IRIS/PASSCAL INSTRUMENT CENTER

GENERATING SEED FROM Q330 DATA USING ANTELOPE FOR STAND-ALONE STATIONS

PASSCAL Platforms: Mac OS10 & Linux
Version 2009.195

Data Group

Questions/comments:
data_group@passcal.nmt.edu

Table of contents

APPENDICES

GENERATING SEED FROM Q330 DATA USING ANTELOPE (V2009-189)

1 Introduction

This detailed document serves to guide the data archiver through the process of data archiving, utilizing Linux or Mac OSX operating systems. This guide assumes the user has basic Linux/UNIX skills, which are essential for completing the archiving task. We begin this process with the data on a field or local computer and end with the submission of these data to the IRIS/PASSCAL Instrument Center (PIC). The submitted data undergo fundamental quality assurance checks prior to PIC submitting the data for archiving at the IRIS Data Management Center (DMC). Individuals utilizing the Solaris operating system will find this guide helpful though specific details, such as software installation, for example, may differ. The archiving of your data fulfills the principle investigator responsibilities defined in the PASSCAL Data Delivery Policy (Appendix A).

You will use tools developed by PASSCAL and Boulder Real Time Technologies (BRTT: Antelope) to create a valid dataless SEED volume (dataless) and mini-SEED (mseed) station-channel-day files for archiving purposes. Examples of command line usage, short scripts, and definitions of Antelope parameter files (pf) to generate a dataless and manipulate mseed may be found throughout this guide and its appendices.

Please take a moment to thoroughly review this guide before you start.

If you have any questions please contact: data_group@passcal.nmt.edu.

The steps described below for data processing and archiving (PASSCAL tools and ANTELOPE) work on most platforms. Please refer to Appendix B for specifics on platforms and/or limitations for the current version of Antelope.

Notice that:

General scripts and commands are in **bold**.
Command-line usage is highlighted yellow
GUI options or menus are highlighted turquoise
Standard output is *italicized.*
URLs and email addresses are blue.
Important notes are **brown**.

2 List of Materials and Steps

Prior to starting this submittal process you should contact the data_group@passcal.nmt.edu and acquire, complete, and/or review:

1) network code request form: http://www.iris.edu/scripts/getcode.html,

2) mobilization form: http://www.iris.edu/stations/mob.htm,

3) demobilization form: http://www.iris.edu/stations/demob.htm (to be completed by the end of the experiment)

4) PASSCAL Data Delivery Policy is found in Appendix A, or online at: http://www.passcal.nmt.edu/information/Policies/data.delivery.html,

5) ANTELOPE – Notes on the current release are in Appendix B. Also see the guide for updating and installing new versions in Appendix C.

6) Install the latest PASSCAL software package for your platform, which may be found at: http://www.passcal.nmt.edu/software/software.html.

   Note: The forms in step 1, 2 & 3 alert the DMC of your temporary network and setup the infrastructure needed for the DMC to accept your data.

PASSCAL field computers loaned to the PI are shipped with PASSCAL software, and the most current version of Antelope pre-installed. However, you may need to update the version of Antelope on the computer if it has been in field for more that one year. Additionally, BRTT releases patches throughout the year and it is recommended that you patch your version of Antelope by running **antelope_update** from the command line. For more information about Antelope visit http://www.brtt.com/ and/or read the man pages.

New versions of Antelope are usually released spring or summer each year. You should check for the most recent version at http://www.brtt.com/. If you have an old version of Antelope please fill out the proper form under http://www.iris.edu/manuals/antelope_irismember.htm. If your institution is not an IRIS member and you will process data from a PASSCAL experiment, please contact data_group@passcal.nmt.edu and we will process your license request.

NOTE: the PASSCAL Instrument Center will provide Antelope support for all PASSCAL experiments, as required by an agreement between IRIS and BRTT. Please direct all Antelope questions to data_group@passcal.nmt.edu .

