Author	Thiébaud Modoux
Reviewer	Ilia Kebets
Date	08.05.2019
Version	3

Pryv.io audit

Installation and configuration

Summary

We have added audit capabilities within Pryv.io through the installation of our service-router, which stands in front of Pryv.io cores, proxies the API calls to them while logging details about the requests and responses to the host's syslog.

Router installation and configuration

The first thing to note is that the service-router will stand inbetween NGINX and the cores. This implies a change in the NGINX configuration, so that the calls targeting the cores first transit through the service-router and are then redirected.

As an example, the following NGINX definition of cores :

```
# core/nginx/conf/site-443.conf

upstream core_server {
  server core:3000 max_fails=3 fail_timeout=30s;
  server core:3001 max_fails=3 fail_timeout=30s;
}
```

would simply be replaced by a link to the router:

```
# core/nginx/conf/site-443.conf

upstream core_server {
  server core_router:1337 max_fails=3 fail_timeout=30s;
}
```

Now, we have to declare the router and instruct it to route to the cores we just removed from the NGINX configuration.

This is done by adding the router service to the docker-compose file:

```
    # core.yml

      core_router:
        image: "pryvsa-docker-release.bintray.io/pryv/router:0.1.18"
        networks:
          - frontend
        links:
          - core
        environment:
          ROUTER_LOG: info
          SCRYPT_SALT: C4E4DA10CC50DE7E93FAC97E4E
        volumes:
          - /dev/log:/dev/log
        command: --core-audit core:3000 --core-audit core:3001
        restart: always
```

Notes:

- The `ROUTER_LOG` environment variable defines the logging level for the router process.
- Replace the `SCRYPT_SALT` environment variable with a salt generated on your own. We recommend providing 26 hexadicemal characters, similarly to a 128-bit WEP key. If this variable is not specified, the string '**00000000000000**' will be used as default scrypt salt.
- The `command` line specifies the core api-servers (backends) it routes to, they should match the ones we removed from the Nginx configuration. The number of api-servers depends on the `core.environment.NUM_PROCS` variable defined within the same docker-compose file, with starting port defined by `core.environment.STARTING_PORT` (default is 3000).

Syslog configuration

Introductory notes about syslog:
The syslog protocol is using a socket in order to transmit messages. For Linux, this socket is a SOCK_STREAM unix socket, which is identified by the name /dev/log. The syslog deamon for Ubuntu is rsyslogd, its configuration files are located in /etc/rsyslog.conf and /etc/rsyslog.d/. In particular, the default logging rules can be found in /etc/rsyslog.d/50-default.conf. These rules typically tell to which actual log files the socket messages will be pipped to (e.g. /var/log/syslog), according to the message type (see the Syslog wiki for more details about Facility and Security levels).*

The router service will write a log entry for each API request and response to the syslog. Thus said, it would be useful for an auditor to have the audit logs organized in separate files, according to some chosen criteria.

For now, we propose to solve this by configuring the way syslog is writting logs. As described below, this can be done by putting in place a template with some filters in the rsyslog configuration (note that it is also possible to configure writing to a database, see this article).

Logs organization

The following rsyslog configuration snippet allows to organize the audit logs per username and per token.

In other words, the username, extracted from the log message, will be used as log folder name, so one log folder will be created per username. Similarly, the token hash will be used as log filename, so one log file will be created per token hash.

```
# /etc/rsyslog.d/pryv-router.rsyslog.conf

$template myfile,"/var/log/pryv/audit/%programname%/%$.username%/%$.token%.log"
if ($programname == "pryvio_core") then {
  set $.username = replace(re_extract($msg,
    "\"(username)\":\"([^\"]+)\"", 0, 2,
    "no_username"),
    "/", "%2F");
  set $.token = replace(re_extract($msg,
    "\"(authorization_hash|auth_hash)\":\"([^\"]+)\"", 0, 2,
    "no_token"),
    "/", "%2F");
 action(type="omfile" dynaFile="myfile")
 stop
}
```

Here is a step-by-step explanation of this configuration:

- Define the name of the log files, in this case logs will be in `/var/log/pryv/audit/pryvio_core/${USERNAME}/${TOKEN_HASH}.log`
- Only consider pryvio_core logs by checking the program name
- Extract the username from log messages, set it to "no_username" if the regex does not match
- Extract the token hash from log messages, set it to "no_token" if the regex does not match
- Both for username and token hash, encode "/" special char to avoid creating subfolders
- Define the action, write to a file in this case
- Prevent (stop) default writing to syslog (optional)

Once the new configuration is in place, the rsyslog service can be restarted by running `sudo service rsyslog restart`. Additionally, the command `rsyslogd -N1` is useful to check if the new rsyslog configuration is valid.

Logs extraction

Now that logs are organized per user and per token, an auditor will be able to fetch all the logs based on a specific username and token. Here is the procedure:

1. Use an implementation of a scrypt hash generator (e.g. Rust, JS, Python).
2. Set the scrypt parameters as follows: **N** = 32'768 (log2(N) = 15), **R** = 8, **P** = 1
3. Set the scrypt salt according to the `SCRYPT_SALT` value provided during the router configuration or to '**0000000000000**' if using the default salt.
4. Compute the hash of the specific token.
5. Use this hash to find the corresponding log file: `/var/log/pryv/audit/pryvio_core/${USERNAME}/${TOKEN_HASH}.log`

Logs rotation

In order to ease administration of large numbers of logs, logs rotation can be configured with a tool like logrotate.

Here is an example of a logrotate configuration for service-router logs:

```
# /etc/logrotate.d/pryv-router.logrotate.conf

/var/log/pryv/audit/pryvio_core/*/*.log {
  rotate 12
  monthly
  missingok
  notifempty
}
```

Logrotate allows automatic rotation, compression, removal, and mailing of log files. Each log file may be handled daily, weekly, monthly, or when it grows too large. Please see the logrotate manpage for a list of all available options.