Cloud

José Manuel Ortega
@jmortegac
Agenda

- Introduction to graylog
- Docker image & compose
- Graylog Architecture
- Connecting with Java
- Connecting with other services
GrayLog

Open Source Log Management

http://www.graylog.org/
http://docs.graylog.org/
Graylog features

- Graylog is an open source logs monitor capable of handling messages from different sources:
- Application servers: IBM Websphere, Weblogic, Jboss
- Framework Applications: JAVA EE, NodeJS, Python, C#
- Web Servers: Nginx, Apache
Install

- Debian / Ubuntu (deb package)
- RedHat / CentOS (RPM package)
- Virtual Machine (OVA / Vagrant)
- Config management (Chef / Puppet / Ansible)
- **Docker** image && docker compose
The Graylog operating system package repository. Please read the documentation for installation instructions.

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Modified</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parent Directory</td>
<td></td>
<td></td>
</tr>
<tr>
<td>graylog-1.3.3-2.ova</td>
<td>889.901.056</td>
<td>2016-03-15T14:25:32.000Z</td>
</tr>
<tr>
<td>graylog-1.3.4-1.ova</td>
<td>961.707.008</td>
<td>2016-03-16T15:32:09.000Z</td>
</tr>
<tr>
<td>graylog-2.0.0-1.ova</td>
<td>1.049.360.896</td>
<td>2016-04-26T14:48:07.000Z</td>
</tr>
<tr>
<td>graylog-2.0.0-2.ova</td>
<td>1.051.596.800</td>
<td>2016-04-29T16:20:11.000Z</td>
</tr>
<tr>
<td>graylog-2.0.1-1.ova</td>
<td>1.035.497.984</td>
<td>2016-05-11T14:40:03.000Z</td>
</tr>
</tbody>
</table>
Docker images
graylog2/graylog

Last pushed: 6 days ago

Repo Info Tags Dockerfile Build Details

Short Description

WORK-IN-PROGRESS Official Graylog Docker image (automated build)

Docker Pull Command
docker pull graylog2/graylog

Full Description

Graylog Docker Image
docker stargazers 29
docker pulls 200k

Owner

graylog2
$ docker run --name mongo -d mongo:3
$ docker run --name elasticsearch \
  -e "http.host=0.0.0.0" -e "xpack.security.enabled=false" \ 
  -d docker.elastic.co/elasticsearch/elasticsearch:5.6.5
$ docker run --link mongo --link elasticsearch \
  -p 9000:9000 -p 12201:12201 -p 514:514 \ 
  -e GRAYLOG_WEB_ENDPOINT_URI="http://127.0.0.1:9000/api" \ 
  -d graylog/graylog:2.4.0-1
Docker compose
# Volumes for persisting data,
# see https://docs.docker.com/engine/admin/volumes/volumes/

volumes:
  mongo_data:
    driver: local
  es_data:
    driver: local
  graylog_journal:
    driver: local
"Mounts": [
  {
    "Type": "volume",
    "Name": "escritorio_mongo_data",
    "Source": "/var/lib/docker/volumes/escritorio_mongo_data/_data",
    "Destination": "/data/db",
    "Driver": "local",
    "Mode": "rw",
    "RW": true,
    "Propagation": ""
  },
  {
    "Type": "volume",
    "Name": "cab6d5d95d92e79466618afedd84219ea8899aed38e5f8716f5591a045838394",
    "Source": "/var/lib/docker/volumes/cab6d5d95d92e79466618afedd84219ea8899aed38e5f8716f5591a045838394/_data",
    "Destination": "/data/configdb",
    "Driver": "local",
    "Mode": "rw",
    "RW": true,
    "Propagation": ""
  }
],

