

# Offsite Installation

It may be advantageous or necessary to do data analysis without a connection to SLAC. Below are the steps to install psana along with the data to be used in the analysis onto a local machine for later, offline use on a RHEL 5, 6 or 7 machine. In the simplest case, a single conda environment is maintained but multiple conda environments can be created and used if necessary. The steps for both cases are shown below.

- [Installation of a Single Conda Environment](#)
- [Installation of Multiple Conda Environments](#)
- [Useful Commands](#)
- [Example](#)

## Installation of a Single Conda Environment

1. Use bash as your shell
2. Go to <https://conda.io/miniconda.html>
3. Install the Python 2.7 64 bit bash installer (unless your OS is 32 bit but it most likely will not be)
4. Find the installed file **Miniconda2-latest-Linux-x86\_64.sh**
  - a. It will probably be in Downloads, so do **cd Downloads**
  - b. Run it with **bash Miniconda2-latest-Linux-x86\_64.sh**
5. Add **miniconda2** to **PATH**
  - a. The installation will automatically add the bin subdirectory of the installation to PATH in your **.bashrc** file
  - b. Close and reopen terminal and check by typing **conda list** which will print out the installed packages
  - c. If **conda** is not found, follow step 5, else skip to step 6
6. If not already created, create a **.bash\_profile** file in your home directory
  - a. Add the following script to it. This checks for the **.bashrc** and will run it on start up

```
#!/bin/bash

if [ -f $HOME/.bashrc ]; then
    source $HOME/.bashrc
fi
```

7. Run **conda update -y conda** to update miniconda
8. To replicate the psana environment on your laptop, download the file with the version requirements for all packages.  
The files can be found on GitHub, at [this location](#).  
For example, at the time of writing, the most recent version of psana is 4.0.28. The file for this version of psana can be found [here](#) and can be downloaded using wget:  
**wget https://raw.githubusercontent.com/slaclab/anarel-manage/master/envfiles/ana-4.0.28.yml**
9. With the downloaded file, create the psana environment with the following command:  
**conda env create -n ana-4.0.28 -f ana-4.0.28.yml**
10. Activate the environment with:  
**conda activate ana-4.0.28**
11. The information in this point is only needed if a customized environment needs to be created.  
The above procedure creates a replica of the psana environment, as available at LCLS. The individual packages are available on the 'lcls-i' public channel on anaconda.org, so  
it is possible to create a personalized environment which mixes specific version of psana with custom version of other packages. However, be aware that not all versions  
of the python packages on conda-forge are compatible with each other, and this might limit the ability to mix-and-match.  
The list of channels needed to build a custom psana environment, with the correct priority, is the following:

```
- lcls-i
- conda-forge
- defaults
```

12. (optional, since we try to have an up-to-date copy of this in conda) Copy the experiment database from **/reg/g/psdm/data/ExpNameDb/experiment-db.dat**
  - a. Make a directory in the home directory with **mkdir -p psdm/data/ExpNameDb**
  - b. Copy the database
    - i. With rsync: **rsync -t psexport:/reg/g/psdm/data/ExpNameDb/experiment-db.dat ~psdm/data/ExpNameDb/**
    - ii. With SCP: **scp -p psexport:/reg/g/psdm/data/ExpNameDb/experiment-db.dat ~psdm/data/ExpNameDb/**
13. Copy the experiment data that will be used for analysis. This step requires patience if many runs will be copied
  - a. For example, downloading run 54 from the experiment **xpptut15**, the following steps were taken:
    - i. First create the required directories with **mkdir -p psdm/xpp/xpptut15/**
    - ii. Then these files were copied
      1. From **/reg/d/psdm/xpp/xpptut15/xtc** to **~/psdm/xpp/xpptut15/xtc**:

```
-bash-4.2$ ls /reg/d/psdm/xpp/xpptut15/xtc/ | grep 54
e665-r0054-s00-c00.xtc
e665-r0054-s01-c00.xtc
e665-r0054-s02-c00.xtc
e665-r0054-s03-c00.xtc
e665-r0054-s04-c00.xtc
e665-r0054-s05-c00.xtc
```

2. From `/reg/d/psdm/xpp/xpptut15/xtc/index` to `~/psdm/xpp/xpptut15/xtc/index`:

```
-bash-4.2$ ls /reg/d/psdm/xpp/xpptut15/xtc/index | grep 54
e665-r0054-s00-c00.xtc.idx
e665-r0054-s01-c00.xtc.idx
e665-r0054-s02-c00.xtc.idx
e665-r0054-s03-c00.xtc.idx
e665-r0054-s04-c00.xtc.idx
e665-r0054-s05-c00.xtc.idx
```

3. From `/reg/d/psdm/xpp/xpptut15/xtc/smalldata` to `~/psdm/xpp/xpptut15/xtc/smalldata`:

```
-bash-4.2$ ls /reg/d/psdm/xpp/xpptut15/xtc/smalldata | grep 54
e665-r0054-s00-c00.smd.xtc
e665-r0054-s01-c00.smd.xtc
e665-r0054-s02-c00.smd.xtc
e665-r0054-s03-c00.smd.xtc
e665-r0054-s04-c00.smd.xtc
e665-r0054-s05-c00.smd.xtc
```

4. And the entire directory `/reg/d/psdm/xpp/xpptut15/calib` to `~/psdm/xpp/xpptut15/calib`

b. Copy using either rsync or SCP. SCP may be simpler in this case because it copies the data of a symbolic link which is what is desired.

