Archiver.pl

(copied from the sdg wiki: https://sdg.rap.ucar.edu/confluence/display/crosspgm/Archiver+Documentation to make more available outside RAL).

Archiver.pl is a general purpose Perl script for archiving data to the mass store. It (optionally) works in conjunction with a mySQL database for storing verification data and meta-data. The mySQL database has a front-end for monitoring at http://sdg.rap.ucar.edu/archive written in PHP/AJAX.

Table of Contents

Features

- Files can be compressed (or not).
- Files can be TAR'd (or not).
- Files can be staged (or not).
- TAR'd files can have a table of contents created and stored along with the .tar file.
- Expected file sizes can be verified locally & on the mass store.
- Expected number of files can be verified locally & on the mass store.
- Records of each archive are stored in a mySQL database, and accessible via web interface.
- Warnings and errors can be automatically emailed.
- Warning levels are configurable.
- Archive date can be set relatively (yesterday, -72 hours) or absolutely (2007-07-01).
- HPSS options are configurable (# of copies).
- Config files written in XML.

Usage

Archiver.pl is located in cvs: *cvs/apps/archive/src/Archiver/Archiver.pl*

Arguments

You can use the -help option to see a general usage statment:
(see the Options/Configuration section below for details)

```
Usage: ./Archiver.pl -config PATH [optionalArgs]

        -config PATH             path to the config file

OPTIONAL ARGUMENTS
        Cmd Line Only Arguments
        -----------------------
        -h              Display a shorter help message
        -help           Display this help message
        -printParams    Write a example config file to stdout

        Cmd Line/Config Options
        -----------------------
        -test                           don't actually msrcp, etc. only log cmds
        -debug                          prints some basic debug info
        -verbose                        prints more detailed debug info
        -dateString string              Use this date string for date substitutions
        -projectNum num                 Charge this project for GAUs
        -tmpDir path                    Use path for staging,TARing, etc.
        -verificationEmail email        Use this email address for reporting errors and warnings.
        -doTar/-nodoTar                 should we create a .tar file?
        -doZip/-nodoZip                 should we compress files first?
        -doMSS/-nodoMSS                 should we send files to the mass store?
        -doClean/-nodoClean             should we clean up our tmp files?
        -forceClean/-noforceClean       should we clean up our tmp files even if there was an error?
        -retentionPeriod num            Number of days to set the retention period to.
        -doTarList/-nodoTarList         should we create & store a TOC along with the tar file?
        -readPassword pw                use pw as the read password
        -writePassword pw               use pw as the write password
        -classOfService string          pass string to msrcp's class of service argument
        -passwordClarity level          if level is clear store passwords as clear text
                                        If level is 'obscure', obscure passwords first.
                                        Anything else, don't store passwords.
        -doStaging/-nodoStaging         if true, copy files to tmp dir before working on them
        -warningLevel float             if expected number of files/file size do not fall within this
tolerance, then warn
        -skipUnderscoreFiles            This only works if you are staging or TARing
        -forceOverwrite                 Uses put instead of cput and will ovewrite existing files.
        -doSQL/-nodoSQL                 Should meta data be stored in the SQL database


Please see the documentation online for more details: https://sdg/confluence/display/crosspgm
/Archiver+Documentation
```

