IRIS/PASSCAL INSTRUMENT CENTER

GENERATING SEED FROM RT130 DATA USING ANTELOPE FOR STAND-ALONE STATIONS

For PASSCAL Platforms: Mac OS10 & Linux
Version 2012-286

This document was written and produced by the
PASSCAL Data Group

Please send your feedback, questions, and comments to:
data_group@passcal.nmt.edu

Table of contents

List of appendix for data & software

Name/Short description

1. ***Data Specific***

- ***Data Delivery Policy:*** *Requirements on Data Delivery for PASSCAL*
- ***SEED Format:*** *Brief description of seed format for archiving of passive source data*
- ***RT130 State of Health Channels (SOH):*** *Description of SOH in a RT130*
- ***Logpeek:*** *Evaluation of Log files using Logpeek (Reftek – rt130)*
- ***Troubleshooting:*** *Common problems when processing data for archiving*

2. *Software Specific*

- ***Antelope:*** *How to update and install Current Version Notes*
- ***How to build a Batch file –PASSCAL'S Antelope guide & examples:*** *Details on how to generate batch file in antelope*
- ***PASSCAL & contributed software guide:*** *Links to main software and brief detail on most used tools for data processing/archiving*
- ***FIXHDR Help:*** *Detail instructions on using fixhdr*
- ***Gui_DoFTP:*** *Detail Instructions on using gui_DoFTP to send data to PASSCAL*
- ***Troubleshooting some basic antelope tools during data processing***
- ***Other useful tools:*** *Common problems software related & other useful tools*

Note: These Documents are available online http://www.passcal.nmt.edu/content/passive-source-seed-archiving-documentation *or by request to the data group data_group@passcal.nmt.edu*

GENERATING SEED FROM RT130 DATA USING ANTELOPE

(V2012-286)

Introduction

This detailed document serves to guide the data archiver through the process of data archiving using a Linux or Mac OSX operating systems. This guide assumes the user has basic Linux and/or 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 submission of the data for archiving to the IRIS Data Management Center (DMC). Individuals using the Solaris operating system will find this guide helpful although 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 (Please refer to additional documentation in our appendixes)

You will use tools developed by PASSCAL and Boulder Real Time Technologies (BRTT) to create a valid dataless SEED volume (a.k.a "dataless") and mini-SEED (mseed) station-channel-day files, which will be archived together. 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.

Notice that:

Headers, 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**.

List of materials and initial steps

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

1) Principal Investigator web page (new) http://www.passcal.nmt.edu/pihomepage
   Create an account and follow your experiment.
2) Mobilization and Network code request form: best done through PI Home Page

3) Demobilization form, also done through PI Home Page
(To be completed by the end of the experiment.)

4) PASSCAL Data Delivery Policy is found online at:
http://www.passcal.nmt.edu/content/general-information/policy/data-delivery-policy

5) ANTELOPE – Notes on the current release are online as well as the guide for updating and installing new versions.

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

7) Submit a bug report or problem to: http://www.passcal.nmt.edu/node/add/externalticket

Note: The forms in step 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 principal investigator (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 data processing for all PASSCAL experiments, as required by an agreement between IRIS and BRTT. Please direct your Antelope questions to**:** data_group@passcal.nmt.edu. Questions regarding further processing such as location of events, etc are beyond the scope of our data archiving support.

Some of the software you will use originates at PASSCAL, please refer to our documentation **web page** http://www.passcal/nmt.edu/content/passive-source-seed-archiving-documentation **for more detailed information.**

Steps in brief

Data reduction and timing quality control

- Back up the raw data from the RT130 compact flash cards
- Create an organized directory structure for your data
- Create a batch file
- Convert raw data into mseed and populate headers, assign endianness, block size, etc.
- View logs and waveforms: Quality Control

Populate the Antelope database

- Build the Antelope database
- View your database
- Create mseed day volumes and add them to your database
- Assign calibration values from calibration table to wfdisc
- Verify the integrity of your database
- Create the dataless SEED volume

Send data to IRIS/PASSCAL

Steps in detail

Data reduction and timing quality control

Back up the raw data from the RT130 compact flash cards

We encourage PASSCAL and USArray/Flex Array users to backup the raw images, and simultaneously generate zip files of the raw images where you can extract the mseed, ref, or log files for quality control and further processing. Instructions on this step vary depending on your platform. Please find below a suggested guide and comments about how to back up your data when working from one of the PASSCAL field machines (LINUX, MAC OS) or a Solaris machine.

