Upgrading Custom Action's Python Version

Written by David Krasnitsky @ November 2019

Introduction

QRadar 7.3.1 & 7.3.2 are using Python 2.7.5 (Red Hat's system default) as a platform for running custom actions script, althought their documentation states that python 2.7.9 is the one that is used (which is also the minimum recommended version for QRadar App SDK).

Since this isn't the case, and python 2.7.5 is missing some crucial functionality (Namely, the ability to access REST API using secure TLS connection & SSLContext while there may be more stuff missing..)

Disclaimer

This guide isn't official, and my solution is "hacky". It was tested many times on the QRadar 7.3.1 Community Edition and verified to have worked without any problems. However I strongly recommend you create a snapshot backup of your system before you do anything that is written and/or described in here.

Also, as long as you do not remove any system files and as long as you don't mess around *wrongly* with the file `/etc/fstab`, then any damage can be fixed by simply restarting the QRadar appliance.

And finally: ***You are fully responsible for your own actions – use good judgment as you use this unofficial guide. By reading this guide you agree that you are completely responsible for any potential damage that may accure when practicing this guide and that you have no; and cannot have any complaints against this guide's author(s)***.

Technical Background

QRadar `Custom Action` functionality allows to upload a script of one of the three programming language & environments: **Bash**, **Perl** & **Python**, and run it with given arguments (**Please refer the next official guide for more**: https://www.ibm.com/support/knowledgecenter/en/SS42VS_7.3.2/com.ibm.appfw.doc/c_appframework_CREResp.html).

Once a *Custom Action* is defined, you are then required by QRadar to *"Deploy Changes"*. Once the changes are *deployed*, your script is placed in a *virtual, sandboxed environment* which is in practice a form of `chroot` command executed on the path: `/opt/qradar/bin/ca_jail` while the script itself is saved inside: `/opt/qradar/bin/ca_jail/custom_action_scripts` with the following naming convention: `customaction_N.script` where **N** is the script number. Take notice that it doesn't matter rather your script was originally a **Bash**, **Perl** or **Python** script, they all loose their original name and file extension.

Also worth mentioning: *QRadar is starting its **Custom Actions** with its own environmental variables which cannot be pre-configured for its running scripts.*

Now here is where our problems begin: It seems that the virtual root environment (aka `chroot`) has its `/bin` and `/usr/bin` directories binded by `mount --bind` command to the main system's (aka **global**) `/bin` and `/usr/bin`, while it really should be using its own local ones (Seems like lazy Ops job?). Red Hat's / CentOS' `/bin` and `/usr/bin` are containing a default *Python 2.7.5* version. The default Python version ***should NOT be upgraded!*** as it may very well cause unexpected problems with some operating system tools, particallary tools that use the `yum` tool.

Finally, the consistent *mount binds* are configured by and inside the file: `/etc/fstab`.

Upgrade Overview

These are the general steps we are about to take, before we dive into the specifics.

1. The very first thing we've got to do is download and compile the latest **Python** version. At the moment of typing this very words, Python 2 is about to turn EOF ("End-Of-Life") and will no longer be supported! The latest version (and most probably the last of Python 2) is ***Python 2.7.17***.

   To compile and produce proper Python binaries, we must use either the same Red Hat version of our QRadar or a paraller version of CentOS (Red Hat & CentOS share the same binaries while CentOS is non-commerical and free).

   Check out the proper version of CentOS for a Red Hat alternative: https://en.wikipedia.org/wiki/CentOS

   - Once you know which version is needed, you can either download the iso file of that operating system or you could use a docker image.
   - In my examples, I'll be using a docker image of CentOS-7.5-1804 and compile everything within a container.
   - But the idea is the same: *Python 2.7.17 compiled within an identical environment to the one of your QRadar appliance.*

CentOS Version	RHEL Base
7.0-1406	7.0
7.1-1503	7.1
7.2-1511	7.2

CentOS Version	RHEL Base
7.3-1611	7.3
7.4-1708	7.4
7.5-1804	7.5
7.6-1810	7.6
7.7-1908	7.7