Some of the software you will use originates at PASSCAL. Please find the RELEASE NOTES here: ftp://ftp.passcal.nmt.edu/passcal/software/RELEASENOTES. This page describes our program updates. If you'd like to be on our mailing list for the next release, please send a note to passcal@passcal.nmt.edu. Refer to Appendix D for a list of software that may be useful for data processing.

3 Directions

3.1 Steps in brief

Data Reduction and Timing Quality control

a. Create an organized directory structure for your data
b. Split the Multiplexed Files into sta.net.loc.chan mseed Files
c. Verify Data Quality Using Q330 log Files
d. Check and Fix the headers using fixhdr
e. Change endianess and Flag or Shift Timing
f. Modify default fileds on your traces to MSEED format

Populating the Antelope Database

a. Create a dbbuild batch file
b. Build the Antelope Database
c. View your database
d. Create mseed day volumes and add them to your database
e. Assign calibration values from calibration table to wfdisc
f. Verify the integrity of your database
g. Create the dataless SEED
h. Verify the dataless

Send Data to IRIS/PASSCAL

3.2 Steps in detail

3.2.1 Data Reduction and Timing Quality control

Once Antelope is installed you will begin the process of preparing your data and creating an Antelope database for producing a dataless seed volume. Your multiplexed waveform files need to be split into individual station-channel-day files, (A station-channel-day file is a file containing a day of data for a given channel from one station.)

a. Create an organized directory structure for your data

You may generate your own directory structure and organize it as you see fit. The following is PASSCAL's suggested structure. Let's call the directory where you have all the data "my_experiment". Create the following directories under the my_experiment directory:

raw_data
mseed_dayv_s1

The raw_data directory will be the location of the multiplexed files offloaded from the baler (i.e. the *.xxx q330 files). The mseed_dayv_s1 directory is where the data, in the station-channel-day de-multiplexed form, will eventually reside later in the processing procedure.

Each service of data should be processed in its own directory for ease of record keeping (i.e. new data versus data already archived), however it is recommended to have only one database to represent your entire experiment (more about the database may be found later in this doc).

b. Split the Multiplexed Files into station-channel-day mseed Files

Move your multiplexed baler data, the .xxx files, in the directory raw_data (where my_path is the directory where you have downloaded the raw data files.

<my_cpu:my_experiment> mv my_path/*.xxx raw_data/

Use the command **miniseed2days** to split the multiplexed baler data files into station-channel-day files. The following command line example will write mseed files, utilizing a specific naming convention required for archiving purposes, into station-named subdirectories beneath the mseed_dayv_s1 directory, provide verbose standard output, and write all standard output and standard error output to a file. The output file miniseed2days.out should be reviewed upon completion to ensure no problems occurred during the process.

<my_cpu:my_experiment>**miniseed2days -v –w "mseed_dayv_s1/%{sta}/%{sta}.%{net}.%{loc}.%{chan}.%Y.%j"** raw_data/*.xxx >& miniseed2days.out

Next we review some example output in which miniseed2days did what it was supposed to do, but the output is unexpected. For other errors one may experience when running **miniseed2days** please refer to Appendix I, Table 1.

> **NOTE:** if the subdirectories and/or files written by miniseed2days are not named what you expected, then you will need to correct the mseed headers and appropriately rename the files to reflect the corrected fields.

For example, let's say before you went to the field you unintentionally did not program digitizer 0361 with the desired station name, STN05. Therefore the station code written by the digitizer to the mseed headers is the digitizer serial number, 0361. Since **miniseed2days** utilizes the mseed header fields to name the station-channel-day files and subdirectories, either the files and/or the subdirectories may appear to be incorrectly named. Let's review the output of a few **ls** commands:

<my_cpu:my_experiment> **ls** mseed_dayv_s1/*

0361/ *STN01/*
STN02/ *STN03/*
STN04/

Notice the 0361 directory. This isn't the station name we expected, we are missing the expected STN05 subdirectory, and all the other directories in this example have expected station names.