[JaMo] password for jaMo:
CONTAINER ID      IMAGE                      COMMAND                                                      CREATED            STATUS     PORTS
59cc81e292d9      graylog/graylog:2.4.0-1    "/docker-entrypoint..." 24 minutes ago  Up 24 minutes 0.0.0.0:514->514/tcp, 0.0.0.0:9000->9000/tcp, 0.0.0.0:514->514/udp, 8.0.0.0:12201->12201/tcp, 0.0.0.0:12201->12201/udp  escritorio_graylog_1
74dd9edd670      docker.elastic.co/elasticsearch/elasticsearch:5.6.3  "/bin/bash bin/es-do..." 25 minutes ago  Up 24 minutes 9200/tcp, 9300/tcp  escritorio_elasticsearch_1
f73fa2e9ea03b      mongo:3                      "/docker-entrypoint.s..." 25 minutes ago  Up 24 minutes 27017/tcp  escritorio_mongodb_1
Graylog features

- Receives messages from multiple input protocols: GELF via HTTP/UDP/TCP, Syslog, Apache Kafka, ...
- Assigns messages to streams
- Triggers user-defined alerts per stream
- Routes messages to different outputs based on streams
- Stores messages in ElasticSearch for graphing
- Uses MongoDB to store metadata and alerts
- Provides search and graphing capabilities for stored messages
Graylog features

- **Streams**: They are message routing mechanisms in categories.
- **Alerts**: Graylog allows to define alerts that are launched when match with configured conditions.
- **Dashboards**: Control panel where you can visualize everything that happens in the monitored systems.
- **Searches**: Graylog provides a search system on the historical from where to locate the messages that help to react before problems.
- **Security**: Allows you to set permissions to users to restrict the access, display and search for messages.
ElasticSearch indexes

Indices
This is an overview of all indices (message stores) Graylog is currently taking in account for searches and analysis.

💡 You can learn more about the index model in the documentation

Settings
- **Index rotation strategy:** Message Count
- **Max docs per index:** 40000000
- **Index retention strategy:** Delete
- **Max number of indices:** 250

Update configuration
ElasticSearch indexes

Index Set: Default index set

This is an overview of all indices (message stores) in this index set. Graylog is currently taking in account for searches and analysis.

You can learn more about the index model in the documentation.

<table>
<thead>
<tr>
<th>Index prefix:</th>
<th>graylog</th>
<th></th>
<th>Index rotation strategy:</th>
<th>Message Count:</th>
<th>Index retention strategy:</th>
<th>Delete:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shards:</td>
<td>4</td>
<td>Max docs per index:</td>
<td>200000000</td>
<td>Max number of indices:</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>Replicas:</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1 Indices with a total of 2 messages under management, current write-active index is graylog_0.

Elasticsearch cluster is green. Shards: 4 active, 0 initializing, 0 relocating, 0 unassigned. What does this mean?

graylog_0 active write index
Contains messages up to a few seconds ago (13.9KB / 2 messages)  Hide Details / Actions
Range re-calculated 3 hours ago in 0ms. 2 segments, 0 open search contexts, 0 deleted messages

Primary shard operations
Index: 0 ops
Flush: 2 ops (took a few seconds)
Merge: 0 ops
Query: 676 ops (took a few seconds)
Fetch: 28 ops (took a few seconds)

Total shard operations
Index: 0 ops
Flush: 2 ops (took a few seconds)
Merge: 0 ops
Query: 676 ops (took a few seconds)
Fetch: 28 ops (took a few seconds)
Inputs

Inputs
Graylog nodes accept data via inputs. Launch or terminate as many inputs as you want here.

Select input   Launch new input   Find more inputs

Global inputs  2 configured

**gelf demo** GELF UDP   1 RUNNING

- bind_address: 0.0.0.0
- decompress_size_limit: 8388608
- override_source: <empty>
- port: 12201
- recv_buffer_size: 262144

Throughput / Metrics
- 1 minute average rate: 0 msg/s
- Network IO: -8B -4B (total: -259.8B +4B)
- Empty messages discarded: 0
- Show details

**gelf tcp** GELF TCP   1 RUNNING

- bind_address: 0.0.0.0
- decompress_size_limit: 8388608
- max_message_size: 2097152
- override_source: <empty>
- port: 12201
- recv_buffer_size: 1048576

