Google

Seamless debugging of emulated applications with LLDB

Pavel Labath

GEMU

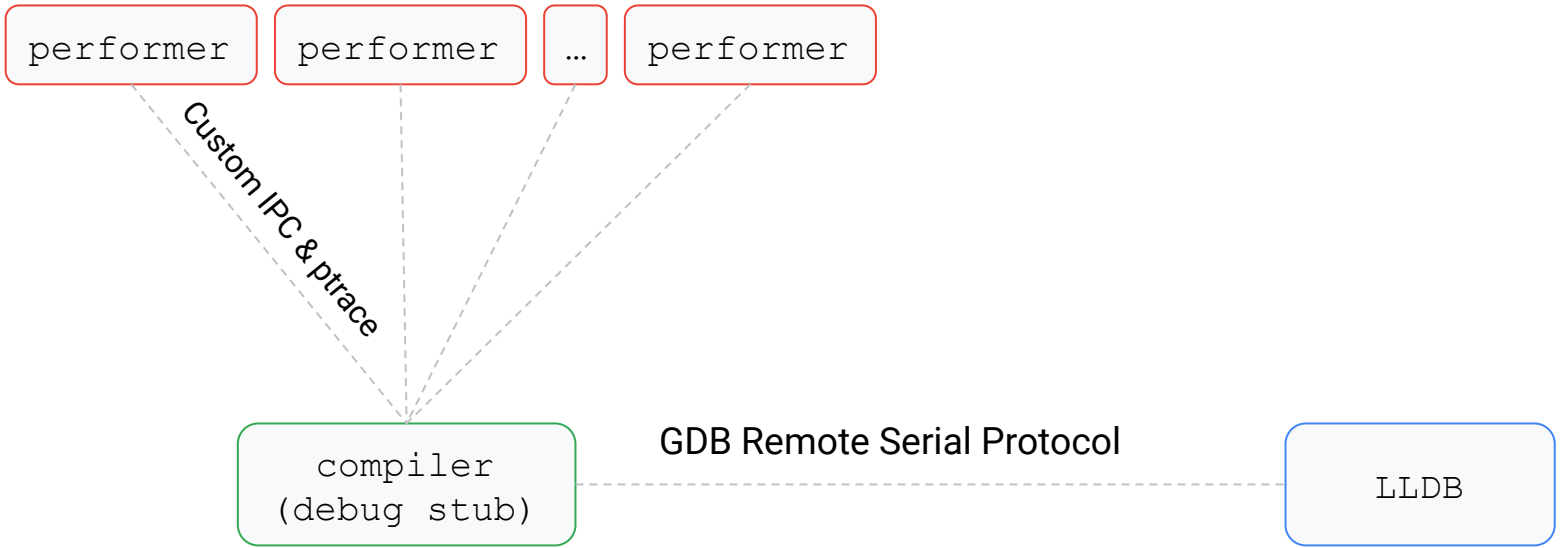
Proprietary AArch64 (linux) user space emulator, with a focus on accurate emulation and first class debugging support.

