

Message logging using Graylog

DM Logging requirements

- Log messages will be sent to a central server
- The format of the network log messages sent over the network is GELF
- Minimum initial requirement: message on startup and clean shutdown
- Requirement down the road: it should be possible to track an event when logging is set to "Ridiculously verbose"

A GELF log message

```
{  
    "level":7,  
    "host":"dmsc-test",  
    "version":"1.1",  
    "timestamp":1485169769.4658048,  
    "short_message":"Hold my beer and  
watch this!",  
    "_thread_id":48731321,  
    "_process_id":4213,  
    "_process":"event_processing",  
}\0
```

Network transmission

- GELF specifications here: <http://docs.graylog.org/en/2.1/pages/gelf.html>
- Over TCP or UDP

Super simple Python example

```
sock = socket.socket(socket.AF_INET, ↴  
socket.SOCK_DGRAM)  
  
sock.sendto(json.dumps(msg).encode("utf-8") ↴  
+ b'\x00', (host, port))
```

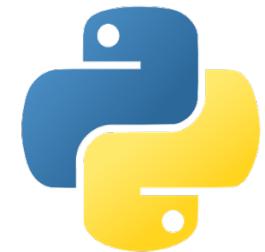
Graylog servers

- Install as a Vagrant machine: <https://bitbucket.org/europeanspallationsource/dm-graylog-machine>
- Available on DMSC network via **ssh0:ssh
user_name@ssh0.esss.dk -L
7778:192.168.12.11:12201 -N**
- Eventually available in the ESSIIP lab

Python GELF libraries

- Some examples
 - pygelf
 - gelf-python
 - python-gelfclient
 - graypy
 - gelfHandler
- GraylogHandler





GraylogHandler

- <https://bitbucket.org/europeanspallationsource/grayloghandler>
- Features
 - Automatic TCP connection to localhost
 - Message queue
 - Extra meta data
 - Thread, process, process id



GraylogHandler usage

```
import logging
from GraylogHandler import GraylogHandler

gHandler = GraylogHandler()
logging.getLogger().addHandler(gHandler)

logging.error("This is a fatal error!")
```

C++ GELF libraries

- gelf4cplus
 - Requires boost, zlib, log4cplus
 - Header only
 - Uses UDP
- dm-graylog-logger
 - Simple logging library
 - Uses only std-libraries
 - Uses TCP
 - Logs to console and file as well

Is there a demand for a boost::log sink?

dm-graylog-logger usage

- [https://bitbucket.org/
europeanspallationsource/dm-graylog-
logger](https://bitbucket.org/europeanspallationsource/dm-graylog-logger)
- Ways to use:
 - Copy files
 - Compile and link to library

dm-graylog-logger usage

```
#include <thread>
#include <chrono>
#include <graylog_logger/Log.hpp>

int main() {
    Log::Msg(Severity::Warning, "Some
message.");
    std::this_thread::sleep_for(std::chrono
        ::milliseconds(100));
    return 0;
}
```

Alternatives

- Use Filebeat to send log messages from text file
- Use other application to send text file logs
- Write your own Graylog/GELF interface

Other considerations

- Additional meta data
- Deciding logging level at compile time
- Maximize logging performance
- Special logging features
- Additional languages

Workshop: Get started with Graylog logging

- Graylog server using vagrant
 - Web interface on <http://localhost:8080/>
 - User: **admin**, password: **admin**
- Graylog server at DMSC
 - Web interface: 192.168.12.11:9000
 - Login using your DMSC username and password
 - Web proxy: **ssh -D 7777 user@ssh0.esss.dk -N**
 - Port tunnel (TCP only): **ssh user@ssh0.esss.dk -L 7778:192.168.12.11:12201 -N**

Practical demonstration checklist

- Graylog web interface running on Vagrant machine
- Accessing the DMSC Graylogserver
- Sending log messages to the DMSC Graylogserver
- A quick overview of the C++ library
- Making *forward-epics-to-kafka* emit GELF messages