Another **ls**, this time of the 0361 directory contents, also infers the datalogger serial number is assigned as the station code in the mseed headers. Below are the command line and a subset of the **ls** output.

<my_cpu:my_experiment> **ls -d** mseed_dayv_s1/0361/*

mseed_dayv_s1/0361/0361.XV.01.HHZ.2005.305 *mseed_dayv_s1/0361/0361.XV.02.LHZ.2005.305*

mseed_dayv_s1/0361/0361.XV.01.HHZ.2005.306 mseed_dayv_s1/0361/0361.XV.02.LHZ.2005.306

The mseed headers, which define the data, must be corrected before submission to the DMC. The files and subdirectory must also be renamed to avoid confusion later in the archiving process. While the file names and subdirectory may be renamed with some Linux/UNIX commands and shell scripts, the mseed headers correction requires specific software, such as **fixhdr**. We introduce **fixhdr** later in this document after review of the LOG files generated by each station. Please see 3.2.1.d, Modify headers using Fixhdr (below), and Appendix E - Fixhdr Help, for more information.

One other complication with the incorrectly assigned mseed headers is your original raw data is flawed as well, so it is recommended to keep a backup copy of the corrected station-channel-day files along with the original raw data.

c. Verify Data Quality Using Q330 log Files

The Q330 Baler creates State Of Health (SOH) files, which allow you to ascertain the health of the station. These files may be viewed with **pql**, **pqlx**, and **qlog.** Appendix F, Table 1 shows a description of the SOH channels for a Q330; please refer to it for more detail about SOH channels, specific checks on data quality of the waveforms, and other uses of the software such as:

- timing quality,
- power problems,
- system reboots, and
- generation of mean station locations and elevations.

d. Modify headers using fixhdr

The PASSCAL software, **fixhdr,** ("fix-header") allows users to make changes to mseed fixed header values, change the endianess of the mseed headers, and apply bulk-timing shifts. It also has a batch mode (-b option) that can be run with template files created either by **fixhdr** or from scratch. Typing **fixhdr** on the command line launches the program.

- **Change Endianess and Flag/Shift Timing**

fixhdr also provides a means for you to modify the endianess (byte-order) of your files from little to big (if required) and setting flags for questionable timing, or to apply time correction when needed. To launch **fixhdr** with a GUI (graphical user interface) you need to type on the command line:

<my_cpu:my_experiment> **fixhdr**

Help is available within the program and may be viewed by choosing the help button in the GUI or by running **fixhdr** with the "–h" option. Also see Appendix E - Fixhdr Help, for more detailed information.

- **Modify default fields on your traces to MSEED format**

Header fields will need to be modified following the SEED format (see Appendix G). Fields that you will be able to modify using **fixhdr** are: station name, channel, location code (optional, only if needed), and network code. Please refer to Appendix G for suggested channel names for PASSCAL sensors. See also the Standard for the Exchange of Earthquake Data, Reference Manual, SEED Format Version 2.4 (http://www.iris.edu/manuals/) for complete details on the SEED format.

To build your batch file please refer to the fixhdr help tab**.** Below is an example of a fixhdr batch file built by saving the template with **fixhdr:**

```
# Header Changes
hdrlist{
# sta:chan:loc:net:sps          new sta:chan:loc:net
0965D:1C1::XX:100.0             PA01:EHZ::YW
0965D:1C2::XX:100.0             PA01:EHN::YW
0965D:1C3::XX:100.0             PA01:EHE::YW
09511:1C1::XX:100.0             PA02:EHZ::YW
09511:1C2::XX:100.0             PA02:EHN::YW
09511:1C3::XX:100.0             PA02:EHE::YW
0969D:1C1::XX:100.0             PA03:EHZ::YW
```

Note that in this example we are not defining the location code. If you decide to use location code (let's say you use 00) it should look something like:

```
# Header Changes
hdrlist{
# sta:chan:loc:net:sps
   0965D:1C1:00:XX:100.0       PA01:EHZ:00:YW
   }
```

