

elfutils debuginfo-server

necessary non-evil

Mark Wielaard, Frank Ch. Eigler

Red Hat

`mark@klomp.org fche@redhat.com`

FOSDEM 2020-02-02

elfutils debuginfod is a web fileserver for debugging artifacts

What's the matter? Bugs got into your code?

No problem!

- Install prerequisites for debugging
this is our topic today
- Use debugger
not talking about this part today
- But I use python or node.js or ...
not talking to you today
- But I use go lang or rust or ...
relevant but not my focus today
- Er, what's a debugger?
there are many other debuginfo tools: crash or systemtap or
perf or dyninst or abrt or abigail or hpctoolkit ... and proceed

congratulations, you need debuginfo!

- Compilers generate metadata about the object code.
CFLAGS+=-g
- Maps between source and object code for text and data
- Quality, quantity differ by compiler and by optimization level
- GCC + DWARF is world-class
(-fvar-tracking-assignments)
- Tools can compress (semantic DWZ or zlib ELF compression)
- Subset format also exist (CTF, BTF, minidebuginfo, STABS)
- Don't forget about source code! (might be partially generated)

debuginfo, where is it now?

- What if debugging your own build tree? → a-ok
- What if “make” stripped your binaries? → sad trombone
- What if you’re using someone else’s build? → depends
- What if you’re debugging your distro? → depends
- What if you’re debugging a container? → depends
- What if you’re debugging remotely? → depends

We’ll try to turn those “depends” into “a-ok”!

why is it not everywhere?

- SIZE: 5-15x the size of the stripped executable

- Fedora 30 x86-64:

program	stripped	unstripped
linux 5.2.11	60,309,296	812,269,008
cc1plus 9.2.1	28,632,168	176,320,528 + 29,579,791dwz

- Don't forget about source code!

- More stats:

distro	main repo	debug repo
rhel 7.6 x86-64	3.4 GB	11 GB
rhel 8.0 baseos x86-64	0.9 GB	2.5 GB

where is debuginfo - fedora

- stripped after the build process
- not lost: packaged into `-debuginfo` and `-debugsource` RPMs

```
% gdb /bin/vi
Reading symbols from /bin/vi...
(No debugging symbols found in /bin/vi)
Missing separate debuginfos, use:
dnf debuginfo-install vim-minimal-8.1.1912-1.fc30.x86_64
```

- available for easy downloading
- ... if you're root, if you have disk space for whole package, if if if
- <https://fedoraproject.org/wiki/StackTraces>

where is debuginfo - ubuntu

- stripped after the build process
- not lost: packaged into `-dbgsym` and `-dbg` ddebs

```
% gdb /bin/vi
Reading symbols from /bin/vi...
(No debugging symbols found in /bin/vi)
```

- available for downloading - not as easy
- ... if you're root, if you guess the right debsources, if you have disk space for whole package, if if if
- <https://wiki.ubuntu.com/DebuggingProgramCrash>

where is debuginfo - arch linux

- never created during the build process
- KKKHHHAAAAANNNNN!!!!
- recompile with custom CFLAGS
- ... if you're root – heck you're always root on arch
- https://wiki.archlinux.org/index.php/Debug_-_Getting_Traces

- Like Fedora, many packages build debuginfo subpackages
- Optional FUSE server maps local debuginfo paths to NixOS CDN via HTTP.
- `https://github.com/edolstra/dwarffs`
- So close!

addressing the depends and what-ifs

- Need a way of quickly delivering the debuginfo needed.
- ... without user privilege
- ... even from private non-distro build trees
- ... even from non-distro package archives
- ... ideally without unneeded debuginfo
- ... able to federate servers
- Don't forget about source code!

- A simple HTTP fileserver of debuginfo to debugger-like tools
- Server released as a component of elfutils
- Numerous clients done or underway
- Indexed by buildid
- Trivial webapi:
`http://server:port/buildid/HEXCODE/debuginfo`
- Doesn't forget about source code!
`http://server:port/buildid/HEXCODE/source/PATH/TO/FOO.c`
- No privilege required for running service

what's a buildid?