Config Files

You can use the -printParams option to see an example config file.
(see the Options/Configuration section below for details)

```
<archiverConfig>
<dateString>-24 hour</dateString>
<debug>true</debug>

<tmpDir>/d1/rapdmg/tmp</tmpDir>
<projectNum>48500002</projectNum>
<archiveRunComment>The archiving run for LDM nids/nowrad/other data for DATEYYYYMMDD (run 24 hours later)<
/archiveRunComment>

<!-- NIDS DATA -->
<archiveItem>
<source>/ldm1_d2/nids/raw/nids/*/BREF1/DATEYYYYMMDD</source>
<destination>/RAPDMG/LDM/ARCHIVE/DATEYYYY/DATEMMDD</destination>
<cdDirTar>/ldm1_d2/nids/raw/</cdDirTar>
<expectedNumFiles>25000</expectedNumFiles>
<expectedFileSize>245000000</expectedFileSize>
<tarFilename>DATEYYYYMMDD_all.nids.tar</tarFilename>
<comment>NEXRAD Information Dissemination Service</comment>
<dataType>radar</dataType>
<dataFormat>nids</dataFormat>
</archiveItem>

<!-- Information regarding GFS data comes from http://www.unidata.ucar.edu/data/conduit/ldm_idd/gfs_files.html
-->
<archiveItem>
<source>/ldm3_d2/grib/GFS002/DATEYYYYMMDD</source>
<destination>/RAPDMG/grib/GFS002</destination>
<expectedNumFiles>68</expectedNumFiles>
<dataFormat>grib</dataFormat>
<dataType>model</dataType>
<comment>2.5x2.5 degree lat/lon grid (Hours F192-F384)</comment>
</archiveItem>

<archiveItem>
<source>/ldm1_d2/NLDN/DATEYYYYMMDD*</source>
<destination>/RAPDMG/LDM/ARCHIVE/DATEYYYY/DATEMMDD</destination>
<tarFilename>DATEYYYYMMDD.nldn.tar</tarFilename>
<dataFormat>binary - http://www.unidata.ucar.edu/data/lightning.html</dataFormat>
<dataType>lightning - ground sensors</dataType>
<comment>United States National Lightning Detection Network (NLDN) located at SUNY at Albany - THIS DATA HAS
RESTRICTIONS ON ITS USE and DISTRIBUTION</comment>
</archiveItem>

<!-- BAD TAILS FILE ON RUMPUS -->
<archiveItem>
<source>/d1/ncar/rap/projects/InsituTurb/ingestHome/params/badTails.txt</source>
<destination>/RAPDMG/InsituTurb/badTailFiles/DATEYYYY/DATEMMDD/badTails.txt</destination>
<dataFormat>Ascii</dataFormat>
<dataType>Ascii</dataType>
<doTar>false</doTar>
<doZip>false</doZip>
<comment>badTails.txt file</comment>
</archiveItem>

</archiverConfig>
```

The config files used to archive the LDM data can be checked out from cvs/projects/rapdmg_archive/

Logging

All warnings, errors, debug info, diagnostics, etc. in Archiver.pl is sent to stdout/stderr. When running via cron, I recommend piping the output to LogFilter as shown below.

I recommend cleaning up the log files with a simple find in cron. Again see the example below.

A typical crontab entry

```
# Archive grib data nightly
0 1 * * *   csh -c "/home/rapdmg/cvs/apps/archive/src/Archiver/Archiver.pl -conf ~/cvs/projects/rapdmg_archive
/Archiver.grib.conf -dateString `date -dyesterday +\%D` -verbose |& LogFilter -d /home/rapdmg/logs/`date -
dyesterday +\%Y\%m\%d` -p Archiver -i grib"
```

```
# Purge all log files older than 30 days, and removed empty directories
0 * * * *  csh -c "find /home/rapdmg/logs -mtime +30 -name Archiver*log  -exec rm \{\} \;"
0 * * * *  csh -c "find /home/rapdmg/logs -depth -type d -empty  -exec rmdir \{\} \;"
```

Running on non-realtime (i.e. archive) data

See Auxiliary Scripts section below rerun_archive.py.

Testing

You can test by setting the **test** option to *true* as described below. This will generate and log the command lines that the script would run, but will not execute those commands. Unfortunately testing in this manner, can generate lots of spurious errors/warnings in the log, due to dependencies of later steps on successful completion of earlier steps that are not run in test mode.

Another way to test is to set the **doArchive** option to *false* as described below. This will allow you to test every aspect of the script right up to the point where the data would be sent to the mass store.

Setting **test** to *true* or **doArchive** to *false* will keep meta-data from being stored in the mySQL database.

Monitoring

The verificationEmail, warningLevel, and verify options can be used to configure a status email. You can also monitor archives, and search archives using the web interface at http://sdg.rap.ucar.edu/archive.