3.2.2 Populating the Antelope Database

a. Create a dbbuild batch file

The next step is to create an Antelope database that defines your network and station configurations. You will use the tool **dbbuild** in batch mode (**dbbuild –b**) to construct a CSS3.0 database. The batch file is an ascii file with specific keywords and details used to build the database without the use of the GUI. It is an effective way to keep a history of your experiment and also allows you to reproduce most of your database from scratch, if necessary. Use the following template as an example and edit accordingly the fields in green. A batch file may have comments (denoted with #). The description for each field in the batch file (and how **dbbuild** works) may be found in the man pages for dbbuild_batch and **dbbuild**.

If you have questions about **dbbuild** or the batch file, please refer to Appendices H & I (table 2). The

fields in green are the important details regarding each item.

NOTE: the example does not use include location codes. See the dbbuild man page for their usage.

```
#comment: This is a batch file.
net PI Pier database at PASSCAL

sta NP00 -77.72237 162.27354 0.042 Socorro, NM
time 02/06/2004 02:50:53
datalogger  rt130 0984
sensor l28 0 0001
axis Z 0 180 - 1 1
axis N 0 90 - 2 1
axis E 90 90 - 3 1
samplerate 200sps
channel Z EPZ
channel N EPN
channel E EPE
samplerate 1sps
channel Z LPZ
channel N LPN
channel E LPE
add

sta NP02 -77.72591 162.26907 0.079 Socorro, NM
time 02/29/2004 02:57:57
datalogger  rt130 0988
sensor l28 0 0015
axis Z 0 180 - 1 1
axis N 0 90 - 2 1
axis E 90 90 - 3 1
samplerate 200sps
channel Z EPZ
channel N EPN
channel E EPE
samplerate 1sps
channel Z LPZ
channel N LPN
channel E LPE
add

close NP00 12/30/2007 23:59:59
close NP01 12/30/2007 23:59:59
```

We discourage the use of locations codes and suggest they be explicitly defined only when necessary to avoid ambiguity (such as when operating a dense network (stations within 1 km) or when recording multiple streams at sample rates sharing a common band code (first letter) within the channel code.

Please read Appendices G & H for more details on how to identify which channel convention you should use and what every field means in the batch file and how to build it. Refer to Appendix G for suggested channel names.

Your batch file is the history of all the changes, editions, removals, etc. done to the stations in your network, so it MUST include all of details covering from the very first sample rate on any channel to the day the station is closed. We suggest a slightly earlier time for star time (second line after station in the batch file) to assure all the traces are included on the meta-data. This will prevent from further errors and problems during archiving.

Please read this man page carefully and refer to the appendices for detailed information regarding building your database.

b. Create your database

Now that you have a batch file you may run **dbbuild** to create your Antelope database.

<my_cpu:my_experiment> **dbbuild -b my_db your-batch-file** >& dbbuild.out

IMPORTANT NOTES:
The configuration for each station in your batch file must agree with the mseed headers.
The batch file filename should not end with a ".pf" suffix.
Before running dbbuild, please make sure that your batch file is absolutely correct by checking station names, location codes (if used), sensor orientation, start times, close statement, etc.
It is best to start your stations a few seconds or minutes early, rather than milli-seconds late.

Below is a subset of output from **dbbuild -b** (in the above example written to dbbuild.out).

loading batch_file_bf
Added 20 records to calibration
Added 2 records to instrument
Added 1 record to network
Added 20 records to sensor
Added 1 records to site
Added 20 records to sitechan
Added 38 records to stage

By running **dbbuild,** a series of tables and "response" directory are created. These tables and directories are the constituents of the database.

