PASIR/Classr API Installation Guide

Predictive Analytics for Server Incident Reduction (PASIR)
IBM GTS – IBM Research

Author: David Lanyi dla@zurich.ibm.com
Last edited: 1/10/2017 (v1)

Overview

This document is a comprehensive step-by-step installation guide for the PASIR/Classr API to support a PASIR infrastructure delivery effort.

PASIR/Classr API is an HTTP-based microservice performing machine-learning-based ticket classification for the PASIR analysis. It is an integral part of the PASIR technical infrastructure.

This guide specifies the steps how to install the API on a freshly provisioned Red Hat Enterprise Linux 6.8 server. The server is required to have:

- Yum configured properly such that the required packages can be installed.
- Network access to the public internet and other PASIR components, including especially
  - The git server where the API code is hosted
  - The PASIR database
- At least 16GB of memory
- At least 50GB of free space at the mount point `/app`.

Preparation

1. Create a GitLab account

Log in to the GitLab server where the API source code is hosted using your IBM Intranet ID. This is required before the next step since the first login automatically registers an account for you.

- https://gitlab.zurich.ibm.com/users/sign_in

2. Request repository access

Contact the app owner (**David Lanyi, dla@zurich.ibm.com**) in e-mail or Sametime about the following matters:

- Request access to the **pasir-classr** repository by specifying the IBM Intranet ID you used in the previous step.
- Request the app owner to create a dedicated configuration file for your server. Please specify the following in your request:
    - the full **hostname** of your server (as returned by `hostname -f`),
    - the **capacity** of your server (CPU cores, memory, disk)

Proceed with the installation, once the repository access is granted.

Installation

1. Log in to your server

You should use a user account with sudo permissions or the root account.

2. Install git

Unfortunately, yum on RHEL 6.8 only supports git 1.7.1, which for some bugs is not recommended to use. Instead, install git 2.0.5 from source executing the commands below.

```
sudo -i
yum install -y curl-devel expat-devel gettext-devel openssl-
devel zlib-devel
yum install -y gcc perl-ExtUtils-MakeMaker
cd /usr/src
wget https://www.kernel.org/pub/software/scm/git/git-
2.0.5.tar.gz
tar xzf git-2.0.5.tar.gz
cd git-2.0.5
make prefix=/usr/local/git all
make prefix=/usr/local/git install
echo "export PATH=$PATH:/usr/local/git/bin" >> /etc/bashrc
source /etc/bashrc
exit
source /etc/bashrc
```

3. Download the API source code

```
git clone https://gitlab.zurich.ibm.com/dla/pasir-classr.git
```

When prompted, please enter the IBM Intranet ID with its password you used for registering the GitLab account.

If the connection fails because the DNS is not configured properly on the server, you can hardcode the IP address for the git server by running the following command, then try cloning again:

```
sudo echo "9.4.137.134 gitlab.zurich.ibm.com" >> /etc/hosts
```

4. Install the API

```
cd pasir-classr
chmod +x install.sh
sudo ./install.sh
```

Should the installer have any problem, please note the actual error message with any contextual information and contact the app owner. Please note the following:

- RHEL 6.8 is shipped with Python 2.6, which is below the requirement of the API (2.7), therefore Python 2.7 is attempted to be installed from source, including the package manager Pip.
- Java 7 is required to run the application, for which the IBM version (1.7.1) is used by the installer script.
- Commands from the installation script `install-rhel.sh` can be tested manually around the error to get an idea of a possible solution or a workaround. Googling actual error messages can also lead to quick solutions.

If the installer finished successfully, you should see a summary with the environmental parameters:

```
**********************************************************************
Destination                : /opt/pasir-classr
Configs location           : cd /opt/pasir-classr/config && ls
Command to start the API   : sudo /etc/init.d/pasir-classr start
Command to follow the log  : tail -f -n 25 /app/pasir-classr-data/logs/classr.log
**********************************************************************
PASIR/Classr API installation finished.
```

5. Verify the configurations

Open the server's configuration file.

```
sudo nano /opt/pasir-classr/config/`hostname -f`.cfg
```

Verify the configuration, especially:

- HTTP port of the API (port 80 requires root to start as)
- API key
- PASIR DB address and credentials

Use CTRL+x to exit the editor.

If you happen to see an empty file, it means the server-dependent configuration is not present or is named improperly. Browse the `/opt/pasir-classr/config` directory to locate the corresponding file, if present, and fix the name to match the actual hostname of the server. Only proceed, if there is a proper configuration in place.

6. Start the API

```
sudo /etc/init.d/pasir-classr start
```

7. Check startup

Follow the API startup by opening the log file in continuous mode, and check if the startup is successful. Use CTRL+C to exit monitoring the **API log**:

```
tail -f /app/pasir-classr-data/logs/classr.log
```

Alternatively, or if the above command does not find the specified file (perhaps because of a start-up error), also check the **error log**:

```
tail -n 30 /app/pasir-classr-data/logs/stderr.log
```

8. Check if API is online

Open a web browser and type the address of the server with the corresponding port to test if the API is accessible from the outside:

- http://<server-hostname>:<configured-port>

Note: Synchronization on first start

The first start of the API usually involves synchronizing classifier objects with another running instance of the API ('autosync'). This consists of a one-time movement of large volume of data (8-9GB) over the network, and the decompression on the server.

Make sure the configured TEMP_PATH and WORK_PATH directories (see the config file) have enough space on their mount to support the storage and decompression of the synced files (~20GB).

This one-off job usually takes 20 to 200 minutes depending on the network and CPU speed. During this time the API will be accessible but will not accept any classification jobs.

Subsequent synchronization of classifier objects will usually take substantially less time, as the biggest resources are re-used. Data transfer of subsequent synchronizations usually involve 30-40MB of data.