ibm.com/saferpayments

Best practices for Python installation and usage with Safer Payments

Safer Payments allows to run Python scripts during the computation of a message. This document describes how to set up the Python environment correctly to avoid conflicts with different Python versions installed on the system and also to avoid conflicts because of different versions of script dependencies.

Installation of Python3

It is highly discouraged to create Python scripts based on Python 2.x, which reached its end of life and support. But Python 2.7 comes as a standard with RHEL7 and it's tempting to just use what is there. Python 2.7 is only there to support the RHEL tool stack and installing system wide packages using pip can cause serious problems.

Since RHEL7.7 the Python3 packages are available in the main repositories and the installation can be done using `yum install python3`. Find instructions at the end of the document for the installation of Python3 on RHEL versions <7.7.

For Safer Payments versions greater than or equal to `v6.1.0.10`, `v6.2.1.05`, `v6.3.0.02` and `v6.4.0.00`, the Python3 libraries will be found automatically. Safer Payments searches for a `libpython3.so` file, typically located in `/usr/lib64`. For older versions of Safer Payments, a symbolic link needs to be created:

```
$ sudo ln -s /usr/lib64/libpython3.so /usr/lib64/libpython3.5.so
```

Safer Payments needs to be restarted for the libraries to be loaded. The Python version can be verified in the startup logs or in "Administration/System internals".

Python virtual environments

It is often desired to isolate project-specific dependencies and to create reproducible environments. This is especially true to make sure that the Python setup and dependencies are all the same on all instances in a Safer Payments cluster. It is easy to maintain multiple projects with different modules and dependencies using pip and virtual environments.

Go to the target directory where the virtual environment should be created and then run:

```
$ python3.6 -m venv sp_virtual_environment
```

`sp_virtual_environment` is the name of the environment, which is basically just a directory. It's a good practice to use the explicit Python version number when running the command. Activate the virtual environment:

```
$ source sp_virtual_environment/bin/activate
```

Once the virtual environment is activated , the prompt will change to show that you are working in the context of a virtual environment. Example:

```
(sp_virtual_environment) $
```

By default, virtual environments will not use any system installed modules. From an isolation perspective and for creating reproducible environments this is considered the correct behaviour. There is one exception to it, which is the package manager `pip`. Usually, `pip` should be upgraded before installing any modules:

```
(sp_virtual_environment) $ pip install --upgrade pip
```

Please note that this will only affect the pip version in the virtual environment. The required modules can now be installed, e.g.:

```
(sp_virtual_environment) $ pip install numpy
```

Connect the virtual environment to the Safer Payments Python execution context

There can be many virtual environments on a system. The one to be used by Safer Payments needs to be referenced. Only modules/dependencies that are available in the virtual environment will be found and loaded by Safer Payments. A missing dependency will result in an error message during the upload of a script. The following lines need to be added to the `systemd` startup script, just above the `ExecStart` line. The paths need to be adjusted:

```
Environment="VIRTUAL_ENV=/home/SPUser/sp_virtual_environment"
Environment="PATH=/home/SPUser/sp_virtual_environment/bin"
ExecStart=/usr/bin/iris rootpath=/opt/ibm/safer_payments id=1
```

With this the setup is completed.

Install Python3 on RHEL <7.7

This section is only relevant for installations with RHEL versions < 7.7. It describes how to use newer versions of Python3 (>=3.5) in a way that is officially supported by Red Hat Enterprise Linux. RHEL7 is developed and tested on an old version of Python (2.7) and all of its base utilities like `yum` rely on this version and its installed packages. Most of the instructions that can be found on the internet about installing newer Python versions "overwrite" the default Python version and might break the operating system. The official way uses Red Hat Software Collections. This installs Python concurrently in an non-

intrusive way. Sane but unsupported alternatives are to use the EPEL repositories or to compile Python yourself.

Installation

First the SCL (software collections) repository must be activated before installing Python 3. In this example Python 3.5 will be installed:

```
su -
# subscription-manager repos --enable rhel-7-server-optional-rpms --enable
rhel-server-rhscl-7-rpms
# yum -y install rh-python35
```

Packages built as a software collection do not overwrite the versions (base tools) included with RHEL. They are installed under `/opt/rh` and aren't automatically added to your `PATH` and `LD_LIBRARY_PATH`. That's why newer Python versions can coexist with the existing base installation. The command `scl enable` will make the necessary changes to the environment.

```
$ scl enable rh-python35 bash
```

Verify it's working:

```
$ python3 -V
```

Once installed, it is necessary to make the Python libraries accessible to Safer Payments. This can be done with a one-time command that creates a symbolic link:

```
$ sudo ln -s /opt/rh/rh-python35/root/lib64/libpython3.5m.so
/usr/lib64/libpython3.5.so
```

It's not possible to expose the library using `LD_LIBRARY_PATH` when Safer Payments is run using a systemd script, because the Safer Payments executable uses Linux capabilities. For security reasons Linux won't pass this environment variable to Safer Payments.