<my_cpu:my_experiment> ls

my_db.instrument	*my_db.site*	*my_db.schanloc*
my_db.sensor	*my_db.network*	*my_db.snetsta*
my_db.stage	*my_db.sitechan*	*response/*
my_db.lastid	*my_db.calibration*	

You may find more information about **dbbuild** in Appendix H, and troubleshooting pointers in Appendix I, table 2**.**

c. View your database

Using **dbe** (a viewing and editting GUI) you may visualize contents of your database. All of the details provided in the batch file are in the various tables of the database.

<my cpu> **dbe** my_db

For detailed description on how to use **dbe** please see the manual page for **dbe ("man dbe")**.

d. Create mseed day volumes and add them to your database

Now you have a database that describes your network, however you have not associated any waveforms with the meta-data.

When adding the data to the database you can use two tools depending on the output files when running **miniseed2days** the first time (page7), when demultiplexing the files.

- If you had to modify the header of the files using fixhdr you will need to run **miniseed2days** on those files so the new files have the proper headers and file name convention as you did on page 7.
- If the header of your files were correct already (meaning the Q330 was programmed with

the proper net code , station name, channel and location code, therefore the miniseed day volumes have the right filename convention and are ready to be added to the database) then you can run **miniseed2db** as specified below:

To add your waveforms details to your database, use the command **miniseed2db.** This will create the miniseed day volumes (from your header-corrected mseed files in the ref_mseed directory) and create an extra table for your database with the information regarding the waveforms called *my_db.wfdisc*:

<my_cpu:my_experiment> **miniseed2db -v** mseed_dayv_s1/* my_db >& miniseed2db.out

Note: the mseed headers read by miniseed2db are the source of information used to populate the database's waveform table. You must ensure the mseed headers in the station-channel-day files produced by miniseed2days are. If the database (and its batch file) does not describe all of the data then errors will result when we check the consistency of the database.

e. Assign calibration values from calibration table to the wfdisc table

To ensure that the calibration values are incorporate into the newly created wfdisc table please run **dbfix_calib**.

<my_cpu:my_experiment> **dbfix_calib** my_db

f. Verify the Integrity of Your Database

Before you create a dataless you will want to ensure your meta-data completely describe the waveform data and your database is free of errors. **Read** the man page on **dbversdwf** and **dbverify** for details regarding tests you may run on your database. Examples of suggested tests are:

<my_cpu:my_experiment> **dbversdwf -tu** my_db
0 bad files

A more detailed description of different warning/errors you may encounter when running verifications on your database can be found in Appendix I**,** Table 3.

<my_cpu:my_experiment> **dbverify –tj my_db** >& dbverify.out

g. Create the dataless SEED volume

The dataless SEED volume, often referred to as a “dataless”, contains the meta-data describing the station and instrumentation of your experiment. To generate the dataless SEED volume, run **mk_dataless_seed**, which builds the dataless from the contents of your experiment’s database. You will submit this file along with the waveforms to PASSCAL.

<my_cpu:my_experiment> **mk_dataless_seed –v –o** PI.04.my_db.20042082000.dataless my_db

Using existing my_db.snetsta table
Finished building dataless wfdisc
PI.04.my_db.20042082000.dataless truncated to 24576 bytes

Using the option -o you may name the dataless using the required format. Please use the following naming convention:

NN.YY.dbname.YYYYJJJHHMM.dataless

Where:

NN is your network code
YY is the year of your data
YYYYJJJHHMM is the approximate current time – year-julian-day-hour-minute

To convert from calday to Julian day, for example March 1, 2007:

<my_cpu> **julday** 03 01 2007

Calendar Date 03 01 2007

To find the current julday:

<my_cpu> **julday**

Calendar Date 03 01 2007

To convert from Julian day to calendar day, for example day 150 of 2006:

<my_cpu> **calday** 150 2006

Calendar Date 05 30 2006

h. Verify the dataless

Now you may check the structure of the dataless with **seed2db.**

<my_cpu:my_experiment> **seed2db –v** my_db_dataless_seed

Please refer to Appendix I, Table 4, for possible cases you may run into when running **seed2db**.