Throughput / Metrics
- 1 minute average rate: 0 msg/s
- Network IO: -8B -4B (total: -259.8B +4B)
- Active connections: 0 (0 total)
- Empty messages discarded: 0
- Show details
Streams

- Incoming messages can be grouped
- Can be used for to assign user permissions
- Stream alerts can send out notifications

**Create Stream**

- Define criteria for streams
- Analyze and configure alerts
- Create pipelines & dashboards
Streams

You can route incoming messages into streams by applying rules against them. If a message matches all rules of a stream it is routed into it. A message can be routed into multiple streams.

You can for example create a stream that contains all SSH logins and configure to be alerted whenever there are more logins than usual. Read more about streams in the documentation.

Take a look at the Graylog stream dashboards for wall-mounted displays or other integrations.

Logins

All login requests

4 messages/second, Must match all of the 1 configured stream rule(s). Show stream rules
Rules of Stream »steam»

1. Load a message to test rules
   - Select an Input from the list below and click "Load Message" to load a message.
   - Enter a message into the Recent Message field
   - Enter a message into the Message ID field

2. Manage stream rules
   - Please load a message to check if it would match against these rules.
   - A message must match all of the following rules.
   - A message must match at least one of the following rules.

Field: full_message
Type: contain
Value: message
Inverted: 
Description (optional): Field full_message must contain message

Result: Field full_message must contain message

Cancel Save
<table>
<thead>
<tr>
<th>File Name</th>
<th>Description</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>AlwaysMatcher.java</td>
<td>Create Default stream (#2881)</td>
<td>a year ago</td>
</tr>
<tr>
<td>ContainsMatcher.java</td>
<td>Add support for arrays to &quot;contains&quot; stream rule (#3380)</td>
<td>a year ago</td>
</tr>
<tr>
<td>ExactMatcher.java</td>
<td>Matching inverted exact/regex stream rules when field is not present (#...)</td>
<td>2 years ago</td>
</tr>
<tr>
<td>FieldPresenceMatcher.java</td>
<td>Update license headers in Java source files</td>
<td>3 years ago</td>
</tr>
<tr>
<td>GreaterMatcher.java</td>
<td>Handle double values in greater/smaller stream matcher.</td>
<td>3 years ago</td>
</tr>
<tr>
<td>RegexMatcher.java</td>
<td>Matching inverted exact/regex stream rules when field is not present (#...)</td>
<td>2 years ago</td>
</tr>
<tr>
<td>SmallerMatcher.java</td>
<td>Handle double values in greater/smaller stream matcher.</td>
<td>3 years ago</td>
</tr>
<tr>
<td>StreamRuleMatcher.java</td>
<td>Update license headers in Java source files</td>
<td>3 years ago</td>
</tr>
<tr>
<td>Stream</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Default</td>
<td>Stream used by default for messages not matching another stream.</td>
<td></td>
</tr>
<tr>
<td>ALL</td>
<td>All messages</td>
<td></td>
</tr>
<tr>
<td>HTTP</td>
<td>All HTTP Traffic (extracted from pipelines)</td>
<td></td>
</tr>
<tr>
<td>HoneyPot</td>
<td>Stream from Honeypot</td>
<td></td>
</tr>
<tr>
<td>SSH (Info)</td>
<td>Show accepted/failed SSH</td>
<td></td>
</tr>
<tr>
<td>d8</td>
<td>All from d8</td>
<td></td>
</tr>
<tr>
<td>mail</td>
<td>All that coming with mail</td>
<td></td>
</tr>
<tr>
<td>mysql</td>
<td>All mysql logs</td>
<td></td>
</tr>
</tbody>
</table>
Alerts configuration for stream »Logins«

You can define thresholds on any message field or message count of a stream and be alerted based on this definition.

Learn more about alerts in the documentation.

Add new alert condition

Configured alert conditions

Message count condition
Alert is triggered when there is less than 1 message in the last 3 minutes. Grace period: 10 minutes. Not including any messages in alert notification.