Source: Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis non erat sem

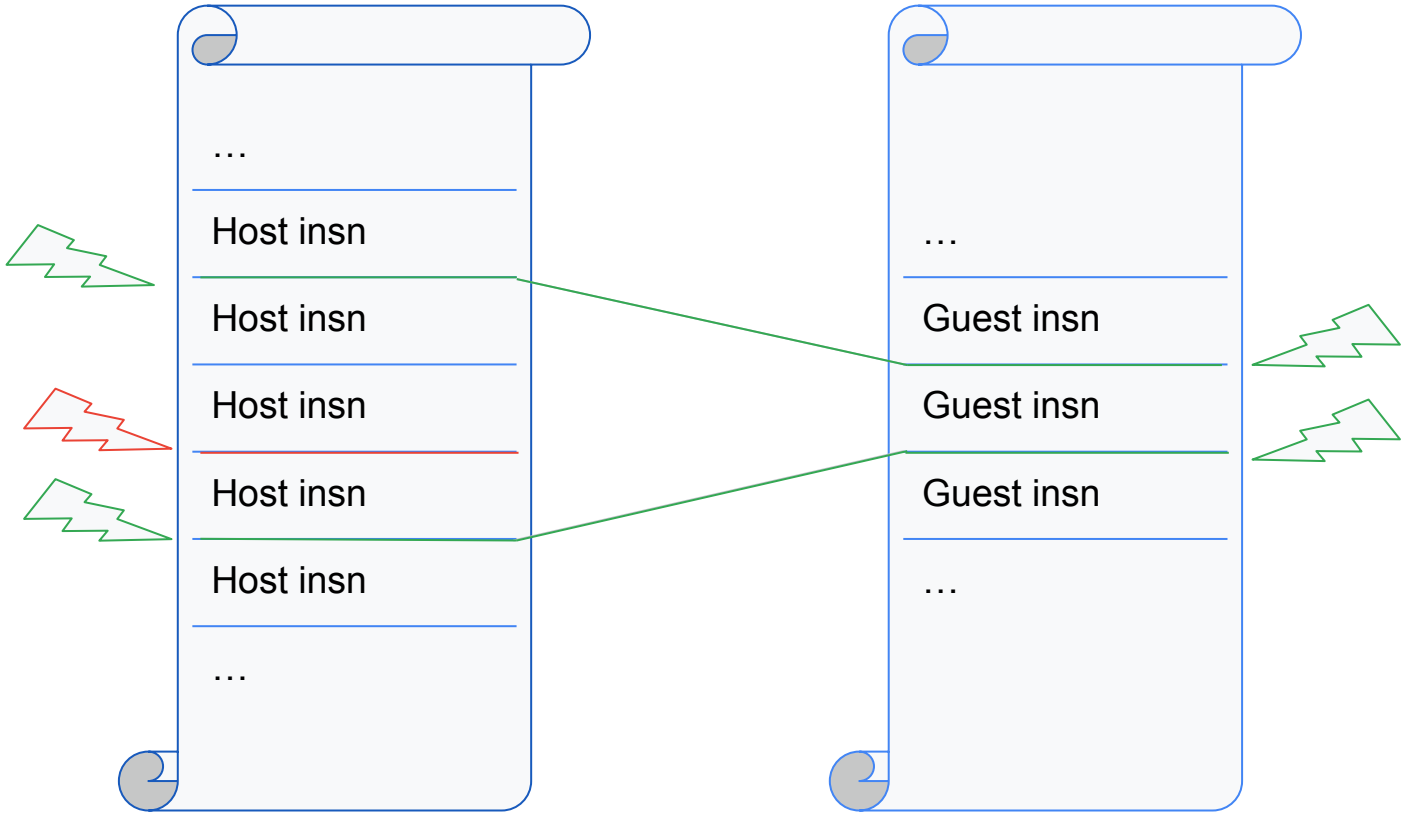
GEMU vs. QEMU

Feature	GEMU	QEMU
Performance	Comparable	Comparable
Multithreaded debugging	Yes	Crashes
Expression evaluation (w/LLDB)	Yes	Almost works
Attach	Yes	No
Other host/target combinations	No	Yes
Full system emulation	No	Yes