The software programs you will use are called **neo and gents.** These are PASSCAL scripts and part of the PASSCAL software release.

> **Neo**: is a method of extracting the data from the flash cards and generates ZIP or TAR files.
> **gents** : produces a time span from the tar of zip file after all the data have been offloaded.

If you cannot find the programs listed above or have any trouble please let us know by writing to passcal@passcal.nmt.edu.

Create an organized directory structure for your data

You can generate your own directory structure and name it as you see fit. The following is PASSCAL's suggested structure. Let's call "EXPT" the directory where you have all the data. Create

the following directories under EXPT *directory* and organize the files accordingly:

<my_cpu:EXPT > mkdir RAW - where raw datafiles (*.ZIP) generated by **neo** will be stored

<my_cpu:EXPT > mkdir rt2ms– location for mseed files, after extracting them with **rt2ms**

<my_cpu:EXPT > mkdir LOGS – location for *.log, *.err, *.run

<my_cpu:EXPT > mkdir day_volumes – where the day volumes will go after running **miniseed2days and log2miniseed**.

<my_cpu:EXPT> mkdir DB – where batch file and par file will go

Note that a "response" and "nom_response" directories will be automatically created by **dbbuild. Note: Antelope versions before 5.0 will not create the nom_response directory.**

IMPORTANT – Place mseed day volumes of waveforms and log files (after running **miniseed2days** and **log2miniseed**) in a directory called, for example, "day_volumes". The name is optional, call it what you like, but be consistent throughout the process.

You may also want to name the directory for the mseed day_volumes by including the service number you are working on (e.g. day_volumes_S1, day_volumes_S2, etc). This will give you some flexibility and easy access for each service run. If pointed to the same Antelope database, the traces for all services will be part of the same wfdisc table.

Create a batch file

In previous guidelines we suggested this step later on, however since the batch file can be used to generate the parameter file that will convert the raw data into mseed and populate its headers, we have to move this step ahead in the procedure.

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 (next page) as an example and edit accordingly the fields in green. A batch file may have comments denoted by #. The description for each field in the batch file (and how **dbbuild** works) may be found in the man pages **dbbuild_batch**, **dbbuild and dbbuild_examples** or in our appendices.

In the example below the fields in green are inputs that you need to provide; while fields in black are recognized by Antelope Header fields will need to be modified following SEED format (for more details please check our appendices for Passive Source documentation and examples of the different channel naming conventions according to SEED). Please refer to the Standard for the Exchange of Earthquake Data Reference Manual SEED Format Version 2.4 (http://www.iris.edu/manuals/) for complete details on SEED format.

NOTE: In the example below we don't include location codes. If you prefer to use location codes you should have something like:

samplerate 200sps
channel Z EPZ 01
channel N EPN 01
channel E EPE 01

where: 01 is the location code for data stream 1.

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

sta NP00 -77.72237 162.27354 0.042 Socorro, NM, USA
time 02/06/2004 02:50:53
datalogger  rt130 0984
sensor sts2_g1 0 0001
axis Z 0 0 - 1 1
axis N 0 90 - 2 1
axis E 90 90 - 3 1
samplerate 40sps
channel Z  BHZ
channel N BHN
channel E BHE
samplerate 1sps
channel Z LHZ
channel N LHN
channel E LHE
add

sta NP02 -77.72591 162.26907 0.079 Socorro, NM
time 02/29/2004 02:57:57
datalogger  rt130 0988
sensor cmg3t 0 0015
axis Z 0  0 - 1 1
axis N 0 90 - 2 1
axis E 90 90 - 3 1
samplerate 200sps
channel Z HHZ
channel N HHN
channel E HHE
samplerate 1sps
channel Z LHZ
channel N LHN
channel E LHE
add