Callbacks
The following callbacks will be performed when this stream triggers an alert.

Email Alert Callback
Executed once per triggered alert condition.

sender: graylog@example.org

body: 
Alert Description: ${check result.resultDescription}
GrayLog architecture
Connecting with Java
Sending log data to graylog

● **Syslog**
  – TCP, TCP+TLS, UDP, AMQP, Kafka

● **GELF**
  – TCP, TCP+TLS, UDP, HTTP, AMQP, Kafka

● **Raw / Plain Text**
  – TCP, TCP+TLS, UDP, AMQP, Kafka

● **Collector**
  – TCP, TCP+TLS
GELF

- Graylog Extended Log Format
- Logstash, fluentd, nxlog, Docker, ...
- Based in syslog and rsyslog
- JSON based format for sending structured data
- JSON Hash with mandatory fields:
  - host, version, short_message, full_message, timestamp, level
GELF document

```json
{
  "version": "1.1",
  "timestamp": 1385053862.3072,
  "host": "example.org",
  "short_message": "A short message",
  "full_message": "A full message,
  "level": 1,
  "_user_id": 9001,
  "_http_response_code": 500
  "_some_env_var": "env_var"
}
```
Graylog message inspector

<table>
<thead>
<tr>
<th>Timestamp</th>
<th>Source</th>
<th>Level</th>
<th>Some Info</th>
<th>Timestamp</th>
</tr>
</thead>
</table>

- **Message**: A short message
- **Received by**: gelf demo on acd99fe3 / 69cc81e202d9
- **Stored in index**: graylog_0
- **Routed into streams**: All messages

**full_message**
A full message

**message**
A short message

**some_info**
message

**source**
localhost

**timestamp**
Apache Log4j 2

Apache Log4j 2 is an upgrade to Log4j that provides significant improvements over its predecessor, Log4j 1.x, and provides many of the improvements available in Logback while fixing some inherent problems in Logback’s architecture.

Features

API Separation

The API for Log4j is separate from the implementation making it clear for application developers which classes and methods they can use while ensuring forward compatibility. This allows the Log4j team to improve the implementation safely and in a compatible manner.

Improved Performance

Log4j 2 contains next-generation Asynchronous Loggers based on the LMAX Disruptor library. In multi-threaded scenarios Asynchronous Loggers have 18 times higher throughput and orders of magnitude lower latency than Log4j 1.x and Logback. See Asynchronous Logging Performance for details. Otherwise, Log4j 2 significantly outperforms Log4j 1.x, Logback and java.util.logging, especially in multi-threaded applications. See Performance for more information.
SLF4J user manual

The Simple Logging Facade for Java (SLF4J) serves as a simple facade or abstraction for various logging frameworks, such as java.util.logging, logback and log4j. SLF4J allows the end-user to plug in the desired logging framework at deployment time. Note that SLF4J-enabling your library/application implies the addition of only a single mandatory dependency, namely slf4j-api-1.7.12.jar.

**Since 1.6.0** If no binding is found on the class path, then SLF4J will default to a no-operation implementation.

**Since 1.7.0** Printing methods in the Logger interface now offer variants accepting varargs instead of Object[]. This change implies that SLF4J requires Java 8 or later. Under the hood the Java compiler transforms the varargs part in methods into Object[]. Thus, the Logger interface generated by the compiler is indistinguishable in 1.7.x from its 1.6.x counterpart. It follows that SLF4J version 1.7.x is totally 100% no-ifs-or-buts compatible with SLF4J version 1.6.x or later.

**Since 1.7.5** Significant improvement in logger retrieval times. Given the extent of the improvement, users are highly encouraged to migrate to SLF4J 1.7.x or later.

**Since 1.7.9** By setting the slf4j.detectLoggerNameMismatch system property to true, SLF4J can automatically spot incorrectly named loggers.

