independIT Integrative Technologies GmbH

Setting up schedulix without a direct Internet connection

20. Februar 2019

Starting point

For the procedure described here it is assumed that there is a machine available that isn't directly connected to the Internet. This machine is equiped with a CentOS7, minimal installation.
The purpose of this document is to describe all steps to install the schedulix server without connecting the machine to the Internet. Obviously some other computer must have access to both the Internet and the new machine.
After the initial installation and a subsequent update of CentOS, the filesystems look like:

```
[root@kitten ~]# df -k
Filesystem          1K-blocks    Used Available Use% Mounted on
/dev/mapper/ca-root  52403200 1729116  50674084   4% /
devtmpfs             32412928       0  32412928   0% /dev
tmpfs                32459456       0  32459456   0% /dev/shm
tmpfs                32459456   13696  32445760   1% /run
tmpfs                32459456       0  32459456   0% /sys/fs/cgroup
/dev/vda2             1038336  218980    819356  22% /boot
/dev/mapper/ca-home 344804868   32944 344771924   1% /home
tmpfs                 6491904       0   6491904   0% /run/user/0
```

That's basically a 50GB root filesystem and a 340GB filesystem for the users. (To be a little more exact, it's a 384GB virtual disk with 402653184 1K blocks in total).
For my convenience the package `net-tools` was installed as well.
In order to cut the connection to the Internet it suffices to remove the default gateway. The routing table looks like:

```
[root@kitten ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.25.1    0.0.0.0         UG    100    0        0 eth0
192.168.25.0    0.0.0.0         255.255.255.0   U     100    0        0 eth0
```

After removing the default gateway only the local network can be reached:

```
[root@kitten ~]# route del default gw 192.168.25.1
[root@kitten ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.25.0    0.0.0.0         255.255.255.0   U     100    0        0 eth0
```

This ends the initial setup of the scheduling server.

Getting the required rpms

In order to be able to install the schedulix server, a bunch of rpms are required. Hence the first step is to retrieve those rpms from some source and to store them on the target machine. In this tutorial the rpms will be stored in `/root/rpms`, but any directory will do.

Apart from the schedulix rpms, the following packages are required as well:

Package	Version	Package	Version
apache-commons-lang	2.6-15.el7	apache-commons-logging	1.1.2-7.el7
avalon-framework	4.3-10.el7	avalon-logkit	2.1-14.el7
cal10n	0.7.7-4.el7	copy-jdk-configs	3.3-10.el7_5
cpp	4.8.5-36.el7	gcc	4.8.5-36.el7
geronimo-jms	1.1.1-19.el7	geronimo-jta	1.1.1-17.el7
giflib	4.1.6-9.el7	glibc-devel	2.17-260.el7
glibc-headers	2.17-260.el7	java-1.8.0-openjdk	1.8.0.191.b12-1.el7_6
java-1.8.0-openjdk-headless	1.8.0.191.b12-1.el7_6	javamail	1.4.6-8.el7
javapackages-tools	3.4.1-11.el7	javassist	3.16.1-10.el7
jna	3.5.2-8.el7	kernel-headers	3.10.0-957.1.3.el7
libICE	1.0.9-9.el7	libSM	1.2.2-2.el7
libXtst	1.2.3-1.el7	libatomic	4.8.5-36.el7
libfontenc	1.1.3-3.el7	libmpc	1.0.1-3.el7
libxslt	1.1.28-5.el7	lksctp-tools	1.0.17-2.el7
log4j	1.2.17-16.el7_4	mariadb	5.5.60-1.el7_5
mariadb-server	5.5.60-1.el7_5	mpfr	3.1.1-4.el7
mysql-connector-java	5.1.25-3.el7	perl-Compress-Raw-Bzip2	2.061-3.el7
perl-Compress-Raw-Zlib	2.061-4.el7	perl-DBD-MySQL	4.023-6.el7
perl-DBI	1.627-4.el7	perl-IO-Compress	2.061-2.el7
perl-Net-Daemon	0.48-5.el7	perl-PlRPC	0.2020-14.el7
python-backports	1.0-8.el7	python-backports-ssl_match_hostname	3.5.0.1-1.el7
python-devel	2.7.5-76.el7	python-ipaddress	1.0.16-2.el7
python-javapackages	3.4.1-11.el7	python-lxml	3.2.1-4.el7
python-setuptools	0.9.8-7.el7	python-virtualenv	15.1.0-2.el7
slf4j	1.7.4-4.el7_4	tomcat-servlet-3.0-api	7.0.76-8.el7_5
ttmkfdir	3.0.9-42.el7	tzdata-java	2018i-1.el7
wget	1.14-18.el7	xalan-j2	2.7.1-23.el7
xerces-j2	2.11.0-17.el7_0	xml-commons-apis	1.4.01-16.el7
xml-commons-resolver	1.2-15.el7	xorg-x11-font-utils	7.5-21.el7
xorg-x11-fonts-Type1	7.5-9.el7		

