Online Debugging and ABI Data Services

FOSDEM 2024-02-03

Frank Ch. Eigler <fche@redhat.com>

Compile, Distribute, Run

- Source code goes in
- Binaries come out
- Binaries go into distros
- Users run binaries

What about debugging?

- When binaries fail, you need debuggers.
- Debuggers examine machine level state.
- This needs metadata: DWARF from compiler.
- Also used by: tracing, profiling, crash-dump systems.
- Very old way: only developers debug based on own binaries.
- Old way: manual install stripped debug packages as root.
- New way (2019+): automated via debuginfod

Online services for debuginfo

- For linux: https://debuginfod.elfutils.org/.
- Gets debuginfo **and source code** on the fly.
- User chooses favorite servers, and may operate own server.
- All major debugging-type tools in linux world are clients.
- Fedora/CentOS/SuSE, Debian/Ubuntu, Arch and more operate servers.
- Coming soon: crypto integrity checking, metadata, srcfiles
- (Some other operating systems and distros have similar solutions.)

debuginfod try this at home

- `% export DEBUGINFOD_URLS=https://debuginfod.elfutils.org/`
- `% eu-stack -v -p $$    # elfutils`
- `% gdb -args /bin/ls`

  `(gdb) break main`

  `(gdb) run`

What about compatibility checking?

- Tricky to build binaries for more than one distro.
- Even building for different version of same OS is tricky.
- Why? Because APIs and ABIs change.
- API: compile-time problem, easier.
- ABI: run-time compatibility, tougher.
- Bundling solutions: flatpak, containers, static linking, etc.
- Can check for ABI changes instead.

What is an ABI?

- Makes shared libraries possible!
- Assembler level interoperability rules.
- The signature of every function entry point.
- Exact memory layout of global variables & types.
- … and their transitive closure.
- Sometimes compiler / version dependent.
- Same information as in the debuginfo!

How to check compatibility?

- Roll the dice and run it? Examine header files?
- Or use analysis tool: https://sourceware.org/libabigail
- Can compare different shared library versions.
- Or compare needs of a binary to any shared library.
- Needs debuginfo of some sort, and works with debuginfod!
- Can generate and use an XML representation of ABI as proxy.
- Still large, but not as large as debuginfo.

Collect it

- ABI XML is textual!
- Text is easy to share centrally, compressible, use git!
- Prototyping a little tool “abidb”, fitting into libabigail.
- Stores abi xml grouped by SONAME into per-distro branches.
- Can generate and submit abi xml documents into a git repo.
- Can take RPMs, DEBs, or just plain binaries.
- Can represent multiple updated versions of same library.
- Can query a binary against any relevant abi xml in git.

Abidb try this at home

- % git clone https://sourceware.org/git/libabigail.git
- % [... build/install from users/fche/abidb branch ...]
- % git clone https://sourceware.org/git/abidb.git # 9 GB
- % cd abidb
- % abidb --check $random_binary
- % abidb --distrobranch ubuntu/20.04/x86_64 --check $random_binary
- % abidb --submit /usr/local/libfoo.so
- % find -name .rpm | xargs abidb -Zrpm \
  --distrobranch foo/bar --submit
- % git push

Online services

- debuginfod: https://debuginfod.elfutils.org/
- abidb corpus: https://sourceware.org/git/abidb.git
- Low complexity, low tech-novelty services you can run for yourself.
- Can also use or federate to public servers.

- Simple tech helps solve the debuginfo and abi metadata distribution problems.