Most problems reported via email are minor, just not as much data as expected. If you get a lot of false positives it is worthwhile to loosen up your error checking with the *warningLevel* option.
If you do get a real problem it may look like this:

```
ERROR!: Non-zero return code:1
MKDIR for /RAPDMG/AOAWS/2013/0929 FAILED!
LOCAL: /taiwan_data1/aoaws/spdb/metar/20130929*
Archive: /RAPDMG/AOAWS/2013/0929
```

or this:

```
FAILURE! - Archive target: </rapdmg2/data/grib/HRRR-wrfnat/20140124> does not exist!!
```

If you see a Archive COUNT of 0, some times, the data did get up ok, but the hpss ls
failed because the HPSS was being flakey. I usually just copy and
paste the HPSS dir into a hpss ls command to see if anything got up there:

```
%> hsi ls /RAPDMG/grib/Eta104/20070427
```

If there is data in that dir, then we are ok. If not, then we need to
rerun the archiver to resolve this.

HPSS return codes

Unlike the MSS the HPSS can send up a partial file. Archiver.pl attempts to detect this by checking return codes from the *hsi put* command. If Archiver.pl sees a non-zero return code (indicating an error), it will attempt the put a second time, and log this message:

```
HSI PUT HAS FAILED!!! - returned: 72 retrying...
```

If this second attempt also fails, it will give up and log this message:

```
ERROR!: Non-zero return code:72
        LOCAL: /d1/prestop/tmp/Archiver.pl-s15LTVH95T/Item1-Tar/20110301.nldn.tar
        Archive: /RAPDMG/LDM/ARCHIVE/2011/0301

Try re-running this command: hsi -a 48500052 put -P -A \"United States National Lightning Detection Network \
(NLDN\) located at SUNY at Albany - THIS DATA HAS RESTRICTIONS ON ITS USE and DISTRIBUTION\" copies=1  /d1
/prestop/tmp/Archiver.pl-s15LTVH95T/Item1-Tar/20110301.nldn.tar :  /RAPDMG/LDM/ARCHIVE/2011/0301/20110301.nldn.
tar
```

Rerunning the Archiver on a failed archive

Please see the Auxiliary Scripts section below for a simple way to resend a subset of failed data to the HPSS.

Alternatively you can also rerun Archiver.pl:

In this example, most of the products in the config got up ok, so we need to edit the config to only contain the Eta104 products:

```
%> cd ~/cvs/projects/rapdmg_archive/
%> cp Archiver.grib.conf Archiver.failedgribs.conf
%> emacs Archiver.failedgribs.conf &
```

Once that config has the products we need, just run the Archiver with the failed date specified on the command line:

```
%> /home/rapdmg/cvs/apps/archive/src/Archiver/Archiver.pl -conf Archiver.failedgribs.conf -dateString 20070427
```

You should get some output to stdout showing that it is staging, zipping, taring, and sending the data up to the mass store.

Disk Usage

Archiver.pl cleans up your tmp directories after each item is sent to the HPSS (assuming doClean is true). You need to have enough disk space in the temporary directory to store the largest tar file that you are creating. If you are staging the data, then in addition, you also need enough disk space to store all of the data of the largest archive item.

Options/Configuration

There are two levels of configuration. Top-level (i.e. Archive-Run) configuration options apply to the entire run of Archiver.pl, unless overridden. Archive-Item configuration options only apply to a single item to be archived.

Command line options are parsed with the Perl Getopt::Long library. Getopt::Long allows you to abbreviate options as long as their usage is unambiguous. Giving a boolean command line option sets it to TRUE. You can set these values to FALSE by prefixing it with a '!' or 'no'. Config file options are parsed with the XML::Simple library.

Priority

Options can be specified in three ways:

1. in the entry for a particular archive item in the config file
2. on the command line
3. in the top level of the config file

Options are given priority in the order given above (i.e. Command line options override top level config options, but not Archive-Item config options. Archive-Item config options are given the highest priority, and are never overridden.)

