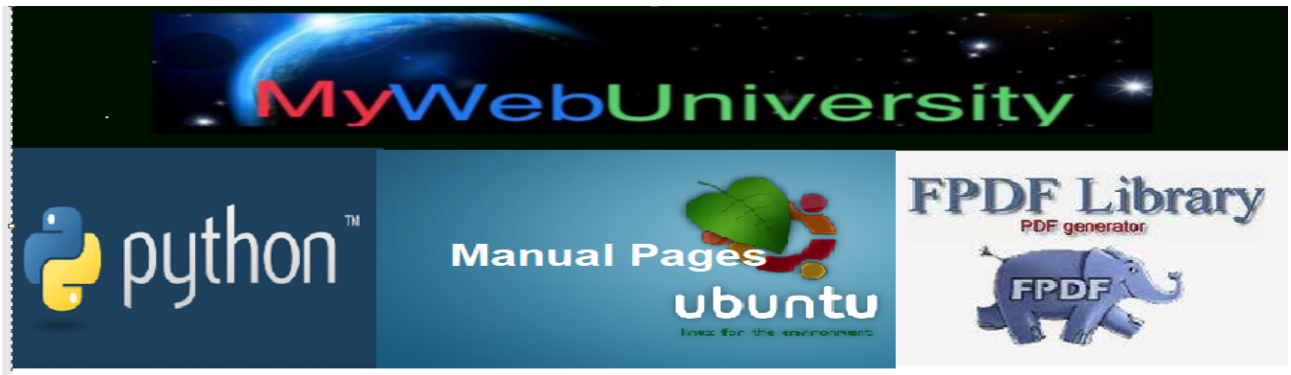

Full credit is given to the above companies including the OS that this PDF file was generated!

Linux Ubuntu 26.04 Manual Pages on command 'gprofng-archive.1'

$ man gprofng-archive.1

GPROFNG-ARCHIVE(1) User Commands GPROFNG-ARCHIVE(1)

NAME

gprofng-archive - Archive gprofng experiment data

SYNOPSIS

gprofng archive [option(s)] experiment

DESCRIPTION

Archive the associated application binaries and source files in a gprofng experiment to make it self contained and portable.

By default, the binaries are archived as part of the data collection, but the application source files are not archived. Use this tool to change this and afterwards archive additional components.

This tool has to be executed on the same system where the profiling data was recorded.

OPTIONS

--version

Print the version number and exit.

--help

Print usage information and exit.

-a {off | on | ldobjects | src | usedldobjects | used[src]}

Specify archiving of binaries and other files. In addition to disable this feature (off), or enable archiving of all loadobjects and sources (on), the other choices support a more refined selection.

All of these choices enable archiving, but the keyword controls what exactly is selected: all load objects (ldobjects), all source files (src), the loadobjects

associated with a program counter (usedldobjects), or the source files associated with a program counter (used[src]). The default is -a ldobjects.

-d path

The path is the absolute path to a common archive, which is a directory that contains archived files. If the directory does not exist, then it will be created. Files are saved in the common archive directory, and a symbolic link is created in the experiment archive.

-F Force writing, or rewriting of .archive files. All archived files will be removed and recreated, except if the -n or -m option is used, or if the experiment is a subexperiment.

-m regex

Archive only those source, object, and debug info files whose full path name matches the given POSIX compliant regex regular expression.

-n Archive the named experiment only, not any of its descendants.

-q Do not write any warnings to stderr. Warnings are incorporated into the .archive file in the experiment directory. They are shown in the output of the gprofng display text command.

-r path

This option specifies the location of a common archive. The value is the relative path to a common archive, which is a directory that contains archived files. If the directory does not exist, then it will be created. Files are saved in the common archive directory, and a symbolic link is created in the experiment archive.

-s selection

Specify archiving of source files. The allowed values for selection are:

no Do not archive any source files.

all Archive all source and object files that can be found.

used[src]

Archive source and object files for functions against which data was recorded in the experiment, and that can be found.

By default, application source files are not archived into the experiment. If the -s all, or -s used option is used, sources and object files are archived. These options also ensure that source files are available in the experiment, even if the original source files have been modified, or are inaccessible afterwards.

In case archive files cannot be found, use the addpath, or pathmap command, or both, in an .er.rc file to specify the location of the missing file(s).

NOTES

- Archiving of application binaries - By default, binaries are archived automatically when an experiment is created. However, archiving does not occur in one or more of the following circumstances:
  - ? If the profiled application is terminated before it exits normally.
  - ? If a running process is profiled.
  - ? If archiving is explicitly disabled when profiling. For example by using the -a off option on gprofng collect app.

  In these cases, gprofng archive must be run manually and on the same machine where the profiling data was recorded.

  Archiving of experiment data during the data collection process can be quite expensive. Especially if the experiment has many descendant processes. In such cases, a more efficient strategy is to use the -a off option when collecting the data. Once the collection has completed, the data can be archived using the -s all option. This saves all executables and source files in the experiment.

  If during the archiving there is an error message that an executable, or source file cannot be found, the addpath command to add the path to the missing file(s) can be included in the .er.rc file. After this command has been added, archive the experiment again. The archiving archiving can be repeated as many times as necessary to archive all files.

  Archiving should be done on the same system as was used to collect the experiment. If some files cannot be accessed from this system (e.g. sources or object files), then additional archiving can be done using another system that can access them. For example, the system where the application was built.

  Some Java applications store shared objects in jar files. By default, such shared objects are not automatically archived. To archive shared objects contained in jar files, make sure to include the addpath command in an .er.rc file. The addpath command should give the path to the jar file, including the jar file itself. The .er.rc file should be saved in the user home directory, or experiment parent directory.
- Archiving of application sources - By default, application source files are not archived in the experiment. Execute the gprofng archive command with the -s all, or -s used

option on each experiment to store source files in the experiment.

- Automatic archiving of application sources - Environment variable GPROFNG_ARCHIVE may be set to automatically archive sources when the experiment has completed. This environment variable can contain -s and -m arguments, as pairs of argument and options, separated by one or more blanks.

  If more than one -s argument appears on the command line, the last one prevails. If -s is both passed on the command line, and set by the environment variable, the option from the environment variable prevails.

  Note that in case automatic source archiving during data collection has been enabled using either the GPROFNG_ARCHIVE variable, or the -a src, or -a usedsrc option, it is recommended to confirm that source files have been correctly resolved by executing the gprofng archive -s all, or gprofng archive -s used command.
- The -d and -r options are mutually exclusive.
- When using the -d or -r option, environment variable GPROFNG_ARCHIVE_COMMON_DIR can be used to specify the location of the common archive. This can be very convenient when using a script to profile applications.
- If more than one -s option is given on the command line, or specified in the environment variable, the specified option for all must be the same. If not, gprofng archive exits with an error.
- This tool does not work on experiments recorded with earlier versions of the tools. If invoked on such experiments, a warning is printed. Use the version of gprofng archive from the same release with which the experiment was recorded.

SEE ALSO

gprofng(1), gprofng-collect-app(1), gprofng-display-html(1), gprofng-display-src(1), gprofng-display-text(1)

The user guide for gprofng is maintained as a Texinfo manual. If the info and gprofng programs are correctly installed, the command info gprofng should give access to this document.

COPYRIGHT

Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".