close NP00 12/31/2007 23:59:59
close NP01 12/31/2007 23:59:59
```

```
#comment: This is a dbbuild batch file example
net network code network name

sta stacode lat long elevation (km) city, state, country of sta
time config start time ← time when you power on or earlier
datalogger code serial number <--code from par files
sensor code edepth serial number ← depth below surface
axis label hang vang [sens [lead [pgain [pstage]]]]
axis2 label hang vang [sens [dlgain [pgain [pstage [lead]]]]
axis3 label hang vang [sens [lead [pgain [pstage]]]]
samplerate code ← appropriate sample rate for your sta
channel axis label
channel axis label
channel axis label
samplerate code ← appropriate sample rate for your sta
channel axis label
channel axis label
channel axis label
add ← adds the current configuration to the database

sta staname lat long elevation (km) city, state, country of sta
time config start time ← time when you power on
datalogger code serial number <--code from par files
sensor code edepth serial number ← depth below surface
axis label hang vang [sens [lead [pgain [pstage]]]]
axis2 label hang vang [sens [dlgain [pgain [pstage [lead]]]]
axis3 label hang vang [sens [lead [pgain [pstage]]]]
samplerate code← appropriate sample rate for your sta
channel axis label
channel axis label
channel axis label
samplerate code ← appropriate sample rate for your sta
channel axis label
channel axis label
channel axis label
add ← adds the current configuration to the database

close sta time (VERY IMP! Closes the sta at this time)
close sta time VERY IMP! Closes the sta at this time)
```

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 of stations within 1 km) or when recording multiple streams at sample rates sharing a common band code (first letter) within the channel code.

It is important to always remember that your batch file is the history of all the changes, editions, removals, etc. done to the stations on your network, so it MUST include all of these changes, covering the times frames from the very first sample rate on any channel, to the day the station is closed.

We suggest a slightly earlier time for the start time (second line after station in the batch file) to assure that all of the traces are included within the meta-data. This will prevent further errors and problems during archiving.

Please read this manual carefully and for more details on specifications of each parameter please refer to our appendix on "How to Build a Batch File and Examples".

Convert raw data into mseed and populate headers

A powerful new tool has been developed by PASSCAL called **rt2ms.** It generates mseed from an rt130 REFTEK raw file into mseed, modifies the headers and specifies the byte order, block size and encoding. Use **rt2ms** with its partner, **batch2par,** to easily specify the parameters found in your batch file.

For help on **rt2ms** please type:

<my_cpu:EXPT > *rt2ms –h*

batch2par: Generates the template for the parameter file ("par_file") to be used when running **rt2ms** to convert multiple ZIP files into mseed with the proper headers and other fields. This tool is OPTIONAL; you can generate and edit the par_file from scratch, or work on a single-file level if preferred (-f option).

NEW TOOL	Feature	Tool previously used or recommended parameter
rt2ms	Generates *ref file	rt130cut (or unchunky *)
	Generates mseed file	ref2mseed (or unchunky*)
	Fixes headers (net:sta:chan:loc)	fixhdr
	Byte order: – always "big" for mseed archiving	
	Block size	Usually 1024 or 4096
	Encoding	STEIM2

***rt130cut** and **ref2mseed** are the scripts executed by **unchunky**, to generate the mseed files.
If used with par_file, **rt2ms** replaces **rt130cut**, **ref2mseed** and **fixhdr.**

To generate mseed day files, please follow these steps:

1. Revise the batch file: Please make sure the configuration defined for each station in your batch file is correct. This information will be used to generate the par_file and create the mseed files.

2. Run batch2par: In the parameter file you can specify multiple fields, an example of the par_file.txt required by **rt2ms** could be:

```
#das;refchan;refstrm;netcode;channel;loccode;encoding
9882; 1; 1; XY; EPZ; 00; STEIM2
9882; 2; 1; XY; EPN; 00; STEIM2
9882; 3; 1; XY; EPZ; 00; STEIM2
```

Adding the mass position channels to your data: now the RT130 is recording the mass positions as part of the data, these data need to be included in your par_file so the headers get populated as well. Please note that since the mass positions are not specified in the batch file (and should not be), batch2par does not add them to the par_file. You must add these yourself to the par_file (recommended) or change the headers separately using **fixhdr**.

You can manually generate the par_file, or you can use batch2par to generate a template by running the following command in the directory where you have the batch file or:

<my_cpu:EXPT > batch2par Batch_expt > par_file_your_expt_name.txt

Where:
Batch_expt is the batch file you created
par_file_your_expt_name.txt is the generated par file from batch2par (named as you so choose)

The output file from ***batch2par*** MUST be edited and corrected, since the current mapping is not one-to-one, from the batch file used in Antelope to the par_file required by **rt2ms**. This tool is a work in progress and we continue to work on improving it in the near future to eliminate the dummy characters batch2par currently adds to the par file.

An example of the resulting par_file after running batch2par using our example batch file looks like:

```
#das; refchan; refstrm; netcode; station; channel; samplerate; gain
9306;  1;    rs100spsrs;    PI;   NP00;   BHZ;   100;   x1
9306;  3;    rs100spsrs;    PI;   NP00;   BHE;   100;   x1
9306;  2;    rs100spsrs;    PI;   NP00;   BHN;   100;   x1
9306;  1;    rs1spsrs;      PI;   NP00;   LHZ;   1;     x1
9306;  3;    rs1spsrs;      PI;   NP00;   LHE;   1;     x1
9306;  2;    rs1spsrs;      PI;   NP00;   LHN;   1;     x1
```

Your final par_file should look something like the one below so you will need to replace the following fields:

I might make this area stand out more
refstrm to 1, 2 or 9 - (1 and 2 usually the main data streams with data, e.g. 40sps and 1sps) and 9 the mass position channels
gain : from x1 to 1 or 32. (Most broadband sensor are set up to gain 1; while l22, l28, GS11V are set to

gain 32)

Adding the mass positions:

Since mass position channels are NOT described in the batch file to use in antelope they are not populated to the par_file needed by rt2ms. The description of these 3 channels needs to be added as described below:

```
9306;  1;     9;     PI;    NP00; VM1;  0.1;    1
9306;  3;     9;     PI;    NP00; VM3;  0.1;    1
9306;  2;     9;     PI;    NP00; VM2;  0.1;    1
```

IMPORTANT (in brown)
The standard sample rate for the mass position channels in rt130's should be 0.1 (this whole sentence in brown)

The par_file to be used by rt2ms must include all the stations (as does the batch file to be used in antelope), so you will need to add the description for mass positions (VM1, VM2, and VM3) for each station.
Thus using our example batch file, the par file, with the description for ONE station with 100sps and 1sps with gain 1 and recording mass position channels should look like:

```
refchan; refstrm; netcode; station; channel; samplerate; gain
9306;  1;     1;     PI;    NP00; HHZ;  100;    1
9306;  3;     1;     PI;    NP00; HHE;  100;    1
9306;  2;     1;     PI;    NP00; HHN;  100;    1
9306;  1;     2;     PI;    NP00; LHZ;  1;      1
9306;  3;     2;     PI;    NP00; LHE;  1;      1
9306;  2;     2;     PI;    NP00; LHN;  1;      1
9306;  1;     9;     PI;    NP00; VM1;  0.1;    1
9306;  3;     9;     PI;    NP00; VM3;  0.1;    1
9306;  2;     9;     PI;    NP00; VM2;  0.1;    1
```

Thus notice that to have ONE station completely described in this case 9 entries are necessary, 3 of which are needed to be input manually. For ONE station with only one sample rate, 6 entries would be needed, three describing the channels, and 3 describing the mass positions.

Other fields like encoding (STEIM2 compression), block size could also be defined on this file.

IMPORTANT**:**
batch2par is a tool that creates the template for the final parameter file to be used by **rt2ms**. Keep in mind the following:

-
- The par_file does not keep track of time frames (start/end time). If an instrument is assigned to multiple stations at different time frames, make sure appropriate mseed header assignments are made for the time period of your data.E-mail the data group if questions
- When using the –F option (file list) **rt2ms** will process ALL the files listed on it, even if they

are NOT described in the par_fileleaving headers as programmed in the RT130.

- There are options to handle one file at the time (-f) , a directory (-D)
- You can also use the command line to assign net codes, channel names, encoding, etc, but this is usually not recommended due to efficiency. A par_files will take care of all the data if properly defined.
- Make sure you add the data for the mass position refstrm 9 or 09
- Rt2ms will identify the data streams (1, 2 and9) and the mapping defined in the par file to convert the data to mseed, generate the log files (when using –L option) and populate the headers.
- Once these points are understood, proceed to run **rt2ms**.

Run **rt2ms**: now that you have the par_file, you can proceed to convert your raw data to mseed format with the proper headers:

<my_cpu:EXPT > rt2ms -F list_S32009_059 –R -Y -L -o TEST -p par_file_labarge_test09173

Where:

-L gives you log files
-Y puts the data in year and day directories, use this if there's a chance of data spanning a year boundary, otherwise omit
-o specifies the name of the directory to put the files into (in our example TEST)
-p specifies the use of the par file with the name of your par file following it
list_S32009_059 is a text file that lists the path to each individual ZIP file (needs to be created by you, the user, before calling rt2ms and can be named according to your choosing)

An example entry of your list_S32009_059 based on our example file structure might be:

EXPT/RAW/2009060.9306.ZIP
etc.

Under the TEST directory you will find:

2009.019.21.29.16.930E.log

R022.01	*R025.02*	*R029.01*	
R019.01	*R022.02*	*R026.01*	*R029.02*
R019.02	*R023.01*	*R026.02*	*R030.01*
R020.01	*R023.02*	*R027.01*	*R030.02*
R020.02	*R024.01*	*R027.02*	*R031.01*
R021.01	*R024.02*	*R028.01*	
R021.02	*R025.01*	*R028.02*	

And under R025.01, for example, are the mseed files:

2009.025.00.29.16.930E.1.1.m
2009.025.06.29.16.930E.1.1.m
2009.025.12.29.16.930E.1.1.m

2009.025.18.29.16.930E.1.1.m
2009.025.00.29.16.930E.1.2.m
2009.025.06.29.16.930E.1.2.m
2009.025.12.29.16.930E.1.2.m
2009.025.18.29.16.930E.1.2.m
2009.025.00.29.16.930E.1.3.m
2009.025.06.29.16.930E.1.3.m
2009.025.12.29.16.930E.1.3.m
2009.025.18.29.16.930E.1.3.m

These files are in mseed format and if everything worked correctly with the par_file and rt2ms the headers should be properly populated with the network code, station name and channel convention, HOWEVER is very important to check if that's the case. To assure all the headers have been properly populated please run fixhdr, build the trace db on the *.m files and make sure all the field in all your traces are as expected. Please e-mail the data_group if you have any questions at (data_group@passcal.nmt.edu)

View logs and waveforms: quality control

This is a good time to do a preliminary quality control on your data and log files. Quality control on mseed data can be extensive. We highly recommend a data evaluation before you continue archivingUse **pql** for waveform QC, and **logpeek** for log file evaluation**.** Both are part of the PASSCAL software release. They may be simply called via the command line by their names. Usage can be found for each program using the -h option.