Hello World

As customary in programming tradition, here is an example illustrating the simplest way to output "Hello world" using SLF4J. It begins by getting a logger by the name "HelloWorld". This logger is in turn used to log the message "Hello World".

```java
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

public class HelloWorld {
    public static void main(String[] args) {
        Logger logger = LoggerFactory.getLogger(HelloWorld.class);
        logger.info("Hello World");
    }
}
```
Logback Project

Logback is intended as a successor to the popular log4j project, picking up where log4j leaves off.

Logback's architecture is sufficiently generic so as to apply under different circumstances. At present time, logback is divided into three modules, logback-core, logback-classic and logback-access.

The logback-core module lays the groundwork for the other two modules. The logback-classic module can be assimilated to a significantly improved version of log4j. Moreover, logback-classic natively implements the SLF4J API so that you can readily switch back and forth between logback and other logging frameworks such as log4j or java.util.logging (JUL).

The logback-access module integrates with Servlet containers, such as Tomcat and Jetty, to provide HTTP-access log functionality. Note that you could easily build your own module on top of logback-core.

Sister projects

The logback-audit project is designed for processing logging events having long-term business significance. Logback-audit is based on logback-core.
Jars

- gelfj-1.1.16.jar
- logback-gelf-1.1.11.jar
# Gelfj » 1.1.16

GELF implementation in Java and log4j appender without any dependencies.

<table>
<thead>
<tr>
<th>License</th>
<th>MIT</th>
</tr>
</thead>
<tbody>
<tr>
<td>HomePage</td>
<td><a href="https://github.com/t0xa/gelfj">https://github.com/t0xa/gelfj</a></td>
</tr>
<tr>
<td>Date</td>
<td>(Jan 08, 2018)</td>
</tr>
<tr>
<td>Files</td>
<td>pom (7 KB)</td>
</tr>
<tr>
<td>Repositories</td>
<td>Central</td>
</tr>
<tr>
<td>Used By</td>
<td>1 artifacts</td>
</tr>
</tbody>
</table>

```xml
<dependency>
    <groupId>org.graylog2</groupId>
    <artifactId>gelfj</artifactId>
    <version>1.1.16</version>
</dependency>
```
private DatagramChannel initiateChannel() throws IOException {
    DatagramChannel resultingChannel = DatagramChannel.open();
    resultingChannel.socket().bind(new InetSocketAddress(0));
    resultingChannel.connect(new InetSocketAddress(this.host, this.port));
    resultingChannel.configureBlocking(false);
    return resultingChannel;
}

public GelfSenderResult sendMessage(GelfMessage message) {
    if (!message.isValid()) {
        return GelfSenderResult.MESSAGE_NOT_VALID;
    }
    return sendDatagrams(message.toUDPBuffers());
}
<appender name="graylog2" class="org.graylog2.log.GelfAppender">
  <param name="graylogHost" value="192.168.0.201"/>
  <param name="originHost" value="my.machine.example.com"/>
  <param name="extractStackTrace" value="true"/>
  <param name="addExtendedInformation" value="true"/>
  <param name="facility" value="gelf-java"/>
  <param name="Threshold" value="INFO"/>
  <param name="additionalFields" value="{"environment": 'DEV', 'application': 'MyAPP'}"/>
</appender>
private void initSender(){
  try{
    // Sender is UDP
    gelfSender = new GelfUDPSender(getServer(), getInputPort());
  } catch (IOException e) {
    e.printStackTrace();
  }

  private boolean sendGelfMessage(GelfMessage message){
    boolean result = false;
    message.setHost(getClient());
    // validate and send message
    if (message.isValid()) {
      result = getGelfSender().sendMessage(message);
    }
    return result;
  }

  public boolean sendMessage(String title, String description){
    // compose message
    GelfMessage message = new GelfMessage(
        title, description, new Date().getTime(), getAlertLevel());
    return sendGelfMessage(message);
  }
}
public static void main(String[] args) {
    ApplicationContext context = new ClassPathXmlApplicationContext("SpringBeans.xml");
    GraylogService graylogService = (GraylogService) context.getBean("graylogService");
    graylogService.sendMessage("Test message", "This message is a test message");
}

