LTFSArchiver 1.0 Configuration and Administration

Summary

This document describes in detail the configuration of the system and its management. It is explained for example how to start and end the system service (ltfsarchiver) and how to keep track of what is going on through log files.

1. Configuration file(s)

The main configuration file of LTFSArchiver is:

`/opt/ltfsarchiver/conf/ltfsarchiver.conf`

The config file has four sections:

Section 1: Runtime parameters and variables section

`LTFSARCHIVER_TITLE` and `LTFSARCHIVER_VERSION`
These strings are shown on the web interface to present the tool.

default:
```
LTFSARCHIVER_TITLE="LTFS Archiver"
LTFSARCHIVER_VERSION="1.0"
```

`LTFSARCHIVER_HOME` and `LTFSARCHIVER_LOGDIR`
These are the paths to home and log directories

default:
```
LTFSARCHVER_HOME="/opt/ltfsarchiver"
LTFSARCHIVER_LOGDIR="$LTFSARCHIVER_HOME/logs"
```

`LTFSARCHIVER_LOGLEVEL` and `LTFSARCHIVER_DEBUG`
`LTFSARCHIVER_LOGLEVEL` is used to set the verbosity of the log (0=minimum, 4=maximum)
If `LTFSARCHIVER_DEBUG` is set to “1”, the log strings (those that have to be printed according to `LTFSARCHIVER_LOGLEVEL` value) will be also sent to standard output (useful when manually launching LTFSArchiver as a script, instead of service)

default:
```
LTFSARCHIVER_LOGLEVEL=1
LTFSARCHIVER_DEBUG=0
```

`LTFSARCHIVER_USER`, `LTFSARCHIVER_GROUP` and `LTFSARCHIVER_DB`
The first two parameters define the user name and group used as UID and GID during the ltfs mount procedure.
The `LTFSARCHIVER_USER` has also to be the owner of the database used by LTFSArchiver, whose name has to be set through the `LTFSARCHIVER_DB` variable

default:
```
LTFSARCHIVER_USER="pprime"
LTFSARCHIVER_GROUP="pprime"
LTFSARCHIVER_DB="ltfsarchiver"
```

`LTFSARCHIVER_LTFSTIMEOUT`
Is the time in seconds LTFSArchiver has to wait after ltfs' unmount command (if mount issued in RW mode) before considering succesfully completed the eject/recovery tape.
After an umount command, in fact, ltfs needs some time to physically update the index partition. Removing the tape before the index update (even if physically allowed by the

device) may cause a filesystem corruption(see also `LTFSARCHIVER_LTFSSYNC` parameter)
This parameter is ignored when unmountig a R/O ltfs. Please be careful when considering to lower the default that is set to 120 seconds.

Default:
```
LTFSARCHIVER_LTFSTIMEOUT=120
```

`LTFSARCHIVER_MAXRESTORE_PRIORITY`
Reserved for future uses, it is intended to set a priority rate between restore and other requests.

Default:
```
LTFSARCHIVER_MAXRESTORE_PRIORITY=0
```

`LTFSARCHIVER_INTERRUN`
It is the number of seconds between each agent run. Lower values assure more promptness of the system but also a higher overhead.

Default:
```
LTFSARCHIVER_INTERRUN=60
```

`LTFSARCHIVER_LTFSSYNC`
Determines the mode in wich ltfs will executes the sync of the index partition (see also `LTFSARCHIVER_LTFSTIMEOUT` parameter). Used only when the LTO Tape is mounted in R/W. Accepted values are the same accepted by the system command ltfs mount: *time, close, unmount.*
The *unmount* mode is preferred (the default) because if the archive procedure has to write a lot of files, *time* and *close* modalities can cause frequent index updates and consequent long rewind/update/fast forward operations).
Using *unmount* instead, the index update will be executed only once per run.