Archive-Run Options

Cmd Line	Top level config	Item level config	Example Values	Default Value	Description
-help			*bool*	false	Give basic usage information.
-test	X		*bool*	false	Commands are constructed but not actually run.

-debug	X		*bool*	true	Outputs commands before they are run, as well as basic debug info.
-verbose	X		*bool*	false	Outputs more debug info than debug.
-projectNum	X		48500052	n/a	The project # to charge
-dateString	X	X	'yesterday', '-48 hours', '20070102'	yesterday	The date/time that is used to generate year/month/day values for substituting in paths, filenames, and comments.
-config			myconfig.xml	n/a	The config file that is to be used. **This is the only required *command line* argument.**
-printParams			*bool*	false	If this is true, Archiver.pl prints out a sample config file and exits.
-verificationEmail	X		you@ucar.edu	n/a	If this is defined an email will be sent to this address if their are any warnings or errors. Multiple email addresses can be specified by separating them with commas.
-verify	X		quiet,full	full	When verify is set to 'quiet' verificationEmail will only receive emails if there are warnings or errors. When set to full, you will always get an email to let you know that everything ran ok.
-doTar	X	X	*bool*	true	Should the files be TAR'd up before being sent to the MSS?
-doZip	X	X	*bool*	true	Should the files be compressed before being sent to the MSS? NOTE: Files are zipped in place, unless doStaging is also true.
-doArchive	X	X	*bool*	true	Should the files be sent to the Archive?
-doClean	X		*bool*	true	Should temporary files be deleted? *note: tmp files are not deleted when an error occurs unless forceClean is also true*
-forceClean	X		*bool*	false	Should temporary files be deleted even in the case of an error?
-doSQL	X	X	*bool*	true	Should meta-data be saved in the SQL database?
-tmpDir	X		*string*	/tmp	Where should temporary files be placed. Archiver.pl creates temporary subdirectories in the directory given.
-doTarList	X	X	*bool*	true	If doTarList is true a table of contents file is created from the .tar file and put on the MSS with the .tar file. If the data file is filename.tar, the TOC file will be TOC.filename.tar.txt.
-doStaging	X	X	*bool*	true	If this is true, files are copied to a temporary directory before being zipped, TAR'd, etc.
-mode	X	X	*777*	none	If a mode is given, a chmod command will change the mode on files after they are sent to the server.
-numCopies	X	X	*1 or 2*	1	The number of copies of the data that are stored on the HPSS
-warningLevel	X	X	*float*	.95	Where **expectedNumFiles** or **expectedFileSize** are defined, this gives the minimum ratio below which warnings will be given. For example with the default warningLevel an expectedNumFiles of 100, will generate a warning if there are less than 95 files. warningLevel is used strictly for determining whether warnings are generted/emailed by the PERL script, it is not used to generate the colored indicators on the website.
-comment	X	X	*string*	none	This is a comment for the entire run if given at the top level config, or a comment for an individual archive item if given in a archiveItem block

-skipUnderscoreFiles	X	X	*bool*	false	If this is true, files beginning with an underscore are not archived. **NOTE: This ONLY works if you are staging or TARing your files**
-forceOverwrite	X	X	*bool*	false	If this is true, *hsi put* is used instead of *hsi cput*.
-posixGroup	X		ralicing	""	If this is defined, files put on the HPSS will be owned by the given group.

Archive-Item Options

Archive Item options can only be defined in the config within <archiveItem></archiveItem> tags. These options can not be defined on the command line, or outside of
<archiveItem> tags in the config file. Within each <archiveItem></archiveItem> group, a source and destination are required. A tarFileName is required if this
archiveItem is to be TAR'd. In addition to the archiveItem only options listed below, any options from the archiveRun options above which has an X in it's archiveItem column can be overridden within the archiveItem tags. None of the options below have default values.