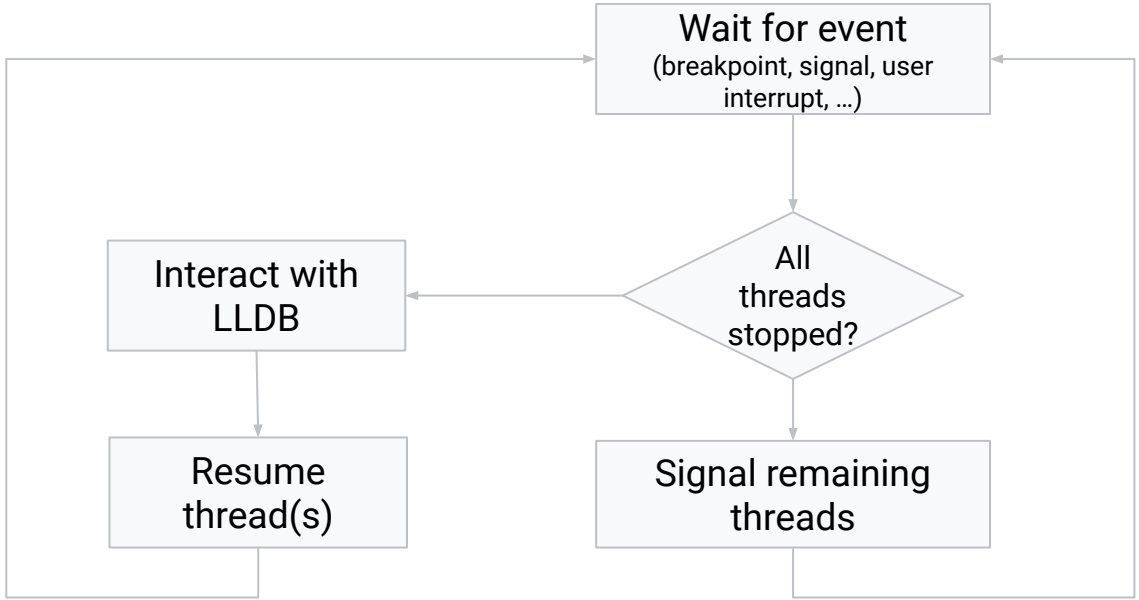
GEMU Architecture



GEMU Architecture: Signal handling



GEMU debug stub operation



Source: Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis non erat sem

LLDB Integration

Initial state

```
term1$ gemu -gemu_args… …/binary -binary_args…

…

term2$ lldb -O “platform select remote-linux” …/binary -o “settings set

target.exec-search-paths …” -o … -o “process connect …”

(lldb) …
```

Solution: Platform plugin

A plug-in interface definition class for debug platform that includes many platform abilities such as:

- *getting platform information such as supported architectures, supported binary file formats and more*
- *launching new processes*
- *attaching to existing processes*
- *download/upload files*
- *execute shell commands*
- *listing and getting info for existing processes*
- *attaching and possibly debugging the platform's kernel*

Source: lldb/include/lldb/Target/Platform.h, see also: lldb/source/Plugins/Platform/QemuUser/PlatformQemuUser.h

Solution: Platform plugin

```
std::vector<ArchSpec>
PlatformGemuUser::GetSupportedArchitectures( ...) {
  return {ArchSpec("aarch64-unknown-linux-gnu");};
}
```

Launch

```
PlatformGemuUser::DebugProcess(ProcessLaunchInfo &launch_info, …) {
  …
  launch_info.SetArguments(adjust(launch_info.GetArguments(),  true);
  …
  error = Host::LaunchProcess(launch_info);
  …
  error = process_sp->ConnectRemote(socket_path);
}
```

Attach

`bool PlatformGemuUser::GetProcessInfo(lldb::pid_t pid, …)`

- Combine emulator-provided info from `unix-abstract://gemu/processinfo/$PID`, with `Host::GetProcessInfo(pid)`.

`lldb::ProcessSP PlatformGemuUser::Attach(ProcessAttachInfo &attach_info,...)`

- `process_sp->ConnectRemote( "unix-abstract://gemu/processes/$PID" )`
- Read additional data (library paths) from `unix-abstract://gemu/processinfo/$PID`

Make platform default (optional)

```
# Invoked from lldbinit

# Currently, there is no way to associate a platform with a debugger without
# selecting it, so set our platform as selected, and then immediately go back
# to the old one.
old_platform = debugger.GetSelectedPlatform()
debugger.SetSelectedPlatform(lldb.SBPlatform( "gemu-user"))
debugger.SetSelectedPlatform(old_platform)
```

Thank you