This list of packages will change over time when new releases are published. The easiest way to get the actual list of required packages is to start with a freshly installed CentOS or RHEL (minimal installation). A `yum install schedulix-server-mariadb` will then produce the list of required packages. The installation of the schedulix-server package can be aborted at that point.
The following script would do the job (assuming that the workstation with internet access is capable of executing bash scripts and has `wget` installed). Depending on the target system the ARCH and MIRROR variables have to be adjusted to match the target system's architecture.

```
#!/bin/bash

ARCH=ppc64le
MIRROR=http://mirror.centos.org/altarch/7/os/$ARCH/Packages

for f in \
apache-commons-lang-2.6-15.el7 \
apache-commons-logging-1.1.2-7.el7 \
avalon-framework-4.3-10.el7 \
avalon-logkit-2.1-14.el7 \
cal10n-0.7.7-4.el7 \
copy-jdk-configs-3.3-10.el7_5 \
cpp-4.8.5-36.el7 \
gcc-4.8.5-36.el7 \
geronimo-jms-1.1.1-19.el7 \
geronimo-jta-1.1.1-17.el7 \
giflib-4.1.6-9.el7 \
glibc-devel-2.17-260.el7 \
glibc-headers-2.17-260.el7 \
java-1.8.0-openjdk-1.8.0.181-7.b13.el7 \
java-1.8.0-openjdk-headless-1.8.0.181-7.b13.el7 \
javamail-1.4.6-8.el7 \
javapackages-tools-3.4.1-11.el7 \
javassist-3.16.1-10.el7 \
jna-3.5.2-8.el7 \
kernel-headers-3.10.0-957.el7 \
libICE-1.0.9-9.el7 \
libSM-1.2.2-2.el7 \
libXtst-1.2.3-1.el7 \
libatomic-4.8.5-36.el7 \
libfontenc-1.1.3-3.el7 \
libmpc-1.0.1-3.el7 \
libxslt-1.1.28-5.el7 \
lksctp-tools-1.0.17-2.el7 \
log4j-1.2.17-16.el7_4 \
mariadb-5.5.60-1.el7_5 \
mariadb-server-5.5.60-1.el7_5 \
mpfr-3.1.1-4.el7 \
mysql-connector-java-5.1.25-3.el7 \
perl-Compress-Raw-Bzip2-2.061-3.el7 \
perl-Compress-Raw-Zlib-2.061-4.el7 \
perl-DBD-MySQL-4.023-6.el7 \
perl-DBI-1.627-4.el7 \
perl-IO-Compress-2.061-2.el7 \
perl-Net-Daemon-0.48-5.el7 \
perl-PlRPC-0.2020-14.el7 \
python-backports-1.0-8.el7 \
python-backports-ssl_match_hostname-3.5.0.1-1.el7 \
python-devel-2.7.5-76.el7 \
python-ipaddress-1.0.16-2.el7 \
python-javapackages-3.4.1-11.el7 \
python-lxml-3.2.1-4.el7 \
python-setuptools-0.9.8-7.el7 \
python-virtualenv-15.1.0-2.el7 \
slf4j-1.7.4-4.el7_4 \
tomcat-servlet-3.0-api-7.0.76-7.el7_5 \
ttmkfdir-3.0.9-42.el7 \
tzdata-java-2018e-3.el7 \
wget-1.14-18.el7 \
xalan-j2-2.7.1-23.el7 \
xerces-j2-2.11.0-17.el7_0 \
xml-commons-apis-1.4.01-16.el7 \
xml-commons-resolver-1.2-15.el7 \
xorg-x11-font-utils-7.5-21.el7 \
xorg-x11-fonts-Type1-7.5-9.el7; do
        if [ -f ${f}.noarch.rpm -o -f ${f}.${ARCH}.rpm ]; then
                echo "skipping $f; present already"
        else
                if ! wget "${MIRROR}/${f}.noarch.rpm"; then
                        wget "${MIRROR}/${f}.${ARCH}.rpm"
                        echo "${MIRROR}/${f}.${ARCH}.rpm"
                else
                        echo "${MIRROR}/${f}.noarch.rpm"
                fi
        fi
done
```

After retrieving all the packages, they can now be copied to the target system.