If timing errors larger than one half of the highest sample rate occur, then it may be best to flag the data as 'timing questionable'. Set the data quality bit in the miniseed header. Use **logpeek** to identify timing issues and **fixhdr** to set flags if the timing is questionable, if there are no timing issues please continue to the next section.

(May want to actually describe how to set these flags, or do so in the documentation for fixhdr (it's not obvious, at least point user to these appendices :))

Basic checks include:

- Timing quality – GPS locks/unlocks, GPS clock lock gaps
- Power problems and system reboots
- Large phase errors, time jumps/jerks, and unexpected gaps
- Station GPS location (average given)
- Voltage drops
- CPU version

Convert Reftek log files into mseed

A complete miniseed day volume contains the traces for each data stream (sample rates) by channel (Z,N,E), the mass position channels, and the state of health channel or LOG file. The LOG file generated by **rt2ms** is an ASCII file that needs to be converted into mseed with the filename

convention as the other traces generated by **rt2ms** this process is done with the Antelope tool called **log2miniseed**, and the steps are described below:

1) Copy and edit the local log2miniseed.pf to define directory structure and file name convention**.**

To complete this step you need to copy the log2miniseed.pf file into your local directory, or edit it in the default pf file located under $ANTELOPE/data/pf/

<my_cpu:EXPT> cp $ANTELOPE/data/pf/log2miniseed.pf .

Make the part below stand out more (I often don't see this and forget about it)

IMPORTANT:

Change the default string*:* **wfname %Y/%j/%{sta}. %{chan}.%Y:%j**
to:
wfname day_volumes_S/%{sta}/%{sta}.%{net}.%{loc}.%{chan}.%Y.%j

The word "**wfname"** is part of the parameter file format so you should keep it. **"day_volumes_S"** is the directory where the log files will go under a station subdirectory. Make sure this directory is the same directory where you will write the mseed day-volumes when running minseed2days –d.

2) Set up the environment

For tsch, type:

<my_cpu:EXPT> setenv PFPATH $ANTELOPE/data/pf:.

For bash, type: <my_cpu:EXPT> PFPATH=$ANTELOPE/data/pf:.

3) Run log2miniseed

For each station (log file) you will need to run **log2miniseed,** or write a simple script to run **log2miniseed** for all your log files. For one log file the command line will be for example:

<my_cpu:EXPT> log2miniseed –n PI –s NP00 run_logs/2005:128:15:09.0965D.log

IMPORTANT: When running **log2miniseed** you are modifying**:**
Network code: -n PI (PI is an example),
Station name: -s NP00 (should correspond to the station name associated with the log file *2005:128:15:09.0965D.log)*
Logs: the directory where you have the log files; will depend on the directory structure you have set up, this is just an example)

Populate the Antelope database

Now that you have the data streams ready in mseed format with proper headers and the LOG files in

mseed format, you can proceed to create the Antelope database, which will generate the products to be sent for archiving.

- By running **dbbuild** – you will generate the Antelope relational database containing all the information required to generate the dataless or meta-data.

- By running **miniseed2days** – you will generate the mseed day volumes per station/channel/day from all the mseed files created by **rt2ms**.

- By running **log2miniseed** – (explained in previous session) you will generate the log mseed day volumes per station/channel/day from the log files generated by **rt2ms** with the -L option.

- By running **dbverify** and **dbversdfwf** – you will execute checks on traces and database to verify the integrity of the database.

- By running **mk_dataless_seed** - you will generate a dataless using all the information from the database tables created when you ran dbbuild, remember the dataless doesn't contain any data, the information on it is basically all the information on your batch file and instrument response information provided by antelope when you run dbbuild)

- By running **seed2db** – you will run verification on the dataless if used as a check tool, alternatively you could use seed2db and the dataless to generate a database from the datales.

Build the Antelope database

Here you will use the batch file previously created with the configuration for each station in your network and will run **dbbuild** to create the Antelope database:

```
<my_cpu:EXPT> dbbuild -b your-database your-batch-file >& my_dbbuild.out
```

IMPORTANT**:**

- The configuration for each station in your batch file agrees 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 you have used any), sensor orientation, start times, close statement, etc.
- It is best to use a start time for each the station, which is conservative (i.e. a little early rather than milli-seconds late).
- The option –b in the command is to run dbbuild in batch mode (using the batch file)

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

: loading your-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 a new directory, "response" is created (there is a second directory created by Antelope 5.0 and later called "nom_response") . These tables and directories are the constituents of the database.

<my_cpu:EXPT> 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*	

View your database

Using **dbe** you can visualize the current information in your database. At this point there are no waveforms, only the information populated into the database from the batch file and the responses directory (/opt/antelope/current_version/responses and opt/antelope/current_version/instruments/)

<my cpu> dbe my_db

dbe is a general purpose tool for examining, exploring and editing Antelope relational CSS databases. For a detailed description on how to use **dbe** please read man dbe.

Create mseed day volumes and add them to your database

By running **dbbuild,** you now have a database that describes your network, however you have not associated any waveforms with the meta-data.

BUGS and CAVEATS: For larger experiments, sometimes it's necessary to increase the file descriptor limit because miniseed2days will keep filling file descriptors until it completes. In order to make sure that it is able to read and convert all of your files, for a Mac, use launchctl. In the command line type:

<my_cpu:EXPT> launchctl limit maxfiles 10000

On a linux and SunOs simply type:

<my_cpu:EXPT> unlimit descriptors

For more information regarding this caveat, please see the man pages for miniseed2days.

To add your waveforms details to your database, use the command **miniseed2days** with the –d option

as shown below**.** 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:EXPT> miniseed2days –d my_db -u –w
"day_volumes_S/%{sta}/%{sta}.%{net}.%{loc}.%{chan}.%Y.%j" rt2ms >& my_minised2days.out

Where:

- w specifies an alternate pattern for the output miniseed volumes. This pattern dictates the way the data records are allocated to files. PASSCAL requires the following format for quality control purposes:

  "day_volumes/%{sta}/%{sta}.%{net}.%{loc}.%{chan}.%Y.%j"

-d db Runs **miniseed2db** on the output files to create the wfdisc table

rt2ms Is the name of your input directory where your mseed files live

-DU When -D is specified duplicate seed blocks are reatined, while -U is a more limited method to supress duplicates

Notice: with the –d option Antelope calls the tool **miniseed2db** and adds the generated miniseed day files to the database

*Recall you can include the service number or other succinct identifiers in the directory names.

IMPORTANT**:**

- The mseed headers read by **miniseed2days** are the source of information used to populate the database's waveform table and name the files using the information contained in the data for the station (sta), network (net), location code (loc), channel name (chan), year of the data (Y) and julian day (j) as in: day_volumes/%{sta}/%{sta}.%{net}.%{loc}.%{chan}.%Y.%j"
- You must ensure the mseed headers in the station-channel-day files produced by **miniseed2days** are correct. 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.
- If you would like to avoid typing the long option for the file description with the –w option, please check our additional documentation in the appendixes on how to modify the miniseed2days.pf .

Assign calibration values from calibration table to wfdisc

To ensure that the calibration values are incorporated into the newly created wfdisc table, please run **dbfix_calib. This step** is performed for the integrity of the database and correlates information from the data (waveforms) with the dataless

<my_cpu:EXPT> dbfix_calib my_db

Verify the integrity of your database

Before you create a dataless you will want ensure that your meta-data completely describes the waveform data, and that your database is error free. **Read** the man page on **dbversdwf** and **dbverify** for other tests you can run on the database. Examples of suggested tests are:

<my_cpu:EXPT> dbversdwf -dtu my_db >& my_dbversdwf_output
0 bad files
0 bad records

<my_cpu:EXPT > dbverify –tj my_db >& my_dbverify.out
0 problems

Common errors associated with these commands can be found in our appendix called "Troubleshooting and identifying errors in basic data processing with antelope-Archiving oriented".

Please read the outputs and if you still have any questions about them, please e-mail the datagroup at data_group@passcal.nmt.edu before submitting your data and dataless.

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:EXPT > 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 can provide the name of the dataless-seed filename to use. 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
dbname is the name of your database
my_db name of your database

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

Verify the dataless

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

<my_cpu:EXPT> seed2db –v my_db_dataless_seed

Where:

my_db_dataless_seed is the name of your dataless, i.e., *PI.04.my_db.20042082000.dataless (from the example above)*

