Logging

Logging in openemail: dockerized consists of multiple stages, but is, after all, much more flexible and easier to integrate into a logging daemon than before.

In Docker the containerized application (PID 1) writes its output to stdout. For real one-application containers this works just fine. Run `docker-compose logs --help` to learn more.

Some containers log or stream to multiple destinations.

No container will keep persistent logs in it. Containers are transient items!

In the end, every line of logs will reach the Docker daemon - unfiltered.

The **default logging driver is "json"**.

Filtered logs

Some logs are filtered and written to Redis keys but also streamed to a Redis channel.

The Redis channel is used to stream logs with failed authentication attempts to be read by netfilter-openemail.

The Redis keys are persistent and will keep 10000 lines of logs for the web UI.

This mechanism makes it possible to use whatever Docker logging driver you want to, without losing the ability to read logs from the UI or ban suspicious clients with netfilter-openemail.

Redis keys will only hold logs from applications and filter out system messages (think of cron etc.).

Logging drivers

Here is the good news: Since Docker has some great logging drivers, you can integrate openemail: dockerized into your existing logging environment with ease.

Docker logging drivers can now be implemented as plugins, next to Dockers integrated drivers. Logging driver plugins are available in Docker 17.05 and higher.

Edit `docker-compose.yml` and append, for example, this block to use the "gelf" logging plugin:

```
logging:
  log_driver: "gelf"
  options:
    gelf-address: "udp://graylog:12201"
    gelf-tag: "openemail-logs"
```

Linux users can also add or edit the Docker daemons configuration file `/etc/docker/daemon.json` to affect the global logging behavior. Windows users please have a look at the docker documentation:

```
{
...
    "log-driver": "gelf",
    "log-opts": {
        "gelf-address": "udp://graylog:12201",
        "gelf-tag": "openemail-logs"
    }
...
}
```

Restart the Docker daemon and run `docker-compose down && docker-compose up -d` to recreate the containers with the new logging driver.