Default:
```
LTFSARCHIVER_LTFSSYNC="unmount"
```

`LTO_ALLOWED_CODES and LTO_ALLOWED_TYPES`
These arrays (they must be set as arrays even if you want to configure LTFSArchiver to allow a single LTO type) contains the Density Code(s) that identify the different LTO types and their well-known names.
Of course, the nth density code of the first array must match with the nth LTO type.
Default:
```
LTO_ALLOWED_CODES=( 0x58 0x99 ) # 0x99 for LTO6 is not actual
LTO_ALLOWED_TYPES=( LTO5 LTO6 )
```

`LTO_WATERMARK`
This array must have the same number of elements as the preceeding two.
Each element represents the minimum available space (in MB) for a tape (according to his known type) to be considered available for a new archive request. This is done because ltfs does not allow to fill a tape more than 98-99% of its declared capacity. If a LTO5 tape has less than about 22 GB of free space, the tape will be mounted in R/O mode even if

R/W was specified.

Default:
```
LTO_WATERMARK=( 20200 30200 ) # 30200 for LTO6 is estimated
```

`LTFSARCHIVER_LTFSRULE`
It is the string sent to the system command *mkltfs* command when formatting a new tape.
This is done to redirect the archiving of files smaller than what specified (1MB as default) to the index partition instead of data partition of the tape.
Please note that changing this parameter can lower the performances when archiving a whole folder.

Default:
```
LTFSARCHIVER_LTFSRULE='size=1M'
```

`LTFSARCHIVER_MAXAVAIL`
Max concurrent active instances of makeavailable requests.
When a tape is loaded due to a "makeavailable" request, it will lock both the tape and the drive device during the time it runs. Please remember to leave at least one drive free to avoid blocking of the elaborations.

Default:
```
LTFSARCHIVER_MAXAVAIL=2
```

`LTFSARCHIVER_MNTAVAIL`
Is the "base" mountpoint where the makeavailable access point will be created e.g., if the tape `"EX000000"` is made available, its mount point will be:
`$LTFSARCHIVER_MNTAVAIL/EX000000`

Default:
```
LTFSARCHIVER_MNTAVAIL="/mnt/pprime/lto-ltfs"
```

`LTFSARCHIVER_LOCK_LABEL`
A sort of "dummy label" used to put a tape device in unusable status through the web interface (see "LTFSArchiverWebGUI").
An attempt to use Tapemanager/Add command to add a tape with this label will be refused.

Default:
```
LTFSARCHIVER_LOCK_LABEL="donotuse"
```

`GUESSED_CONF`
It's the name of the auto-configuration file generated by the script `/opt/pprime/sbin/utils/guess_conf.sh` (see further).

Default:
```
GUESSED_CONF=$LTFSARCHIVER_HOME/conf/guessed.conf
```

Section 2: Paths to command and db access

It includes the major paths to the OS commands used for the core activities.
This section should not be modified, except if a non-default version of a command or utility has to be used.

```
MTX_CMD=`which mtx`
MT_CMD=`which mt`
PSQL_CMD=`which psql`
LTFS_CMD=`which ltfs`
RSYNC_CMD=`which rsync`" -va"
DBACCESS="$PSQL_CMD -U $LTFSARCHIVER_USER -d $LTFSARCHIVER_DB -t -c "
DBACCESS_HTML="$PSQL_CMD -U $LTFSARCHIVER_USER -d $LTFSARCHIVER_DB -H -c
"
```

Section 3: Operating mode

`LTFSARCHIVER_MODE`
This parameter sets the operating mode used by LTFSArchiver, accepted values are:

`"B"`: this corresponds to the Mixed mode, LTFSArchiver will hence use both internal (library managed) and external (manually managed) LTO tape devices.

Effects: Each archive request will be evaluated to verify if an existing internal tape (loaded into a managed library) can be used to write the data on it. If a tape is found, the request will be queued in “library mode”.
If no internal tape can be used, the system will try to find an external usable one (a tape not loaded into a library but known). If such a tape is found, the request will be queued in “manual mode”.
If also this search fails, the request will be refused (fallout status).

`"C"`: LTFSArchiver will use only internal (library managed) LTO tape devices.
Effects: Each archive request will be evaluated to verify if an existing internal tape can be used to write the data on it. If a tape is found, the request will be queued in “library mode”.
If no internal tape can be used, the request will be refused (fallout status).

`"M"`: LTFSArchiver will use only external (manually managed) LTO tape devices.
Effects: Each archive request will be evaluated to verify if an existing external tape can be used to write the data on it. If a tape is found, the request will be queued in “manual mode”. If no external tape can be used, the request will be refused (fallout status).