2. Copy our compiled **Python 2.7.17** from the previous section to the **QRadar** appliance and set it up correctly under `/opt/qradar/bin/ca_jail` and merge IBM's `site-packages` to it.
3. Create an overall system snapshot just in case something goes wrong.
4. Modifiy our `/etc/fstab` to remove the two persistent mounts to the sandboxed environments `/bin` and `/usr/bin`.
5. Unmount & Recreate `/opt/qradar/bin/ca_jail/bin` and `/opt/qradar/bin/ca_jail/usr/bin` to contain everything from the global `/bin` and `/usr/bin`.
6. Copy **Python 2.7.17** binaries to `/opt/qradar/bin/ca_jail/bin` and `/opt/qradar/bin/ca_jail/usr/bin`.
7. We're done! All python scripts should now run with Python 2.7.17.
    - We will upload a short test script to verify that everything is working well.

Setup Environment

In our example, we're going to use docker since my own workstation is running on Linux with Docker already installed.

If you're not using Docker or Linux, just skip all the Docker parts and focus on the CentOS/Red Hat environment parts which you may either install on a virtual machine or bare metal.

Docker: Download CentOS Image

```
$ docker pull centos:centos7.5.1804
```

Docker: Create & Run CentOS Container

```
docker run -it --name "CentOS-for-QRadar" centos:centos7.5.1804
```

- **Notice:** I do not use the `--rm` switch because I personally like the idea that I can resume my container later from where I left off. But you may have a different agenda.
- If you exit your container and would like to resume it later:
    - Use `$ docker ps -a` and find the container ID of `CentOS-for-QRadar`
    - Use the next command to resume its run:

      ```
      $ docker container start -i <container ID>
      ```

    - Remove the container (if you wish):

      ```
      $ docker rm <container ID>
      ```

        - Just make sure the container isn't running.

Update CentOS/RedHat Environment

All commands assume you are running as `root`, which as marked as #.

However if you're in normal mode, marked as $, just add `sudo` to the start of each command.

Run the following commands:

```
# yum check-update && yum update
```

- Answer *"YES"* / *"Y"* to any prompt.

Now we need to install updated GCC compilers with some basic tools for CentOS/RedHat:

```
# yum install wget make gcc openssl-devel bzip2-devel
```

- Answer *"YES"* / *"Y"* to any prompt.

Download & Compile The Latest Python 2

Assuming you are already running a setup of the appropriate Red Hat or CentOS to compile the latest **Python 2**, we will start with the next set of commands:

1. For convenient reasons, lets work in the home directory of the current user.

   `# cd ~`

   - **Notice:** If you're running with the *root* user, you may need to make sure that the directory `/root` exists or else you may get an error. Just call `mkdir /root` and try to `cd ~` again.

2. Since Python 2 is about to reach its "End-Of-Life" status and will no longer be supported, it is safe to assume that **Python2.7.17** is also the last version of Python 2 to be released (unless they release something right before they kill it).

   Because of this, I'll be using a direct download link to Python2.7.17. However if you wish to download a different version, just refer to: https://www.python.org/downloads/

   Download Python-2.7.17:

   `# wget https://www.python.org/ftp/python/2.7.17/Python-2.7.17.tgz`

3. When done downloading, it is time to extract the Python-2.7.17 installation:

   `# tar zxvf Python-2.7.17.tgz`

   - A new directory will be created: `Python-2.7.17`

4. Get to the `Python-2.7.17` directory:

   `# cd Python-2.7.17`

5. Set Python-2.7.17 setup configurations:

   `# ./configure --prefix=/python-2.7.17 --with-ensurepip=install --enable-optimizations`

   - **Worth mentioning:** We are installing Python to the `/python-2.7.17` directory so its natural path will be under the root `/` directory as this will be the case in the future QRadar virtual environment.