- Unique hash code embedded into object files as ELF note.

```
% readelf -n /bin/vi | grep -A4 build-id
Displaying notes found in: .note.gnu.build-id
Owner   Data size   Description
GNU     0x00000014  NT_GNU_BUILD_ID (unique build ID)
Build ID: d153e961b07a044d66e523f03e00e7615ab56c4d
```

- Default-on in GCC/toolchain for 10 years
(`--enable-linker-build-id`)
- <https://fedoraproject.org/wiki/Releases/FeatureBuildId>
- Compatibly supported by llvm, partly by golang
- Identifies unique builds (reproducible build → same build-id)
- Works best when using `-g` (captures environment/flags/sources)
- Can be used to match separated debuginfo (build-id in both main and debug file)

how to use server

```
% debuginfod -R -F /var/tmp/rpmbuild /usr/lib/debug
[... ] Opened database /$HOME/.debuginfod.sqlite
[... ] started http server on IPv4 IPv6 port=8002
[... ] search concurrency 8
[... ]           file d/e 68
[... ]           file s 3019
[... ]           archive d/e 23
[... ]           archive sref 48
[... ]           archive sdef 2514
[... ]           buildids 83
[... ]           filenames 7835
[... ]   files scanned (#) 1752
[... ]   files scanned (mb) 269
[... ]   index db size (mb) 1
```

... or systemd service

... or container

how to use client

```
% export DEBUGINFOD_URLS="http://buildhost:8002/"  
% gdb $anything  
or  
% debuginfod-find source hexcode /path/to/foo.c
```

... that's it!

insert demo here

how debuginfo-server works

- Given some directory names, build trees or RPM/DEB archives
- Periodically rescan all contents, extract buildids
- Only stream-process RPMs/DEBs, don't store contents
- Locate any referenced source files (not easy!)
- Store in persistent sqlite database, groom periodically
- To service a query, stream data based on buildid record

- Indexes from buildid to filenames or (package,content) tuples
- Supports DWZ / altdebug compression
- Indexes source code references
- Also stores file mtime to validate cache
- Normalizes representation so strings not duplicated
- Grooming involves compaction, diagnosing duplication (maybe hostile!), garbage collection for removed/updated files
- In principle transportable, mergeable across cluster
- p.s. SQLite is great, use it for most of your database needs

how debuginfo-server client works

- Given buildid and artifact type (debuginfo/executable/source)
- Given one or more server URLs in `$DEBUGINFOD_URLS`
- Performs one or more HTTP queries, until timeout or conclusion
- Caches resulting file in local cache directory
- Returns file name and/or descriptor to caller
- Available as a library and a command line tool
- Also built into server, ergo federation

- Databases can be large, $\frac{1}{100}$ of RPM size
- Indexing scan can be slow, $10 \frac{MB}{s}$
- Fedora koji build system: 77 TB of RPMs, do the math
- Good news: embarrassingly parallel problem
- Can merge databases after indexing
- Can configure each server to delegate to others
- Natural lines: personal \rightarrow team, frontend \rightarrow replicas, local \rightarrow remote, private \rightarrow public, shard \rightarrow complete
- Directed acyclic delegation graph

I want it all, and I want it now

- Client cli & library and server released with elfutils 0.178
- Elfutils-based tools automatically take advantage (systemtap, dyninst, eu-*)
- Prototype gdb client ... on a branch, RFC posted
- Work for other clients under way
- Some public servers already available!

ELFUTILS DEBUGINFOD

elfutils debuginfod is a client/server in elfutils 0.178+ that automatically distributes elf/dwarf/source-code from servers to clients such as debuggers across HTTP.

servers

Anyone may run debuginfod servers for their private purposes. Organizations are invited to run public servers for use by the community. To configure a tool to use them, simply add any of the server URLs to your **\$DEBUGINFOD_URLS** environment variable (space-separated). The following public services currently [2020-01] welcome users.

server	status	operator	distro	packages	architectures
https://debuginfod.systemtap.org/	experimental	fche@redhat.com	recent fedora, centos, ubuntu, debian	kernel, glibc, httpd, systemtap, binutils, elfutils, coreutils	mostly x86_64
https://debuginfod.opensuse.org/	experimental	mliska@suse.cz	opensuse TW	all	x86_64

--

clients

debuginfod client-side support is under construction or already available in a variety of binary-related utilities. We summarize current upstream status [2020-01] below. Note that distros may lag behind upstream developments.

tool	status
elfutils	released in version 0.178, 2019-11
systemtap	automatic via elfutils
dwarves	automatic via elfutils
binutils	merged , forthcoming in version 2.34
gdb	proposed
dyninst	in progress amerey@redhat.com
annocheck	in progress amerey@redhat.com
libabigail	in progress amerey@redhat.com
delve	in progress amerey@redhat.com
lldb	help wanted
perf	help wanted

more readings

- [debuginfod\(8\) man page](#)

- Client support in all the tools
- Improve server security posture
- Continue improving gcc debuginfo quality
- Run a server for your distro or ISV binaries
- Investigate extending webapi for DWARF content queries, to offload search computation

- <https://sourceware.org/elfutils/Debuginfod.html>
- #elfutils ON irc.freenode.net
- <https://dwarf.org/>
- <https://submission.fosdem.org/feedback/10308.php>