The installation Part I

After the retrieval of the rpms, it's now time to start the installation. Basically we can install everything except for the schedulix-zope package and the schedulix-examples package. The first one requires a lot of python packages (eggs, wheels, ...), the second one the eclipse-swt. And the schedulix-server-pg package conflicts with the schedulix-server-mariadb package.
Hence we mv those two aside and remove the server-pg package:

```
[root@kitten rpms]# mv schedulix-examples-2.8-18.el7.noarch.rpm \
schedulix-zope-2.8-18.el7.ppc64le.rpm ..
[root@dxc rpms]# rm schedulix-server-pg-2.8-18.el7.ppc64le.rpm
rm: remove regular file 'schedulix-server-pg-2.8-18.el7.ppc64le.rpm'? y
```

And now we can simply install all the packages within the directory.

```
yum install *
```

The installation Part II

The component still missing is the Zope server. This piece of software is written in Python and requires an impressive list of Python packages. The file

```
    /opt/schedulix/schedulix/etc/zope_requirements-2.13.26.txt
```

lists all the required components.
In the "online" installation procedure this file is used by `pip` to retrieve and install all the packages. In an "offline" installation this has to be done in two steps:

1. Download of the packages
2. Installation of the downloaded packages

The following commands, executed on the machine with Internet access, will download all the packages and create an archive that can be copied to the target system. It is assumed that `pip` is found within the PATH.

```
cd /opt/schedulix
mkdir python_pkgs
cd python_pkgs
pip install download -r ../schedulix/etc/zope_requirements-2.13.26.txt
cd ..
tar cvzf python_pkgs.tgz python_pkgs
```

After copying the tgz to the target system (it is assumed that the file is copied to the `/tmp` directory), the second part of the installation can begin. First a Python virtual environment, as user schedulix, is created:

```
cd /opt/schedulix
mkdir software
cd software
virtualenv --no-side-packages Zope
```

This also creates a `pip` executable below `software/Zope/bin` which can be used for the consecutive steps. But we'll need to unpack our tgz file first:

```
cd /opt/schedulix
tar xvzf /tmp/python_pkgs.tgz
```

After unpacking the archive, the Python packages can be installed in the virtual environment:

```
cd /opt/schedulix/software/Zope
bin/pip install --no-index --find-links file:/opt/schedulix/python_pkgs PasteDeploy==1.3.4
bin/pip install --no-index --find-links file:/opt/schedulix/python_pkgs Paste==1.7.5.1
bin/pip install --no-index --find-links file:/opt/schedulix/python_pkgs -r \
    /opt/schedulix/schedulix/etc/zope_requirements-2.13.26.txt
```

Unfortunately `pip` doesn't always work flawlessly. While testing this procedure it turned out to be necessary to install two of the packages seperately.
Now we have the Zope Application server software installed. We need to do a few other things. First of all we'll have to create an instance:

```
bin/mkzopeinstance -d /opt/schedulix/schedulixweb -u sdmsadm:sdmsadm
```

This creates the instance with one administrator called sdmsadm identified by sdmsadm. You can choose another password here of course, or you can change the password any time later.
Now we have the instance, we're going to need the software that runs within the instance. This software is present in the schedulix-zope rpm. And fortunately rpm knows an option that prevents the execution of the pre- and post-scripts.
As root you execute

```
    rpm -i --noscripts schedulix-zope-2.8-18.el7.ppc64le.rpm
```

The rest is done as user schedulix.

```
cd /opt/schedulix/schedulixweb
mkdir Extensions
cd Extensions
ln -s /opt/schedulix/schedulix/zope/*.py .
cd ../Products
ln -s /opt/schedulix/schedulix/zope/BICsuiteSubmitMemory .
cd ../import
ln -s /opt/schedulix/schedulix/zope/SDMS.zexp .
```

And now we're almost done. We now start the Zope server and perform the last few remaining actions.

```
/opt/schedulix/schedulixweb/bin/zopectl start
```

This starts the Zope server. Later we'll use the init.d script which starts the process slightly differently. The next operations can either be done with the help of `wget`, or from within the browser. Clearly the post-script of the package uses `wget`.

```
wget --user=sdmsadm --password=sdmsadm --output-document=/dev/null \
    "http://localhost:8080/manage_importObject?file=SDMS.zexp"
wget --user=sdmsadm --password=sdmsadm --output-document=/dev/null \
    --keep-session-cookies --save-cookies /tmp/cookies.txt \
    "http://localhost:8080/SDMS/Install?manage_copyObjects:method=Copy&ids:list=User&ids:list=Custom"
wget --user=sdmsadm --password=sdmsadm --output-document=/dev/null \
    --load-cookies /tmp/cookies.txt "http://localhost:8080?manage_pasteObjects:method=Paste"
rm /tmp/cookies.txt
```

Now everything's done (if you didn't get any strange error messages). But there's a finishing touch that wouldn't harm.
As user schedulix you shut down the Zope server

```
    /opt/schedulix/schedulixweb/bin/zopectl stop
```

As root, you enable and start the service:

```
    chkconfig schedulix-zope on
    service schedulix-zope start
```

Now it's time to open a bottle of champagne :-)