IMPORTANT: 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.

Send Data to IRIS/PASSCAL

stadayvols

When ready to submit the data to PASSCAL, please make sure that you all your data are complete and the dataless fully describes the network. One simple way to verify all data for each station/Julian day is complete is by running the PASSCAL tool called **stadayvols**, which is part of the PASSCAL software release.

<my_cpu:EXPT> stadayvols -d my_db -f day_volumes_S >& stadayvols.out

What is a full miniseed day volume?

At PASSCAL a full miniseed day volume is compound by each data stream and state of health channel defined on the dataless per station and Julian day. Thus, if in a dataless there is description for a STS-2/RT130 station with 2 data streams (e.g. 40sps and 1sps) recorded in 3 channels each (Z,N,E => BHZ,BHN,BHE & LHZ,LHN and LHE respectively) from day 001 2010 until day 365 2010, a full miniseed day volume expected for each Julian day will include:

Full miniseed day volume STS-2/RT130 = BHZ+BHN +BHE +LHZ+LHN+LHE +VM1+VM2+VM3+LOG (miniseed SOH)

What is the purpose of running **stadayvols** before sending the data to PASSCAL?

stadayvols - Determines what files in the day_volumes directory (or your input directory name) can create complete station day volumes (vols). The channels needed for completeness are taken from the css db "db" which must have sitechan, snetsta, and chanloc tables. If an 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.

IMPORTANT NOTE: StaDayVols.py does NOT provide input on possible GAPS on the data, it only verifies if all the day volumes (net/station/channel/loc code/ and time frame) described in the dataless EXIST on the indir directory.

When using **stadayvols** to send data there is NO need to use the –i option, the purpose of this tool is for verification of what is complete or what is missing before sending data to PASSCAL for archiving. By typing in the command line **stadayvols** –h you will get the usage of this tool:

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

Once you have verified that your data are complete, 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.

Sending data – gui_DoFTP and con_DoFTP

There are two options for data submission via FTP:using the command line or a GUI. We recommend the use of **gui_DoFTP** to submit data to the PIC (current version 2008.038 or later version). See the appendix 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/content/software-resources

<my_cpu:EXPT> gui_DoFTP

DoFTP will:

- Descend the specified directory path, identify, and pack ALL miniseed 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
- 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 much free disk space as the size of the dataset 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:EXPT> con_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:

<my_cpu:EXPT> ./con_DoFTP -a -f -r 366 -t 15 /Users/kxu/FA_tremor

Adding more data: future services

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

The answer is yes. You just need to be consistent, do the initial quality control on your data and follow the same steps previously described. If during this new service runchanges 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. Follow the same steps for data reduction and timing quality control as you did 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 service runs

a. Update the batch file (if needed)

At this point you already have 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 was neverrecorded for this station there's 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.

IMPORTANT:
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. There's also no need to build a new dataless as no new information has been added. If one of the three points were required then you will need to update your database with **dbbuild** as shown below:

<my_cpu:EXPT> dbbuild -b my_db your_updated_batch_file

*What is batch_mynet? I'm assuming it should by your-batch-file (as used before), if we keep the names consistent this helps the user follow where you're going.

c. View your database- same as before.

d. Add your waveforms to the database

Once you have the new data ready to add to the database (QC done, timing issues evaluated, headers fixed, etc), you can 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 say you have it under service2, then you will run:

e. Verify the integrity of your database -same as before

f. Create a new dataless SEED volume if you ran **dbbuild** (revised or rebuilt the database)

g. Verify your dataless file and rename to conform to convention

Updating the meta-data without processing new data

All changes (changed/new stations/removed stations) in your network configuration must be described within 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 generate a dataless within it. The steps you should follow are:

a. Create a temporary 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 your_ updated__**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 into 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.

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 much data volume is 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 have 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.

*IRIS/PASSCAL Documentation- Created by Eliana Arias (*eliana@passcal.nmt.edu*, 2006,2007)*
Revised Bruce Beaudoin (2006,2007,2008)
Revision by data group (2009-2010)
Revision 8 by Mouse Marie Reusch 24 January 2011
Revision 9 Katyliz Anderson, September 2012
Revision 10 Eliana Arias Dotson October 12, 12