14. Two environment variables must be set, `SIT_DATA` and `SIT_PSDM_DATA` by adding the following commands to the `.bash_profile` file

a. (optional, see step (9)) `export SIT_DATA=$HOME/psdm/data`

b. `export SIT_PSDM_DATA=$HOME/psdm`

c. If step 5 was not done, create a `.bash_profile` file instead and begin it with `#!/bin/bash`

Now psana conda can be used. If new packages need to be added, removed, etc., the simple conda commands can be used. A link the specifics of these commands is given in the section below.

## Installation of Multiple Conda Environments

More information on managing Conda environments can be found [here](#). Begin by following the steps above for the single environment but do not install psana conda yet since it will be installed when the new environment is created (step 8) or set the environment variables (step 10). It is recommended to have these variables set when the environment is activated and unset when deactivated. First create the environment with **conda create --name <environment name> -c lcls-rhel<num> psana-conda** where **<environment name>** is the name of the environment to be created and **<num>** is the RHEL version. Then follow the steps below, for a given environment **exemplenv**, which were pulled from the link given before.

1. Change directories to the environment directories like `cd ~/miniconda2/env/exemplenv` (the actual path may be slightly different but it will be under **env** in the conda package)
2. Create the following directories and files:
  - a. `mkdir -p etc/conda/activate.d`
  - b. `mkdir -p etc/conda/deactivate.d`
  - c. `touch etc/conda/activate.d/env_vars.sh`
  - d. `touch etc/conda/deactivate.d/env_vars.sh`
3. Add the following to this files
  - a. The complicated piece is that `SIT_DATA` must include the sub-directory 'data' to your Conda environment, as well as where the `experiment-db.dat` file is. To `etc/conda/activate.d/env_vars.sh`:

```
#!/bin/sh

# Make sure to replace exemplenv with the actual environment name
export SIT_DATA=$HOME/miniconda2/envs/exemplenv/data:$HOME/psdm/data
export SIT_PSDM_DATA=$HOME/psdm
```

b. To `etc/conda/deactivate.d/env_vars.sh`:

```
#!/bin/sh

unset SIT_DATA
unset SIT_PSDM_DATA
```

Now **exampenv** will have the correct environment variables when activated with the command **source activate exampenv** and deactivate with **source deactivate**. And packages may be added, removed, upgraded, downgraded, etc. from this conda environment like usual. For information on this, follow link given at the beginning of this section.

## Useful Commands

To copy files from one machine to another that aren't huge (i.e. hundreds of GBs), rsync and SCP are good choices. Their basic use is like so:

- With rsync: **rsync -rt <machine copying from>:<file/directory name being copied> <directory copying to>**. The **-r** tag will copy recursively, so it is necessary if a directory is being copied but not necessary if only a file is being copied. The **-t** tag preserves the modification times.
- With SCP: **scp -rp <machine copying from>:<file/directory name being copied> <directory copying to>**. Like with rsync, the **-r** tag copies recursively and the **-p** tag has the same functionality as rsync's **-t** tag.

Below are several useful Conda commands that may have been stated above:

- To update a package, simply use **conda update <package name>** and this includes psana conda. So if psana conda needs to be updated, just do **conda update -y psana-conda**. The **-y** tag removes the "Proceed ([y]/n)?" question that is asked before installing/updating or whatever is being done.
- Packages can be similarly be installed and uninstalled replacing **update** with **install** or **uninstall** above.
- If using multiple packages, one can activate an environment with **source activate <env name>** and deactivate an environment with **source deactivate <env name>**.
- More commands can be found [here](#).

## Example

It's important to test whether or not the above steps actually worked. Using examples copied from the [Building Blocks](#) section will work. The peak finding algorithm specifically is a good example to use because it calls **det.calib()** and uses **ImgAlgos**. Below is the code copied from this example.

```
from ImgAlgos.PyAlgos import PyAlgos
from psana import *

ds = DataSource('exp=xpptut15:run=54:smd')
det = Detector('cspad')

alg = PyAlgos()
alg.set_peak_selection_pars(npix_min=2, npix_max=50,
                           amax_thr=10, atot_thr=20, son_min=5)

hdr = '\nSeg  Row  Col  Npix  Amptot'
fmt = '%3d %4d %4d  %4d  %8.1f'

for nevent, evt in enumerate(ds.events()):
    if nevent >= 2:
        break

    nda = det.calib(evt)
    if nda is None:
        continue

    peaks = alg.peak_finder_v1(nda, thr_low=5, thr_high=21, radius=5, dr=0.05)

    print hdr
    for peak in peaks:
        seg, row, col, npix, amax, atot = peak[0:6]
        print fmt % (seg, row, col, npix, atot)
```

The experiment name, run number and detector name should all be replaced with whatever is being used. If this can run successfully, then psana conda has been installed completely and correctly.