MONARC technical guide

"security made in Lëtzebuerg" (SMILE) g.i.e

Table of Contents

1. Introduction

This document is currently under active development and discussion!

This document is intended to administrators of a MONARC instance. If you find errors or omissions in this document, please don't hesitate to submit an issue or open a pull request with a fix.

2. Modules of the project

The source code of MONARC is divided in several modules with dedicated Git repositories.

Summary of the different modules of the project

	Back office	**Front office**
	MONARC BO: Management of the database, clients and FO servers.	MONARC: The MONARC front office
Modules	• MonarcCore (zm-core in `vendor/monarc/core`) • MonarcBO (zm-backoffice in `vendor/monarc/backoffice`)	• MonarcCore (zm-core in `vendor/monarc/core`) • MonarcFO (zm-client in `vendor/monarc/frontoffice`)
Interfaces (in node_modules/)	• ng_anr • ng_backoffice	• ng_anr • ng_client

- MonarcCore (zm-core) is the API in charge of providing an access to the data. Common to the front office and the back office.
- MonarcFO (zm-client) is the interface for the front office.
- MonarcBO (zm-backoffice) is the interface for the back office.

3. Architecture

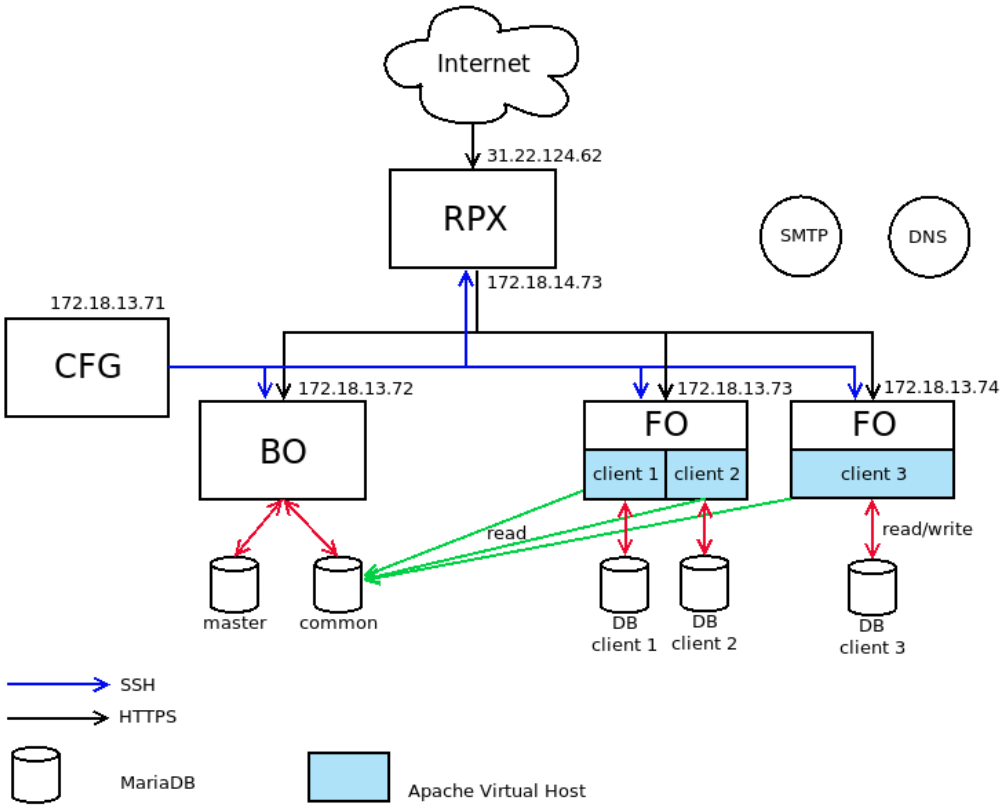
3.1. Global architecture

Basically MONARC is composed of two main parts:

- a back office: for the management of the database, servers and clients of MONARC installations;
- a front office: for the management of the risks analysis. A front office instance can be used without the back office.

If you want run your own risk analysis you only need the front office. More information about the installation in this section.

The whole architecture includes an additional reverse proxy (**RPX**) and a configuration server (**CFG**). It is possible to connect many front offices (**FO**) to one back office (**BO**). It is automatically deployed with ansible. More information in the repository of the ansible playbook.



Example of IPs and FQDNs:

- 172.18.14.73 - monarc2-rpx.test.your.domain.com - my.monarc.lu
- 172.18.13.71 - monarc2-conf.test.your.domain.com
- 172.18.13.72 - monarc2-master.test.your.domain.com
- 172.18.13.73 - monarc2-fo01.test.your.domain.com
- 172.18.13.74 - monarc2-fo02.test.your.domain.com

The **FO** server *monarc2-fo01* hosts two distinct databases for two clients. An Apache Virtual Host is configured for each clients.

A user who wants to do a security analysis with MONARC will for example access to his MONARC account via the address https://my.monarc.lu/client1 (assuming my.monarc.lu is the FQDN of the **RPX**).

