MyWebUniversity
python
Red Hat Enterprise Linux Manual Pages
FPDF Library PDF generator
FPDF

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'debuginfod-find.1' command

$ man debuginfod-find.1

DEBUGINFOD-FIND(1)   General Commands Manual   DEBUGINFOD-FIND(1)

NAME

debuginfod-find - request debuginfo-related data

SYNOPSIS

debuginfod-find [OPTION]... debuginfo BUILDID

debuginfod-find [OPTION]... debuginfo PATH

debuginfod-find [OPTION]... executable BUILDID

debuginfod-find [OPTION]... executable PATH

debuginfod-find [OPTION]... source BUILDID /FILENAME

debuginfod-find [OPTION]... source PATH /FILENAME

DESCRIPTION

debuginfod-find queries one or more debuginfod servers for debuginfo-related data. In case of a match, it saves the the requested file into a local cache, prints the file name to standard output, and exits with a success status of 0. In case of any error, it exits with a failure status and an error message to standard error.

The debuginfod system uses buildids to identify debuginfo-related data. These are stored as binary notes in ELF/DWARF files, and are repre? sented as lowercase hexadecimal. For example, for a program /bin/ls, look at the ELF note GNU_BUILD_ID:

```
% readelf -n /bin/ls | grep -A4 build.id
Note section [ 4] '.note.gnu.buildid' of 36 bytes at offset 0x340:
    Owner          Data size  Type
    GNU                   20  GNU_BUILD_ID
    Build ID: 8713b9c3fb8a720137a4a08b325905c7aaf8429d
```

Then the hexadecimal BUILDID is simply:

8713b9c3fb8a720137a4a08b325905c7aaf8429d

In place of the hexadecimal BUILDID, debuginfod-find also accepts a path name to to an ELF binary, from which it extracts the buildid. In this case, ensure the file name has some character other than [0-9a-f]. Files ambiguously named files like "deadbeef" can be passed with a ./deadbeef extra path component.

debuginfo BUILDID

If the given buildid is known to a server, this request will result in a binary object that contains the customary .*debug_* sections. This may be a split debuginfo file as created by strip, or it may be an original unstripped executable.

executable BUILDID

If the given buildid is known to the server, this request will result in a binary object that contains the normal executable segments. This may be a executable stripped by strip, or it may be an original un? stripped executable. ET_DYN shared libraries are considered to be a type of executable.

source BUILDID /SOURCE/FILE

If the given buildid is known to the server, this request will result in a binary object that contains the source file mentioned. The path should be absolute. Relative path names commonly appear in the DWARF file's source directory, but these paths are relative to individual compilation unit AT_comp_dir paths, and yet an executable is made up of multiple CUs. Therefore, to disambiguate, debuginfod expects source queries to prefix relative path names with the CU compilation-directo? ry, followed by a mandatory "/".

Note: the caller may or may not elide ../ or /./ or extraneous /// sorts of path components in the directory names. debuginfod accepts both forms. Specifically, debuginfod canonicalizes path names accord?

ing to RFC3986 section 5.2.4 (Remove Dot Segments), plus reducing any

// to / in the path.

For example:

```
#include <stdio.h>               source BUILDID /usr/include/stdio.h
/path/to/foo.c                   source BUILDID /path/to/foo.c
../bar/foo.c AT_comp_dir=/zoo/   source BUILDID /zoo//../bar/foo.c
```

OPTIONS

-v   Increase verbosity, including printing   frequent   download-

progress messages and printing the http response headers from

the server.

SECURITY

debuginfod-find does not include any particular security features.   It

trusts that the binaries returned by the debuginfod(s) are accurate.

Therefore, the list of servers should include only trustworthy ones.

If accessed across HTTP rather than HTTPS, the network should be trust?

worthy. Authentication information through the internal libcurl li?

brary is not currently enabled, except for the basic plaintext

http[s]://userid:password@hostname/ style. (The debuginfod server does

not perform authentication, but a front-end proxy server could.)