Default:
`LTFSARCHIVER_MODE="M"`

`CHANGER_TYPE`
Used to set the type of external library used.
For now the only accepted value is `"MSL"`, standing for HP MSL-xxxx tape libraries.
In the future it's expected to extend support to other libraries.

Default:
`CHANGER_TYPE="MSL"`

`TAPE_TYPE`

Used to set the type of tape device used.
The only accepted value is `"LTO"`, that supports LTO5 and LTO6 devices.
This is kept for compatibility with older version of data archivers.

Deafult:
```
TAPE_TYPE="LTO"
```

`CHANGER_ADDRESSES` (Not yet implemented)

Array containg the URLs of the administrator web interface for MSL libraries.
If more than one library is used, a blank must be used as a separator.
Useful only if the library(ies) will be network connected.

Default:
```
CHANGER_ADDRESSES=( http://198.162.0.2 )
```

Section 4: Library and tape devices (see usage of guess_config further)

`CHANGER_DEVICES`
String array containing the physical device name(s) of the used library(ies), a blank must be used as a separator.
Sample of a system with two libraries with device name sg5 and sg8:
```
CHANGER_DEVICES=( /dev/sg5 /dev/sg8 )
```

Default:
```
CHANGER_DEVICES=( /dev/sg5 )
```

`CHANGER_SLOTS`
String array containing the number of slot (Storage Element) available for each library.
A blank must be used as a separator.
The order must be the same used for libraries with the `CHANGER_DEVICES` array.
Referring to previous sample, and supposing that sg5 library has 24 slot and sg8 has 48 slots, the array must be:
```
CHANGER_SLOTS=( 24 48 )
```

Default:
```
CHANGER_SLOTS=( 24 )
```

`TAPE_SLOTS`
String array containing the number of tape devices (Data Transfer Element) available for each library. A blank must be used as a separator.
The order must be the same used for libraries with the `CHANGER_DEVICES` array.
Referring to previous sample and supposing that sg5 library has 2 tapes and sg8 has 1 tape, the array must be:
```
CHANGER_SLOTS=( 2 1 )
```

Deafult:
```
CHANGER_SLOTS=( 1 )
```

`CHANGER_TAPE_DEV_X` (where x= 0, 1... n)

Array containing the device address(es) of the tape devices associated to library X (where X is the zero-based index of the library array `CHANGER_DEVICES`).
A blank must be used as a separator.
Referring to previous sample and supposing that sg5 library has two tape drives st6 and st7 and sg8 library has a single tape st9, the arrays must appear as:

```
CHANGER_TAPE_DEV_0=( /dev/st6 /dev/st7 )
CHANGER_TAPE_DEV_1=( /dev/st9 )
```

Default
```
CHANGER_TAPE_DEV_0=( /dev/st0 )
```

`MANUAL_TAPE_DEVICES`
Is the array containing the device addresse(s) of ALL external tape device connected to the LTFSarchiver system.
A blank must be used as a separator.

Sample:
```
MANUAL_TAPE_DEVICES=( /dev/st2 /dev/st3 )
```

Default:
```
MANUAL_TAPE_DEVICES=( /dev/st1 )
```

Guessed_config file

Trying to make simpler the parameters set up in section 4 of the configuration file, it is possible to execute a script based on the *sg_map* system utility; please ensure that *sg_map* package has been installed or the script will fail.

Launching the script.

Login as root user and run: `/opt/pprime/sbin/utils/guess_conf.sh`

Each time the script finds a device that announce itself as a library or as a tape, a message will be printed. An example follows:

```
scsi device /dev/sg5 appears to be a library
scsi device /dev/st1 appears to be a tape owned by /dev/sg5 library
scsi device /dev/st0 appears to be an external tape
```

Finally, the guessed configuration found will be compared with the existing (if existing...) guessed configuration file (see `GUESSED_CONF` variable) and shown:

```
The guessed conf doesn't differ from running conf:
--------------------------------------------------
CHANGER_TAPE_DEV_0=( /dev/st1 )
CHANGER_DEVICES=( /dev/sg5 )
CHANGER_SLOTS=( 24 )
TAPE_SLOTS=( 1 )
MANUAL_TAPE_DEVICES=( /dev/st0 )
--------------------------------------------------
```

Updating the configuration

If the guessed conf file doesn't exist or differs the user is asked whether to update the configuration or not. Answering “Y”, the old `GUESSED_CONF` file will be overwritten by the new one. Please refer to Appendix A for more information and cautions about configuration guessing.