Note: the dataless must describe the entire data set, including all service runs of data. The agreement, or lack thereof, between the dbbuild batch file, the resulting database, the dataless, and waveforms will be reflected in the availability of the data at the DMC.

3.3 Send Data to IRIS/PASSCAL - DoFTP

When ready to submit the data to PASSCAL, please contact us by sending an email with your experiment name and network code to data_group@passcal.nmt.edu. For example: XO –Terra data 2004-2005.

There are two options data submission via FTP: using the command line or a GUI. We recommend use of **GUI_DoFTP** to submit data to the PIC (current version 2008.038 or later version). See Appendix J

for more details on **gui_DOFTP**, which is a python-based package available from PASSCAL as part of our software release: http://www.passcal.nmt.edu/software/software.html.

<my_cpu:my_experiment> do_ftp_path/**gui_DoFTP**

(Where do_ftp_path is where you have installed **DoFTP**.)

DoFTP will:

- Descend the specified directory path, identify, and pack ALL mseed files found
- Create .tar and .md5 (similar to check sums) files of the data
- Send the dataless and its .md5 file
- Build a report (list) of all data files sent and its md5
- Start an FTP session to PIC and send the data

Note:

- Be as specific as possible when specifying the path to the data, so unintended files are not packed
- The software requires at least as freer disk space as the size of the data set to be sent. That is, if you have 100 GB of data to send, **DoFTP** will need at least another 100 GB of free space to build the tar files.

To use in the command line option, type **con_DoFTP** .

<my_cpu:my_experiment> do_ftp_path/**con_DoFTP**
(do_ftp_path is where you have installed **DoFTP**.)

-#	print version of this program
-a	force ACTIVE FTP mode (default is PASSIVE mode)
-f	ftp the tarred data or resume ftp from the last broken pt.
-r	gives an integer from 1 to 366 (default is today's julday)
-t	set FTP timeout with a positive integer (default: no timeout)
-help	print help information

A typical command line usage may look like this:

./con_DoFTP -a -f -r 366 -t 15 /Users/kxu/FA_tremor

4. Adding more data during future services

A typical question from a data archiver: "I have more data from the last service run. Is there a way to add the new data to the existing database? "

Yes - just need to be consistent, do the initial quality control on your data and follow the same steps previously described. If during service changes have been made to the initial configuration of your stations, make sure those changes are also included in the batch file and, therefore, in your database.

Here are some examples of what to do in each case:

To add new data to the existing database you will follow the same steps as before with some slight variations. Data reduction and timing quality control will remain the process as before for previous services. Make sure to be consistent with the use of location codes, network and channel assignment, etc when fixing headers. Sending data to PASSCAL will be the same as well. Below you may find some points to consider while populating the database during later services.

1. Data Reduction and timing quality control – same as before
2. Populating the Antelope Database for further services

    a. Update the Batch File (if needed)

At this point you have already a batch file. You may need to update or modify it if any of the following situations apply:

i. NEW STATIONS - you need to add each new station with its proper configuration to the batch file and re-run dbbuild in the same directory where you create the database the first time.

ii. REMOVED STATIONS – if there is any existing data for this station you simply add a close statement (e.g. if the station NP00 was removed April 10 2006, use "close NP00 04/10/2006 10:15:59"). If data never was recorded for this station no need to add it to the batch file.

iii. CHANGED sensor, digitizer, sample rate, gain or fix orientation – in this case you will add an extra block describing the same stations with the modified fields below the first description. The start time of the second configuration will be the end time for the initial configuration.

