A Quick Guide to Archiving Your Data - Sheri Sanders (ss93@iu.edu)

There are two major ways in which to archive your data, either for individual files or large tarballs of files (recommended). Both will be covered here.

1) Individual files:

For individual files, you will want to use the hsi system. You can load hsi using the prebuild module:

```
module load hpss
hsi      #this will bring up the hpss interface for the SDA
```

The commands for this module are a bit different than standard UNIX commands. The major ones you will need are listed below:

```
ls -lB    #this will list the files in your home folder on the SDA, with human readable sizes
put file  #copy file to SDA
get file  #copy file off of SDA
rm file   #remove it from the SDA
exit      #to exit back to where you were on the non-archival file system
```

Example: So for instance, if you wished to archive a file called Data.tar.gz:

```
cd /location/of/Data.tar.gz
ls -lh Data.tar.gz          #note size of file
module load hpss
hsi
ls -lB                      #see what the status quo is before you dump files
put Data.tar.gz
ls –lB                      #see that the file is now on the archive and that the sizes match
exit                        #back to the normal UNIX prompt
rm Data.tar.gz              #remove the now archived file from the dc2
```

2) Groups of files:

While in the above example the file is already tarred/zipped, it is actually computationally faster to combine the compression and archiving steps. This is accomplished with “htar”, which will tar, compress, and archive your files all at once. In addition to the archived file, htar creates two additional files:

- Index file (name.tar.idx): Usually stored in the same directory where the archive itself resides. This is a binary file that is only listed when the -v flag is used (i.e.: -cv in creation of an archive). The file allows the computer to efficiently store location of the file, and is required for every htar action (as htar needs a link to the file referenced).

- Consistency file (/tmp/HTAR_CF_CHK_####): Stored in /usr/tmp/ or /var/tmp/uname. This is also a binary file, but used to check the consistency between the archive file and the index file.

You can leverage these files to determine if your file has transferred properly through the use of two different commands:

```
a) htar -cv -Hverify=paranoid -f <tar name to be created> <files to include>   #at creation
```

When creating archives, this command will force all available checks for consistency after creation of an archive and will report name, size, and other information of the file. This size information can be crosschecked with the files size of the original.

```
b) htar -kv -f <existing tarfile> #check existing file
```

Runs the same checks as above, only on an existing file.

More Detail: According to https://computing.llnl.gov/LCdocs/htar/:
Verify (k or -Hverify) opens a connection to storage, verifies the index file for the archive that you specify with **-f**, then uses the index file to verify every entry in (member of) the archive file itself. If you combine **-K** with **-v**, HTAR lists the name of each file that it finds in the specified archive in alphabetical order, one per line, along with the size of each in bytes and in blocks (excluding the consistency file), then gives a total file count.

So in short, htar allows you to can either check files as you go (recommended) or check it afterwards (useful if you want to check before you delete from dc2).

A couple tips:
This takes FOREVER – what can I do to make this easier?
While we cannot make the process faster (other than using htar over hsi), you can make it easier to move files without babysitting them. To do this, you can use screen.

To enter screen:

```
screen          #This will start a screen session
<whatever commands you want to run – namely htar>
ctrl-A, d       #This will “detach” your screen – it will continue to run without you logged in!
```

You are now free to log out and let htar do its thing! If you want to check screens or reattach:

```
screen –r       #This will “reattached” your screen, or list the screens if you have multiple
screen –r <number>    #reattach one a multiple screens you have
exit            #This will end a screen if you are actively using it
```

Big Red Users – we can make this a bit faster!
If you have a Big Red allocation, you have access to the data transfer nodes (DTNs or dataxfer). You can log onto these as follows:

```
ssh dataxfer.bigred2.uits.iu.edu   #enter name and password when prompted
```

From there, you can move to your home, scratch, or project directories on Karst or Mason. For example:

```
cd /N/u/username/Karst
cd /N/u/username/Mason
cd /N/dc2/scratch/username
cd /N/dc2/project/username
```

Files can then be transferred to SDA using the commands described above, using the specially set up and designated data transfer nodes to make the process faster.

What do I archive?
There is no need to archive every single file generated. Typically files that you may want to archive:

- Raw data – tar by project and archive, you can always reorganize or subset later if need be.
- Output from analyses that took a long time to do
- Output from resources you may not have access to in the future (licensed software)
- Output that depends on random number generation that would not be exactly replicated

Files that are less necessary to archive:

- Files that are easily regenerated – if you can regenerate intermediate files in ~1-2 days, it may be easier to just to that than archive and find it at a later time

READMEs:
A description of analyses/commands/versions of software and redo the analyses is much more convenient to have rather than just file names/output. You can always rerun analyses if you have the commands!

It is highly recommended that you annotate what the files are, what the names mean, and who generated the data, etc. in a README with your archive to include in the tarball.