

debuginfod

and its interaction with the Yocto Project

Dylan Garza

June 3, 2022

debuginfod

- What is debuginfod? debuginfod is a daemon turns a machine that holds debug artifacts into file server for easier debugging.
- Why use it? Debug info is too big to be stored locally on the target devices. Different versions of binaries exist, storing and searching for debug artifacts manually is too slow.

Installing debuginfod

There are two ways to install on a Ubuntu machine:

- Upgrade to Ubuntu 22.04
- append the impish package to `/etc/apt/sources.list`

Setting up and using debuginfod

Starting the debuginfod server:

- To start the file server, simply run `debuginfod -F`. Different arguments will search for different debug artifacts (`-F -R -U`)
- Providing a path to a directory will point debuginfod where to scan for the debug artifacts.
- To set a rescan time, provide `-t [time in seconds]`

Setting up and using debuginfod (cont.)

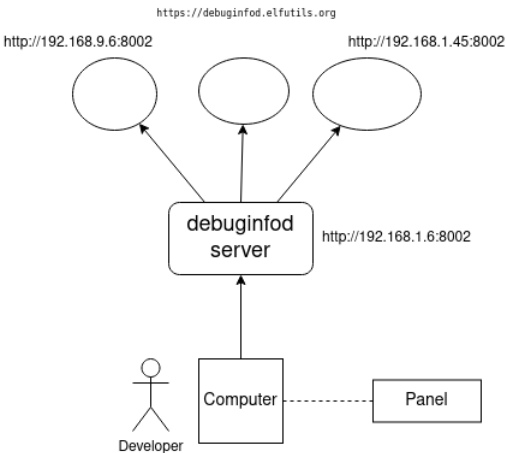
On the host machine:

- set environment variable `DEBUGINFOD_URLS` to the ip address of the file server, prefixed by `http://` and suffixed by the port number defaulted to `:8002`, which can be changed by `-p [port num]`

```
export DEBUGINFOD_URLS="http://ip_addr:8002"
```

Federating debuginfod servers

debuginfod servers can act as a host machine and query other servers
federated debuginfod servers



Federating debuginfo servers (cont.)

```
[0] ~ tmux-2
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from nvim...

This GDB supports auto-downloading debuginfo from the following URLs:
https://debuginfo.archlinux.org
Enable debuginfo for this session? (y or [n]) y
Debuginfo has been enabled.
To make this setting permanent, add 'set debuginfo enabled on' to .gdbinit.
Downloading 14.73 MB separate debug info for /home/dylandy/nvim
Reading symbols from /home/dylandy/.cache/debuginfod_client/2e49f4960123ce02457f5ad3fc668c518c32849/debuginfo...
(gdb) List
Downloading 0.00 MB source file /usr/src/debug/neovim-0.7.0/src/win/main.c
200 set_lang_var(); // set vimlang and v:ctype
201
202 init_signs();
203 ui_comp_syn_init();
204
205
206 #ifdef MAKE_LIB
207 int nvim_main(int argc, char **argv); // silence -Wmissing-prototypes
208 int nvim_main(int argc, char **argv)
209 #elif defined WIN32
210 (gdb)

X Error of failed request: BadName (named color or font does not exist)
Major opcode of failed request: 140 (RANDR)
Minor opcode of failed request: 16 (RRCreateMode)
Serial number of failed request: 35
Current serial number in output stream: 35
dylandy@super-desktop
os Artix Linux
host 192.168.1.6
kernel 5.17.6-artix1-1
uptime 15d 22h 15m
pkgs 805
memory 562M / 15689M

dylandy@super-desktop ~$ export DEBINFO_URLS="https://192.168.1.2:8002"
dylandy@super-desktop ~$ rm -rf .cache/debuginfod_client
dylandy@super-desktop ~$

[0] ~ gdb*
"super-desktop" 09:53 03-Jun-22

[0] ~ bash*
dylandy@super-compute ~$
Can't open display ''
dylandy@super-compute
os Artix Linux
host 17295N-G.AASBU1 0.1
kernel 5.17.6-artix1-1
uptime 15d 22h 15m
pkgs 805
memory 562M / 15689M

Couldn't get a file descriptor referring to the console.
-bash: [: : integer expression expected
dylandy@super-compute ~$ export DEBINFO_URLS="https://debuginfod.elfutils.org"
dylandy@super-compute ~$ ls ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: wlan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 1000
    link/ether c4:23:6d:a0:11:brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.2/24 brd 192.168.1.255 scope global dynamic noprefixroute wlan0
        valid_lft 86214sec preferred_lft 86214sec
    inet6 fe80:b86d:fbe:2023:e27f:64 scope Link noprefixroute
        valid_lft forever preferred_lft forever
dylandy@super-compute ~$ ls
```

Demo

debuginfod with the Yocto Project

For versions after 3.4(honister) only 2 changes are needed for debuginfod to work on a build. Append the following the `local.conf`:

1

```
PACKAGE_CONFIG_pn-elfutils-native="debuginfod libdebuginfod
```

2

```
DISTROFEATURES += "debuginfod"
```

Yocto Project has built in script for debuginfod to eliminate compatability issues.

debuginfod-find

Command apart of elfutils, it performs debuginfod server querying without gdb. Can query for debuginfo, executables, and sources.

```
debuginfod-find [debuginfo/executable/source] [path/to/file or Build-ID] [path/to/source]
```

If the path to the source is not known, but debug info is available, use the `list` command under `gdb` to provide source path.

Demo