<bean id="graylogService" class="graylogtest.GraylogService">
    <property name="server" value="server"/>
    <property name="inputPort" value="port"/>
    <property name="alertLevel" value="6"/> <!-- 6 means INFO -->
</bean>
LogBack

- [https://github.com/pukkaone/logback-gelf](https://github.com/pukkaone/logback-gelf)
- JDK >= 1.7

Add the following dependency to your project:

```xml
<dependency>
  <groupId>com.github.pukkaone</groupId>
  <artifactId>logback-gelf</artifactId>
  <version>1.1.11</version>
</dependency>
```
package com.github.pukkaone.gelf.protocol;

import com.fasterxml.jackson.core.JsonProcessingException;

public class GelfMessage {
    public static final String FACILITY = "facility";
    private static final String VERSION_VALUE = "1.1";
    private static final String HOST = "host";
    private static final String SHORT_MESSAGE = "short_message";

    private static final ObjectMapper OBJECT_MAPPER = new ObjectMapper();
    private long timestampMillis;

    private Map<String, Object> fieldNameToValueMap = new HashMap();

    public GelfMessage() {
        this.fieldNameToValueMap.put("version", "1.1");
    }

    public long getTimestampMillis() {
        return this.timestampMillis;
    }
LogBack appender

Configure a logback appender to send by UDP (XML configuration format):

```xml
<appender name="GRAYLOG" class="com.github.pukkaone.gelf.logback.GelfAppender">
    <graylogHost>graylog.example.com</graylogHost>
    <originHost>my.machine.example.com</originHost>
    <levelIncluded>true</levelIncluded>
    <locationIncluded>false</locationIncluded>
    <loggerIncluded>true</loggerIncluded>
    <markerIncluded>false</markerIncluded>
    <mdcIncluded>false</mdcIncluded>
    <threadIncluded>false</threadIncluded>
    <facility>gelf-java</facility>
    <additionalField>application=MyApplication</additionalField>
    <additionalField>environment=development</additionalField>
</appender>
```
public class GraylogRestInterface {

    private final RestTemplate restTemplate = new RestTemplate();

    private final UriComponentsBuilder uriBuilder = UriComponentsBuilder.newInstance()
        .scheme("http").host("localhost").port(12900);

    public void logEvent(GelfMessage message) {
        ResponseEntity entity = new ResponseEntity<>(message, buildHeaders());

        restTemplate.postForEntity(uriBuilder.cloneBuilder().port(12202).path("gelf")
            .toUriString(), entity, null);
    }
}
// Log message via Graylog HTTP Input
GelfMessage message = new GelfMessage();
message.setShortMessage("Short message");
message.setFullMessage("Full message");
message.getAdditionalProperties().put("elapsed_time", timer.stop().
elapsed(TimeUnit.MICROSECONDS));
graylog.logEvent(message);
Connecting with other services
Hundreds of Add-ons for Graylog.

How would you like to extend Graylog today?

Search Marketplace
Sending syslog via AMQP into Graylog

Other Solutions
How to use send Syslog messages via AMQP to Graylog

logstash-forwarder rsyslog rabbitMQ AMQP

jalogisch

View on Github

Published: 27 May 06:49
Last Push: 08 Sep 06:10
Marketplace Rating: No rating yet
Discussion: 0 Comments
Sending syslog via KAFKA into Graylog

Other Solutions

This Guide will give you little help on using Graylog with Kafka Input to get Syslog Data

syslog kafka

jalogisch

View on Github

Published 12 Sep 03:47
Last Push 12 Sep 03:45
Marketplace Rating No rating yet
Discussion 0 Comments
References

- https://github.com/Graylog2/graylog-docker
- https://hub.docker.com/r/graylog2/graylog/
- http://docs.graylog.org/en/2.4/pages/installation/docker.html
Thanks!

Contact:

@jmortegac
jmortega.github.io
about.me/jmortegac