2. Starting/stopping service

Actually the real actions (e.g. read/write operations) are performed by a unique system service called *ltfsarchiver* that works by carrying out jobs queued on the database.
The service should be configured to be started at runlevels 3 and 5, using 90 as start priority (or, at least, at a lower priority than postgresql and apache daemons)

Status

To check the status of the agent, supply use the command:

```
    service lftsarchiver status
```

Possible answers are:

```
ltfsarchiver is running
ltfsarchiver is stopped
[WARNING] ltfsarchiver (RUNPID) is running, but pidfile is missing.
[WARNING] ltfsarchiver not running, but pidfile (FOUNDPID) exixts
```

RUNPID is the process id (PID) of the running instance.
FOUNDPID is the process id (PID) found into the pidfile.

Start

To start the agent, supply the command:

```
    service lftsarchiver start
```

Start will be actually attempted only if no other instance is currently running AND pidfile doesn't exists.

Possible answers are:

```
ltfsarchiver started
ltfsarchiver (RUNPID) is already running
[WARNING] ltfsarchiver not running, but pidfile (FOUNDPID) exixts
[WARNING] ltfsarchiver (RUNPID ) is running, but pidfile is missing
```

Stop

To stop the agent, supply the command:

```
      service lftsarchiver stop
```

Stop will be actually attempted only if an instance is currently running AND pidfile matches the one of the running instance.

possible answers are:

```
ltfsarchiver stopped
ltfsarchiver already stopped
[WARNING] ltfsarchiver not running, but pidfile exists... removing
[WARNING] Running process pid (RUNPID) didn't match pidfile (FOUNDPID)
[WARNING] ltfsarchiver (RUNPID ) is running, but pidfile is missing
```

Please note that stopping the agent will NOT break currently active requests, but simply prevents LTFSArchiver from trying to satisfy further requests. The requests in wait status at the moment of the switch off are not lost but executed when the service will be started again.
This means that a tape which has been put in “available online” status will be left mounted if a umount command has been issued but still not executed.

3. Log files

LTFSArchiver logs its activities using several files, all of them are placed in the directory specified with the `LTFSARCHIVER_LOGDIR` parameter.

The main logfile has the following name:

`ltfsarchiver.YYYYMMDD.log`

where `YYYYMMDD` is the date when tape_agent started.
It reports if new incoming requests are found and if they are going to be processed or not (and the reason in the last case).

When a request is processed the corresponding activity is reported in a separate log file named:

`tape_agent_ssssssssss.log`

where **`ssssssssss`** is the time when tape agent started in “seconds since 1970/01/01”

The log level is determined by the parameter `LTFSARCHIVER_LOGLEVEL`

4. Appendix A

Take care that the result of the guess_conf script may, in some cases, lead to wrong configurations.

As an example, an MSL4048 with two tape drives appears to be a tape library with a single drive, while the second internal tape will be detected as an external one.

This happens because the arm is accessible through the same FC bus assigned to the first OR the second tape device.

Here follows a real example of the `guess_config` output obtained with the guessing utility on a server with a MSL 4048 with two LTO drives (st0 and st1) and an external HP desktop LTO drive (st2).

```
CHANGER_TAPE_DEV_0=( /dev/st0 )
CHANGER_DEVICES=( /dev/sg6 )
CHANGER_SLOTS=( 48 )
TAPE_SLOTS=( 2 )
MANUAL_TAPE_DEVICES=(  /dev/st1 /dev/st2 )
```

While the right configuration file has to be changed to:

```
CHANGER_TAPE_DEV_0=( /dev/st0 /dev/st1 )
CHANGER_DEVICES=( /dev/sg6 )
CHANGER_SLOTS=( 48 )
TAPE_SLOTS=( 2 )
MANUAL_TAPE_DEVICES=( /dev/st2 )
```

Consider the guess_conf script as an aid to configure the system, not as an error proof tool.