XML Key Word	Example Values	Description
source	/ldm1_d2/ddp/DATEYYYYMMDD	This is the source of the data to be archived and is **required**. It must be on the local machine. Wildcards are allowed, as well as date substitution as described below.
destination	/RAPDMG/LDM/ARCHIVE/DATEYYYY/DATEMMDD	This is the destination on the mass store where the archive item will be placed and is **required**. Date substitution is allowed as described below. Do not put *mss:* at the beginning.
cdDirTar	/ldm1_d2/	If doTar is true for this archive item, then the tar file will only include the portion of the directory structure stored below this level. In this example, the tar file would have the following directory structure: ddp/DATEYYYYMMDD instead of the default: ldm1_d2/ddp/DATEYYYYMMDD. Wildcards are not allowed, but date substitution is allowed as described below.
cdDir	/ldm1_d2/	This is similar to cdDirTar (and is in fact an alias for that command - so either can be used interchangably). cdDir is used when data is not TAR'd before being sent to the MSS. If you have a wildcard in your source, then without this option the entire path to the data will be sent to the MSS. This option allows you to specify how much of the source path is placed on the MSS. Wildcards are not allowed, but date substitution is allowed as described below.
tarFilename	DATEYYYYMMDD.ddp.tar	**If doTar is true** for this archive item, **than this field is required**. Date substitution is allowed as described below. If two or more archive items have the same tarFilename, their sources, will all be added to the same tar file. See the section on appending multiple sources in a single tar file below.
expectedFileSize	500000000	If this field is defined, Archiver.pl will generate a warning if the ratio between the actual file size and this value is less than the *warningLevel*. Archiver.pl expects this size in bytes. This verification is not done if multiple files are defined by a single archiveItem, and they are not TAR'd before being sent to the mass store.
expectedNumFiles	130	If this field is defined, Archiver.pl will generate a warning if the ratio between the actual number of files and this value is less than the *warningLevel*.
dataFormat	netCDF,ascii,csv,grib	This field is basically a string comment that is stored on the mySQL database with the other metadata, and gives you the ability to better search/organize the metadata.

dataType	radar,model,satellite	This field is basically a string comment that is stored on the mySQL database with the other metadata, and gives you the ability to better search/organize the metadata.
comment	This data is downloaded nightly from Mr. Mxyzptlk's ftp server via a script on *host*.	This field is basically a string comment that is stored on the mySQL database with the other metadata, and gives you the ability to better search/organize the metadata.
mode	777	If a mode is given, a chmod command will change the mode on files after they are sent to the server.

Date Substitution

In many cases it is useful to have a date or date fragment in a path, comment or filename which is defined elsewhere. The is accomplished in Archiver.pl by having a *dateString* defined in the config file or on the command line. This dateString is passed to the date command via it's --date option, and therefore anything supported by the date command is a valid dateString. Possible values include relative dates like 'yesterday', 'today', and '-48 hours', as well as absolute dates like '20070101', '2007-01-01', and 'Jan 4 2007'. Because Archiver.pl depends on the version of *date* installed on your system, you should verify that your date strings work with your date command.

To use this derived date, simply put one of that valid Date Substitution strings in your filename, path, etc. For example */RAPDMG/ARCHIVE/DATEYYYY/DATEMMDD* would be converted to */RAPDMG/ARCHIVE/2007/0101* for the dateString 'Jan 1 2007'. See below for details on which fields Date Substitution is performed on, and which Date Substitution strings are valid.

Fields with Date Substitution

Date Substitution is done on five fields:

- ArchiverRun.comment
- ArchiveItem.source
- ArchiveItem.destination
- ArchiveItem.cdDirTar
- ArchiveItem.tarFilename
- ArchiveItem.comment

Valid Date Substitution Strings

- DATEYYYYMMDD - Year, Month, DayOfMonth
- DATEYYYY- Year
- DATEMMDD - Month, DayOfMonth
- DATEHHMM - Hour, Minutes
- DATEYY - 2 digit Year
- DATEMM - Month
- DATEDD - DayOfMonth
- DATEHH - Hour
- DATEJJJ - Julian Date (Day of year)

Environment Variables

The *source*, *destination*, *tmpDir*, *tarFilename*, *cdDir* and *cdDirTar* fields allow environment variables in their values. Environment variables start with a '$', and contain upper or lower case letters and the underscore character ('_'). Environment variables end at the first non-valid character.