ENVIRONMENT VARIABLES

$TMPDIR

This environment variable points to a file system to be used for

temporary files. The default is /tmp.

$DEBUGINFOD_URLS

This environment variable contains a list of URL prefixes for

trusted debuginfod instances. Alternate URL prefixes are sepa?

rated by space.   This environment variable may be set by

/etc/profile.d scripts reading /etc/debuginfod/*.urls files.

$DEBUGINFOD_CACHE_PATH

This environment variable governs the location of the cache

where downloaded files and cache-control files are kept.   The

default directory is chosen based on other environment vari?

ables, see below.

$DEBUGINFOD_PROGRESS

This environment variable governs the default progress function.
If set, and if a progressfn is not explicitly set, then the li?
brary will configure a default progressfn. This function will
append a simple progress message periodically to stderr. The
default is no progress function output.

$DEBUGINFOD_VERBOSE

This environment variable governs the default file descriptor
for verbose output. If set, and if a verbose fd is not explic?
itly set, then the verbose output will be produced on
STDERR_FILENO.

$DEBUGINFOD_RETRY_LIMIT

This environment variable governs the default limit of retry at?
tempts. If a query failed with errno other than ENOENT, will
initiate several attempts within the limit.

$DEBUGINFOD_TIMEOUT

This environment variable governs the download commencing time?
out for each debuginfod HTTP connection. A server that fails to
provide at least 100K of data within this many seconds is
skipped. The default is 90 seconds. (Zero or negative means "no
timeout".)

$DEBUGINFOD_MAXTIME

This environment variable dictates how long the client will wait
to complete the download a file found on a server in seconds. It
is best used to ensure that a file is downloaded quickly or be
rejected. The default is 0 (infinite time).

$DEBUGINFOD_MAXSIZE

This environment variable dictates the maximum size of a file to
download in bytes. This is best used if the user would like to
ensure only small files are downloaded. A value of 0 causes no
consideration for size, and the client may attempt to download a
file of any size. The default is 0 (infinite size).

$DEBUGINFOD_HEADERS_FILE

This environment variable points to a file that supplies headers to outbound HTTP requests, one per line. The header lines shouldn't end with CRLF, unless that's the system newline con? vention. Whitespace-only lines are skipped.

CACHE

Before each query, the debuginfod client library checks for a need to clean the cache. If it's time to clean, the library traverses the cache directory and removes downloaded debuginfo-related artifacts and newly empty directories, if they have not been accessed recently.

Control files are located directly under the cache directory. They contain simple decimal numbers to set cache-related configuration pa? rameters. If the files do not exist, the client library creates the files with the default parameter values as content.

After each query, the debuginfod client library deposits newly received files into a directory & file that is named based on the build-id. A failed query is also cached by a special file. The naming convention used for these artifacts is deliberately undocumented.

$XDG_CACHE_HOME/debuginfod_client/

Default cache directory, if $XDG_CACHE_HOME is set.

$HOME/.cache/debuginfod_client/

Default cache directory, if $XDG_CACHE_HOME is not set.

$HOME/.debuginfod_client_cache/

Deprecated cache directory, used only if preexisting.

cache_clean_interval_s

This control file gives the interval between cache cleaning rounds, in seconds. The default is 86400, one day. 0 means "immediately".

max_unused_age_s

This control file sets how long unaccessed debuginfo-related files are retained, in seconds. The default is 604800, one week. 0 means "immediately".

cache_miss_s

This control file sets how long to remember a query failure, in

seconds.   New queries for the same artifacts within this time
window are short-circuited (returning an immediate failure in?
stead of sending a new query to servers). This accelerates
queries that probably would still fail. The default is 600, 10
minutes. 0 means "forget immediately".

SEE ALSO

debuginfod(8) debuginfod_find_debuginfod(3)