6. Install Python-2.7.17:

   `# make && make install`

   - A new directory will be created: `/python-2.7.17`

7. Lets change to the root directory:

   `# cd /`

8. Time to pack our build into a `tar/gzip` file:

   `# tar -czvf python-2.7.17-compiled.tar.gz python-2.7.17`

   - A new file will be created: `python-2.7.17-compiled.tar.gz`

Copy The Compiled Python to QRadar

1. Retrieve the file `python-2.7.17-compiled.tar.gz` from your setup machine\container.

Non-Docker Users:

- You will have to find a way to do it according to the method that you chose: *virtual machine* or *bare metal*.

Docker Users:

1. Open a new terminal on your Linux machine: `CTRL` + `ALT` + `T` is the default shortcut to open a terminal window.
2. You'll need your container ID (if you forgot it) which can be found within the container itself.

   Type this within the container's shell:

   `# hostname`

   - By default, you should see the container's ID at the bash prompt: `[root@<container ID> /]#`

   or, use this command outside your container's shell to find out what is your container's ID:

   `$ docker ps -a`

   - If you've followed this manual, look for a container named `CentOS-for-QRadar`.

3. Make sure the container is running.
    - If you've exit your container, use the next command to resume its run:

      ```
      $ docker container start -i <container ID>
      ```

4. Copy the `python-2.7.17-compiled.tar.gz` file from your container to your local Linux machine:

   ```
   $ docker cp <container ID>:/python-2.7.17-compiled.tar.gz ~
   ```

    - You should now have the file `python-2.7.17-compiled.tar.gz` placed in your home directory `~`.

Copy python-2.7.17-compiled.tar.gz to QRadar

- Use your favorit tool and\or method: `WinSCP`, `FileZilla`, etc.

Extract Python to QRadar

1. Login into QRadar terminal\shell environment, either directly or with your favorit SSH tool.
2. Get into the directory ("*cd*") where you previously copied the file `python-2.7.17-compiled.tar.gz` to QRadar.
3. Extract the compiled **Python 2.7.17**:

   ```
   $ tar xvzf python-2.7.17-compiled.tar.gz
   ```

    - The directory `python-2.7.17` should now appear.

4. Move the `python-2.7.17` directory to the QRadar's sandboxed environment:

   ```
   # mv python-2.7.17 /opt/qradar/bin/ca_jail
   ```

    - The directory `python-2.7.17` should now be appear under `/opt/qradar/bin/ca_jail`. Full path: `/opt/qradar/bin/ca_jail/python-2.7.17`

Merge IBM Python Dependencies

```
# cp -R -n /usr/lib/python2.7/site-packages/* /opt/qradar/bin/ca_jail/python-2.7.17/lib/python2.7/site-packages
```

```
# cp -R -n /opt/qradar/lib/python/* /opt/qradar/bin/ca_jail/python-2.7.17/lib/python2.7/site-packages
```

Backup QRadar

It's time to create a snapshot out of our QRadar appliance system as we are about to touch some sensitive and dangerous files!

Setup Sandbox Environment

First, we need to turn the sandbox environment to become more indepeneded from its main QRadar appliance.

1. Backup `/etc/fstab`:

   ```
   # cp /etc/fstab /etc/fstab_backup
   ```

2. Remove persistent mounts of `/bin` and `/usr/bin` to `/opt/qradar/bin/ca_jail/bin` and `/opt/qradar/bin/ca_jail/usr/bin` from `/etc/fstab`

   ```
   # vi /etc/fstab
   ```

    - Press `INSERT` to edit.

   /etc/fstab:

   ```
   #
   # /etc/fstab
   # Created by anaconda on Sun Nov 17 18:33:14 2019
   #
   # Accessible filesystems, by reference, are maintained under '/dev/disk'
   # See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info
   #
   /dev/mapper/centos-root /                       xfs        noatime              0 0
   UUID=e8f7c4c5-3cc7-4215-a143-b92227ce7150 /boot                   xfs        defaults             0 0
   /dev/mapper/centos-home /home                   xfs        defaults             0 0
   /dev/mapper/centos-swap swap                    swap       defaults             0 0
   /bin                /opt/qradar/bin/ca_jail/bin none       bind
   /lib                /opt/qradar/bin/ca_jail/lib none       bind
   /lib64              /opt/qradar/bin/ca_jail/lib64 none       bind
   ```

```
/usr/bin             /opt/qradar/bin/ca_jail/usr/bin none       bind
/usr/lib             /opt/qradar/bin/ca_jail/usr/lib none       bind
/usr/lib64           /opt/qradar/bin/ca_jail/usr/lib64 none       bind
/usr/share           /opt/qradar/bin/ca_jail/usr/share none       bind
/opt/qradar/conf/custom_action_scripts /opt/qradar/bin/ca_jail/custom_action_scripts none       bind
sysfs                /sys                sysfs      rw                  0 0
/dev/cdrom      /media/cdrom    auto    pamconsole,exec,noauto 0 0
```

- Comment the line `/bin /opt/qradar/bin/ca_jail/bin none bind` by adding # sign at the beginning of the line.
- The result should be: `#/bin /opt/qradar/bin/ca_jail/bin none bind`
- Comment the line `/usr/lib /opt/qradar/bin/ca_jail/usr/lib none bind` by adding # sign at the beginning of the line.
- The result should be: `#/usr/lib /opt/qradar/bin/ca_jail/usr/lib none bind`

Final Result:

/etc/fstab:

```
#
# /etc/fstab
# Created by anaconda on Sun Nov 17 18:33:14 2019
#
# Accessible filesystems, by reference, are maintained under '/dev/disk'
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info
#
/dev/mapper/centos-root /                       xfs        noatime              0 0
UUID=e8f7c4c5-3cc7-4215-a143-b92227ce7150 /boot                   xfs        defaults             0 0
/dev/mapper/centos-home /home                   xfs        defaults             0 0
/dev/mapper/centos-swap swap                    swap       defaults             0 0
#/bin                 /opt/qradar/bin/ca_jail/bin none       bind
/lib                 /opt/qradar/bin/ca_jail/lib none       bind
/lib64               /opt/qradar/bin/ca_jail/lib64 none       bind
#/usr/bin             /opt/qradar/bin/ca_jail/usr/bin none       bind
/usr/lib             /opt/qradar/bin/ca_jail/usr/lib none       bind
/usr/lib64           /opt/qradar/bin/ca_jail/usr/lib64 none       bind
/usr/share           /opt/qradar/bin/ca_jail/usr/share none       bind
/opt/qradar/conf/custom_action_scripts /opt/qradar/bin/ca_jail/custom_action_scripts none       bind
sysfs                /sys                sysfs      rw                  0 0
/dev/cdrom      /media/cdrom    auto    pamconsole,exec,noauto 0 0
```

3. When you have verified that the changes are correct, save the file and exit `vi`:
   - Press ESC
   - Type `:wq`
   - Press ENTER
4. Unmount `/opt/qradar/bin/ca_jail/bin` and `/opt/qradar/bin/ca_jail/usr/bin`:

   `# umount /opt/qradar/bin/ca_jail/bin`

   `# umount /opt/qradar/bin/ca_jail/usr/bin`
5. Copy all contents of `/bin` and `/usr/bin` to `/opt/qradar/bin/ca_jail/bin` and `/opt/qradar/bin/ca_jail/usr/bin`:

   `# cp -R /bin/* /opt/qradar/bin/ca_jail/bin`

   `# cp -R /usr/bin/* /opt/qradar/bin/ca_jail/usr/bin`
6. Replace sandbox **Python 2.7.5** with **Python 2.7.17**:

   `# \cp -R /opt/qradar/bin/ca_jail/python-2.7.17/bin/* /opt/qradar/bin/ca_jail/bin`

   `# \cp -R /opt/qradar/bin/ca_jail/python-2.7.17/bin/* /opt/qradar/bin/ca_jail/usr/bin`
7. We're done! Now it's time to verify that the upgrade was indeed successful.