Example
<source>$RAP_LIB_DIR/archiveTest1/DATEYYYYMMDD</source>

Appending multiple sources to a single TAR file

If multiple archive items have the same string for their *tarFilename*, then the data defined by these sources will all be added to a single tar file. The tar file will only be sent to the mass store once, when the final archive item with that *tarFilename* is processed. Using multiple identical _tarFilename_s has some consequences:

- *expectedFileSize* and *expectedNumFiles* must be running totals. These numbers are calculated by looking at the contents of the tar file, so the target numbers must include not just the expected sizes for their associated archive item, but also all previous archive items with an identical *tarFilename*.
- Since only a single entry is added to the meta-data database for a group of archive items, *dataFormat*, *dataType*, and *comment* fields are ignored for all but the last archive item of a set of archive items with identical *tarFilename*.

    - The *source* field stored on the meta-data database will be a concatenation of the *source* of all the individual archive items, separated by commas.
- Since only a single tar file is sent to the mass store for a group of archive items, *numCopies* and *doArchive* are ignored for all but the last archive item of a set of archive items with identical *tarFilename*
- Begin and End times stored in the meta-data database will refer to the begin and end times of the final archive item in the set of items with an identical *tarFilename*.

Dependencies

Email

If you want to receive emails from Archiver.pl, you need to have sendmail installed and in your path. It is usually located at /usr/sbin/sendmail

Perl/MySQL

Archiver.pl **requires** installation of the debian package *libdbd-mysql-perl*, *libdbi-perl*, and *libxml-simple-perl*.

HPSS

If you want to send things to the mass store, you need to have the *hsi* command in your path.

Kerberos

You need to be authenticated by Kerberos when you are doing archiving. This may get integrated into the Archiver.pl script once the Kerberos authentication is better understood.

Limitations

- If you are sending up a single file, you have to specify that filename as part of the destination also. If you give only the directory for the destination, behavior may be incorrect. (See the badTails.txt archiveItem in the example given above and as the result of the -printParams command line option)
- If you are sending a directory up without TARing it up:
    - NO FILE SIZE VERIFICATION - it will not try to verify file sizes on local disks or the MSS.
- **If you use a relative date (like 'yesterday'), and your program runs over a date boundary**, the date being used will change. WORKAROUND - Specify the date string on the command line like this:
    -

        ```
        Archiver.pl -conf Archiver.wrf.conf -dateString `date -dyesterday +%D`
        ```

        note that the '%' must be escaped if used in crontab

- If you specify doZip (the default), files will be zipped in place. If you do not want your files zipped, make sure to also turn on doStaging (also the default).

Auxiliary Scripts

There are other scripts checked into cvs/apps/archive/src/Archiver that you may find useful.

If you have errors with an archive run, I recommend running diffHSI.csh first to verify that the locations you are comparing are the ones you expect, and then resendHSI.csh if diffHSI.csh finds differences.

diffHSI.csh

diffHSI.csh takes a local path and a HPSS path, and compares the files (name and size) between both of them. Any files that are in the local path but not the HPSS path are printed to stdout.

```
hoe:~/bin> ./diffHSI.csh /rapdmg1/data/grib/WRF-RR-wrfnat/20110404 /RAPDMG/grib/WRF-RR-wrfnat/20110404
getting local file list
getting hsi file list
20110404_i18_f002_WRF-RR.grb2.gz
```