It is strongly recommended to use a TLS certificate on the reverse proxy in order to provide a HTTPS connection between Internet and the reverse proxy. Depending your setup you will ideally provide a HTTPS connection after the reverse proxy. If you want so, make sure to have a TLS certificate for each server (or a wildcard certificate).

3.1.1. Network requirements

The configuration server (**CFG**) manages the configurations of the back office, reverse proxy and different front office(s) via SSH.

The reverse proxy (**RPX**) should of course be available from Internet. The **DNS** is important for the reverse proxy in order to resolve the FQDNs of the servers inside the network.

Through the reverse proxy you should be able to contact the **BO** and the **FO**s from Internet. No need to able able to contact the configuration server from the outside.

The database *common* of the BO should also be available to the FO servers.

Postfix should be installed on the BO and on the FO. Indeed, SMTP is used for the account creation and the password recovery. The easiest configuration is to set up Postfix for relaying emails through your internal mail server (**SMTP**).

3.2. Specifications by server

Specifications given only as advices.

	Back office	**Front office**	**Reverse proxy**	**Configuration server**
Number of vCPU	4	4		
RAM (GB)	16	16		
HDD (BG)	25	25		
Operating system	Ubuntu Server 16.04.2 LTS	Ubuntu Server 16.04.2 LTS	Ubuntu Server 16.04.2 LTS	Ubuntu Server 16.04.2 LTS
Comment			Not that powerful. But all connections (for the BO and FOs) will go through it.	Not that powerful. Mainly used for the creation of clients on the different FOs.

4. Requirements and deployment

4.1. Requirements

4.1.1. Network

The deployment on the different servers requires an Internet connection since the updates are retrieved from our GitHub repositories. If this is hardly possible in your environment due to an

internal policy, you need to configure a proxy in order to access to Internet. This is an important requirement since all MONARC updates, including security updates, are provided from the Git repositories.

Our ansible playbook also needs an Internet connection in order to dynamically deploy new clients.

The servers (or the proxy) should be able to contact:

- github.com
- api.github.com
- pypi.python.org
- registry.npmjs.org
- deb.nodesource.com
- getcomposer.org
- .packagist.org
- packagist.phpcomposer.com
- letsencrypt.org

4.1.2. TLS certificate

Is is strongly advised to use a TLS certificate on the reverse proxy. Depending on your configuration, you may need to secure the connection after the reverse proxy (see the global architecture).

The certificate **yourcert.crt** and the key file **yourcert.key** are by default located at `/etc/sslkeys/` (location configurable using the variables in the `inventory/hosts` file of the ansible playbook). You can also use a single **.pem** file, but make sure it includes the certificate and the key.

4.1.3. System and software

The deployment of MONARC has been tested on Ubuntu 16.04 LTS and Ubuntu 17.04.

- Git;
- PHP (version 7.0 recommended);
  - PHP extensions: xml, mbstring, mysql, zip, unzip, mcrypt, intl, gettext, imagick, ssl
  - PHP libraries are managed with composer;
- JavaScript libraries are managed with npm;
- Apache 2;
- MariaDB;
- Postfix (SMTP is used for the account creation and password recovery).

4.2. Deployment

4.2.1. Only the front office

Prepared virtual machine

A virtual machine for use with VirtualBox is available if you want to quickly test MONARC.

The best way if you want to try an up-to-date version of MONARC.

Manual deployment

Follow the instructions here.

4.2.2. The back office and the front office

The whole architecture can be deployed with ansible. More information here.

5. Updates

Before updating MONARC it is strongly advised to configure database backup. For that you just need to create a file `data/backup/credentialsmysql.cnf`:

```
[client]
host     = localhost
user     = sql-monarc-user
password = your-password
socket   = /var/run/mysqld/mysqld.sock
[mysql_upgrade]
host     = localhost
user     = sql-monarc-user
password = your-password
socket   = /var/run/mysqld/mysqld.sock
basedir  = /usr
```

If this file is not present, a warning message will be displayed during the update.

5.1. System update

Keep the software of your distribution up-to-date (Apache, PHP, MariaDB, Postfix, etc.). At least the security updates from the GNU/Linux distribution.

5.2. MONARC update

The master branch should always be working and it is recommended to install the project using this one.

If you have already installed MONARC and want to update to a later version, you can use the provided script:

```
$ scripts/update-all.sh
$ sudo systemctl restart apache2
```

This script will retrieve the updates from the last stable release of MONARC.

For more details the script will:

- backup your databases in case the update fails;
- check the presence of composer;
- retrieve the new code from the Git repositories from the last stable release:
  - pull updates for module/[MonarcCore, MonarcBO/MonarcFO];
  - pull updates for node_modules/[ng_anr, ng_backoffice/ng_client].
- run the appropriate database upgrade scripts (pathCore, pathBO, pathFO);
- install/update JavaScript libraries with npm;
- update the translations.