Author	Thiébaud Modoux (Pryv SA)
Reviewer	Ilia Kebets (Pryv SA)
Date	18.07.2019
Version	5

Pryv.io audit

Installation and configuration

Introduction

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.

In addition, our service-audit serves an API that allows to fetch audit records produced by the service-router. Reference documentation for this API can be found here.

Audit router setup

The first thing to note is that the service-router will stand inbetween NGINX and the cores. As we can see in the NGINX configuration, API calls targeting the cores first transit through the service-router and are then redirected:

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

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

Then, the router Docker image is added to the docker-compose file:

```
# core.yml

 core_router:
  image: "pryvsa-docker-release.bintray.io/pryv/router:1.0.2"
  networks:
   - frontend
  links:
   - core
  environment:
   ROUTER_LOG: info
  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.
- The `command` line specifies the core api-servers (backends) it routes to. 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. By default, all these logs will be written to one file (i.e. /var/log/syslog).

Thus said, it would be more convenient to have a specific folder for these audit logs and to organize them in separate files. As described below, this can be done by putting in place a template within the rsyslog configuration.

Logs organization

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

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.

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

$template myfile,"/var/log/pryv/audit/%programname%/%$.username%/audit.log"
if ($programname == "pryvio_core") then {
  set $.username = replace(re_extract($msg,
    "\"(username)\":\"([^\"]+)\"", 0, 2,
    "no_username"),
    "/", "%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}/audit.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
- 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 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.

Audit API setup

The audit logs produced by service-router can be fetched thanks to the audit API.

Thus, the audit Docker image is also added to the docker-compose file:

```
# core.yml

 core_audit:
  image: "pryvsa-docker-release.bintray.io/pryv/audit:1.0.0"
  networks:
   - frontend
  links:
   - core
  volumes:
   - ./core/audit/conf/:/app/conf/:ro
   - ./core/audit/log/:/app/log/
   - /var/log/pryv/audit/pryvio_core/:/app/data/
  restart: always
```

Notes:

- In addition to the usual `conf` and `log` folders, we mount the audit logs folder as a `data` volume. The path `/var/log/pryv/audit/pryvio_core/` is as defined previously during the Syslog configuration/Logs organization. Please note that service-audit needs read access on this folder (`chmod -R 755 /var/log/pryv/audit/pryvio_core/`).

The audit api can be configured through the configuration file `/core/audit/conf/audit.json`. Here is an example of such configuration:

```
# /core/audit/conf/audit.conf

{
 "core": {
  "url": "http://core:3000"
 },
```

```
  "dataFolder": "/app/data",
  "http": {
    "port": 5000,
    "ip": "0.0.0.0"
  },
  "logs": {
    "prefix": "audit",
    "console": {
      "active": true,
      "level": "debug",
      "colorize": true
    },
    "file": {
      "active": false
    }
  }
}
```

Notes:

- The `core:url` should specify the url to reach service-core. Since service-core is on the same machine as service-audit, the endpoint corresponds to the local core Docker container. Service-core is mandatory to ensure the validity of the tokens used through service-audit.
- The `dataFolder` should match the volume we mounted above for `core_audit` within `core.yml`.