resendHSI.csh

resendHSI.csh works much the same as diffHSI.csh, except instead of printing the missing files to stdout, it resends them to the HPSS.

```
hoe:~/bin> ./resendHSI.csh /rapdmg1/data/grib/WRF-RR-wrfnat/20110404 /RAPDMG/grib/WRF-RR-wrfnat/20110404
getting local file list
getting hsi file list
hsi put /rapdmg1/data/grib/WRF-RR-wrfnat/20110404/20110404_i18_f002_WRF-RR.grb2.gz : /RAPDMG/grib/WRF-RR-wrfnat
/20110404/20110404_i18_f002_WRF-RR.grb2.gz
Username: rapdmg  UID: 8752  Acct: 48500052(P48500052) Copies: 1 Firewall: off [hsi.3.5.7 Mon Feb 7 12:17:30
MST 2011]
put  '/rapdmg1/data/grib/WRF-RR-wrfnat/20110404/20110404_i18_f002_WRF-RR.grb2.gz' : '/RAPDMG/grib/WRF-RR-wrfnat
/20110404/20110404_i18_f002_WRF-RR.grb2.gz' ( 104546403 bytes, 9190.6 KBS (cos=1012))
```

rerun_archive.py

rerun_archive.py takes a Archiver configuration file as well as a begin & end date range. It iterates over all days between begin & end and calls Archiver.pl with the date & config file.

```
%> ./rerun_archive.py  20140402 20140616 ~/archiverConfs/Archiver.ppi.conf
```

Future Work

There are a number of additional features that should be considered for inclusion in Archiver.pl if more development is done.

- The htar command which combines tar & hsi put into one statement is supposedly more efficient and may be faster and offer other advantages over the current approach.
    - http://www.sdsc.edu/us/resources/hpss/htar.html
- We need some utilities that better integrated the MSS & archiver mySQL database. For example a ArchiverRM that removed the path from the MSS & all references to it in the mySQL db.
- The $TMPDIR env variable should be used for the default tmp dir if it is set wikipedia reference
- Allow expected file size in MB, KB, GB, etc. (Now it has to be in bytes).
- When doing a file count for verification, check that the files have at least 100 bytes or at least were not zero size.
- I think it is not strictly necessary to handle a source with an '' **at the very end as a wildcard source. (e.g. if the source is '/d1/data/**' we should maybe undef @expandedSource).
- Counting the number of files in a .tar is done via:

  ```
  tar tf scripts.tar | wc -l
  ```

  This counts directories also, so it would be better to just count files, but that would be much more complicated.

- It would be very nice to validate the xml config against a dtd (or more likely an .xsd file).
    - Several resources for validation in XML
    - Good XML Schema Tutorial
    - This would probably require users to put a link to the .xsd file in their config.

- Can I verify somehow that things like gzip, tar, etc. didn't fail? I guess the return from system() is maybe $?.
- Archiver.pl should respect a .lock file (or some equivalent), that is in a directory, and either wait and retry or come back to the directory later. This would allow other processes that are in the midst of modifying those files, to tell the archiver to wait.
    - This desirement came about due to a find/zip via cron interfering with data being archived, because it renamed files out from under Archiver as they are being copied the staging area. Other solutions might include Archiver.pl retrying a copy after a incomplete copy is done, or having Archiver.pl put a .lock file in the dir, that other programs must respect.
- Remove the limitations listed above.
- Zip TOC files before putting on the mass store
- Change defaults on the website so that 'exact' is not checked.
- Add regular expression support to the selection criteria used by the web interface.
- Add regexp support in the script itself for specifying files to be archived.

Known Bugs

- Not really a bug, but if the script dies in the middle of execution (most often due to the MSS going away), you can be left with files not cleaned up in your tmp area. It is a good idea to clean it up by hand every once in a while, or have a cronjob that deletes older files from the tmp area.
- A Wildcard as the top level directory of a source will cause problems.
- You can not have quotes in your archive item comments. Comments have been disabled because of this.
- You can not have single quotes (') in any fields that are stored in the mySQL DB.
- The website is not working under IE6 (via rdesktop - vise)
- skipUnderscoreFiles only works if you are staging or TARing.
- If <doZip> is true (the default), and files are already zipped, and your source is a directory, the script will pause to ask if you want to zip already zipped files if you are running interactively.

See Also

- Archiver_Design
- Archiver_MySQL_Notes
- MSS to HPSS Official Transition Website
- MGleicher.us - HSI developer