IF THERE ARE NO MODIFICATIONS (different sensor type and/or serial number, digitizer, gain, sample rate, orientations): there is NO need to re-run dbbuild since your stations are already accurately described in your database.

    b. Building the Antelope Database

If none of the above 3 points come up, there is no need to re-run dbbuild since your stations are already described on your database or build a new dataless. If one of the three points were required then you will need to update your database with dbbuild as shown below:

<my_cpu:my_experiment> **dbbuild -b my_db** your_updated_batch_file

    c. View your database – same as before

    d. Adding Your Waveforms to the Database

Once you have the new data ready to add to the database (QC complete, timing issues evaluated, headers fixed, etc), you may add the waveform information to the database using the same command as before, however be sure to specify the

location of the new service run's station-day-volumes (let's assume you have it under service2), then you will run:

<my_cpu:my_experiment> **miniseed2db -v** service2/* my_db

e. Verify the Integrity of your Database – same as before
f. Create the dataless SEED – same as before
g. Verify your dataless file and Rename to Conform With Convention -same as before

5. Updating the meta-data without processing new data

All changes change/addition/removal in your network configuration must be described on your dataless. This dataless must be submitted to the PIC for review and archiving at the DMC so the appropriate changes are visible for data and meta-data requests. Meta-data/dataless changes may occur at any time including between service runs and after an experiment is complete.

There are a couple of way to update your database and dataless. One clean way to add to or change a dataless is to simply create a temporary database in a separate directory and generating a dataless within it. The steps you should follow are:

a. Create a temporal directory to work on your new dataless (e.g. my_new_dataless)

<my_cpu> **mkdir** my_newdataless

b. Copy your existing batch file to the temporary directory.

<my_cpu> **cp** my_batch_file my_newdataless/modified_batch_file

c. Edit it.

d. Create a new database by using **dbbuild.**
e. Check the database with **dbverify.**
f. Fix any errors, if you have any questions please e-mail data_group@passcal.nmt.edu

<my_cpu> **dbbuild –b** modified_db modified_batch_file
<my_cpu> **dbverify –tj** modified_db

g. Create a new dataless, which will contain all the information that needs to be incorporated in the database you have with all your waveforms.

<my_cpu> **mk_dataless_seed –v –o** PI.04.my_db.20072201700.dataless modified_db

h. Verify your dataless:

<my_cpu> **seed2db –v** PI.04.my_db.20072201700.dataless

i. Contact the data_group@passcal.nmt.edu regarding how to submit the updated dataless.

6. Verifying archived data

Usually once the data makes it to our system, it will run through verification software. If the data and dataless pass all the checks in the Quality Control System (QCS), the data are prepared for submission, as station-day volumes, to the DMC. This process may take between one to two weeks depending on how data volume flowing through the PIC and to the DMC. Once the data are sent to the DMC, the waveforms and meta-data are read and loaded into an ORACLE database and the waveforms are archived. Once we confirm the data has been archived we will send you an e-mail with a summary of the data archived for your experiment. Please take a moment to ensure this summary agrees with your records of data you expect to be archived.

7. Other antelope tools that may be helpful

PASSCAL Tools

- **StaDayVols.py -** Determines what files in indir can create complete Station Day Volumes. The channels needed for completeness are taken from the css db "db" which must have sitechan, snetsta, and chanloc tables. If outdir is given; complete station day volumes will be created in directory outdir. If -i is used incomplete station day volumes will be created as well.

usage: StaDayVols.py [options]
options:
-h, --help show this help message and exit
-v Verbose; will also describe complete Vols
-d DB CSS database to use for station description
-f INDIR Dir to search for MSEED files
-o OUTDIR Dir to place created vols, will be created if doesn't exist
-i Used with -o, will also create incomplete vols from files that exist

- ***ckMseed*** *– with the –h option provides its use, it helps to "map" the mseed traces on a given directory providing information on the headers, endiannes start and end times. This tool is very useful when trying to visualize the headers of your traces and possible issues in them and to adjust dataless accordingly.*

Usage : ckMseed -h

ckMseed [-d DataDirs] [-s] [-v]
-d DataDirs - colon separated list of data directories [default: cwd]
-h Usage
-s Simple mode. Check and report endianess of first fixed header only
-v Check and report endianess of first fixed header and determine
start times for first and last block of mseed file
-V Read all header blockettes
NOTE: -s, -v, & -V are mutually exclusive.-V supercedes -v supercedes -s

Please request the man page for these tools for more detail.