Verify Python Version

Add, upload and **test the execution** of the next code snippet as a Python file *(.py)* to find out which Python version is currently set to run in your *Custom-Action* sandbox environment:

```
import sys
print "Python", sys.version
```

Rollback & Switching Between Python Versions

Go Back to the default Python 2.7.5

If you wish to rollback to the default Python 2.7.5 (for no special reason where you do not have to rollback by snapshot), all you need is to reassign the mounts in `/etc/fstab` and then # `mount --all` the sandbox's directories: `/opt/qradar/bin/ca_jail/bin` and `/opt/qradar/bin/ca_jail/usr/bin` to `/bin` and `/usr/bin` respectively.

Reasign Persistent Mounts

Uncomment the two lines from the previous **"Setup Sandbox Environment"** section, *step 2* in the file `/etc/fstab`.

/etc/fstab:

```
#
# /etc/fstab
# Created by anaconda on Sun Nov 17 18:33:14 2019
#
# Accessible filesystems, by reference, are maintained under '/dev/disk'
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info
#
/dev/mapper/centos-root /                       xfs        noatime              0 0
UUID=e8f7c4c5-3cc7-4215-a143-b92227ce7150 /boot                   xfs        defaults             0 0
/dev/mapper/centos-home /home                   xfs        defaults             0 0
/dev/mapper/centos-swap swap                    swap       defaults             0 0
#/bin                  /opt/qradar/bin/ca_jail/bin none       bind
/lib                  /opt/qradar/bin/ca_jail/lib none       bind
/lib64                /opt/qradar/bin/ca_jail/lib64 none       bind
#/usr/bin              /opt/qradar/bin/ca_jail/usr/bin none       bind
/usr/lib              /opt/qradar/bin/ca_jail/usr/lib none       bind
/usr/lib64            /opt/qradar/bin/ca_jail/usr/lib64 none       bind
/usr/share            /opt/qradar/bin/ca_jail/usr/share none       bind
/opt/qradar/conf/custom_action_scripts /opt/qradar/bin/ca_jail/custom_action_scripts none       bind
sysfs                 /sys                  sysfs      rw                   0 0
/dev/cdrom      /media/cdrom    auto    pamconsole,exec,noauto 0 0
```

Uncomment by removing the # sign from the beginning of the lines:

1. `#/bin /opt/qradar/bin/ca_jail/bin none bind`
2. `#/usr/bin /opt/qradar/bin/ca_jail/usr/bin none bind`

The final result should look like this:

/etc/fstab:

```
#
# /etc/fstab
# Created by anaconda on Sun Nov 17 18:33:14 2019
#
# Accessible filesystems, by reference, are maintained under '/dev/disk'
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info
#
/dev/mapper/centos-root /                       xfs        noatime              0 0
UUID=e8f7c4c5-3cc7-4215-a143-b92227ce7150 /boot                   xfs        defaults             0 0
/dev/mapper/centos-home /home                   xfs        defaults             0 0
/dev/mapper/centos-swap swap                    swap       defaults             0 0
/bin                  /opt/qradar/bin/ca_jail/bin none       bind
/lib                  /opt/qradar/bin/ca_jail/lib none       bind
/lib64                /opt/qradar/bin/ca_jail/lib64 none       bind
/usr/bin              /opt/qradar/bin/ca_jail/usr/bin none       bind
/usr/lib              /opt/qradar/bin/ca_jail/usr/lib none       bind
/usr/lib64            /opt/qradar/bin/ca_jail/usr/lib64 none       bind
/usr/share            /opt/qradar/bin/ca_jail/usr/share none       bind
/opt/qradar/conf/custom_action_scripts /opt/qradar/bin/ca_jail/custom_action_scripts none       bind
sysfs                 /sys                  sysfs      rw                   0 0
```

```
/dev/cdrom      /media/cdrom    auto    pamconsole,exec,noauto 0 0
```

Mount Everything Back

```
# mount --all
```

- You have now rolled-back the sandbox environment to Python 2.7.5!

Go Back to the latest Python 2.7.17

If you wish to get the sandbox environment back to the latest Python again, just repeat *steps 1 to 4* in the **"Setup Sandbox Environment"** section.

Best of luck!