- **dbplotcov** reads the wfdisc table from the specified database, determines the periods of time for which waveform segments exist for each station-channel and prints and plots this information. A PostScript version of the coverage plot, named dbplotcov.ps, is created in the current directory. This tool has several caveats but for small databases it works to give a visual display of the coverage for each station, all channels on your db.

  dbplotcov usage: dbplotcov dbname stachan tstart tend [-h [ntrigs]] [-wftar]

- **db2sync** is designed to create a synchronization file of a datascope (antelope) database in the format utilized by the IRIS DMC.

usage: db2sync [-s start_time] [-e end_time] [-S subset_expression] [-p pf_file] [-a] [-l] [-d] [-h] [-o] [-w] [-v] dbin fileout

- **BRTTPLOT** allows one to view axes, grid, ptext, polyline, polypoint, map - BRTT tk canvas item extensions. These are all special tk canvas item extensions available through the Brttplot package in the Antelope tcl/tk extensions. All of these canvas item widgets act as normal tk canvas items, including such functionality as the ability to display these in scrolled canvases and the ability to generate PostScript output, and they should be thought of as extensions to the various items that are described in canvas(n).

- **dbsnapshot** collects some information about and some records from a database db into a single tar file. This can be helpful for providing some information when trying to resolve a database problem.

- **q330gpslocate** program parses a series of log files from a Quanterra Q330 data logger. From these files it extracts lines in the log file containing latitude, longitude, and elevation to produce a mean of each parameter.

8. Using DMC tools to View/Requests your archive data

a) IRIS/DMC Meta-data Aggregator

Using the meta-data aggregator from DMC (http://www.iris.edu/mda/) you can view the complete list of assigned FDSN network codes, including all the networks that submit data to the DMC. Therefore, this is a good portal that summarizes data collected from PASSCAL experiments since 1986. Parametric information that is extracted from all submitted dataless SEED volumes for each network (location, time span, type of data –Real-time (R) -or Archive A)-, station names, number of stations, channels, instrument response plots, etc) can be found for each network available, as well as a link to the DMC's Google map service (http://www.iris.edu/gmap) that the locations for each network that has submitted meta-data to the DMC. Using the network code and the years of your experiment you can see

the information stored about your network.

b) VIRTUAL NETWORKS

Currently the DMC has available 18 virtual nets (http://www.iris.edu/mda#vnetlist) including 2 from PASSCAL (_PASSCAL & *PAS-OPEN),* all EarthScope stations *(*US_ALL), USArray Flexible Array (_US-FA), and USArray Transportable Array (_US-TA), among others.

_PASSCAL Virtual Net – contains over three thousand stations with analog and real time data since 1990 to the present. Data from all PASSCAL experiments with archived data and currently deployed and/or submitting data.

_PAS-OPEN Virtual Net – Data from stations that have been made available to the public from PASSCAL experiments (since January 2005) following the data delivery policy, as explained in APPENDIX A.

c) BUD_stuff, Monitor, QUACK and others

BUD is the IRIS DMC's acronym for the online data cache from which we distribute our near-real time miniSEED data holdings prior to formal archiving.

d) VASE

VASE is a Java-based client application designed for viewing and extracting seismic waveforms from the DHI waveform repository via BUD.

e) JWEED

JWEED is a Java update of WEED allowing users to access event and station data through an interactive map.
You can find some interesting links under: http://www.iris.edu/

*IRIS/PASSCAL Documentation- Created by Eliana Arias (*eliana@passcal.nmt.edu*, 2006, 2007).*
Revised Eliana Arias (2007)
Revised Bruce Beaudoin (2006,2007,2008)
Revised Lisa Foley (June, 2008)
Revised Eliana arias (July 10, 2008)
Revised by George Slad, August 27, 2008
Revised by Eliana Arias August 5, 2009
Revised by Lisa Foley July 14, 2009