glvnd Status Report

XDC 2016

Kyle Brenneman, Andy Ritger

NVIDIA

glvnd Overview

Defines a new ABI between OpenGL applications, libraries, and implementations:

- Window System libraries: EGL, GLX.
- Client API libraries: OpenGL, OpenGL ES.
- Allows multiple vendor implementations to co-exist on the file system.
- Allows multiple vendor implementations to co-exist in the same process.

glvnd Overview, continued

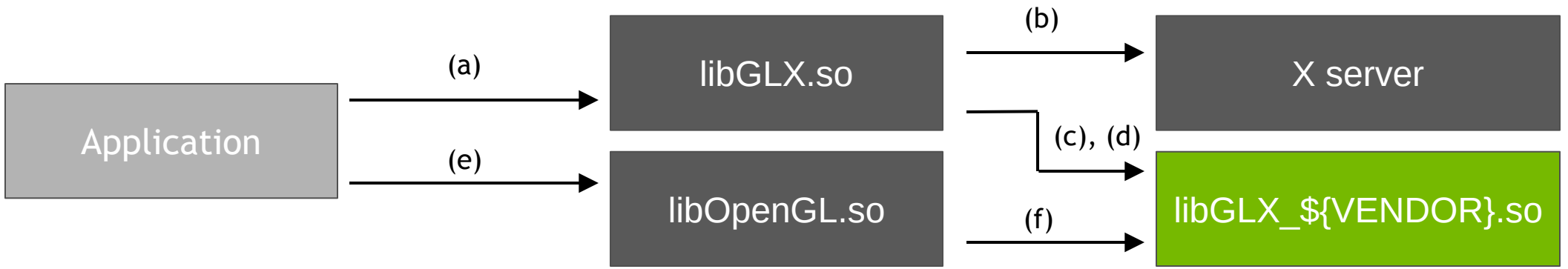
Vendor-neutral libraries (provided by glvnd):

- libGLX.so
- libEGL.so
- libOpenGL.so
- libGLESv1_CM.so
- libGLESv2.so
- libGL.so

Vendor-specific libraries (provided by each vendor):

- libGLX_${VENDOR}.so
- libEGL_${VENDOR}.so

glvnd Example: libGLX.so and libOpenGL.so



a) Application calls any GLX entry point.

b) libGLX.so queries GLX_EXT_libglvnd in X server, to map X screen to vendor.

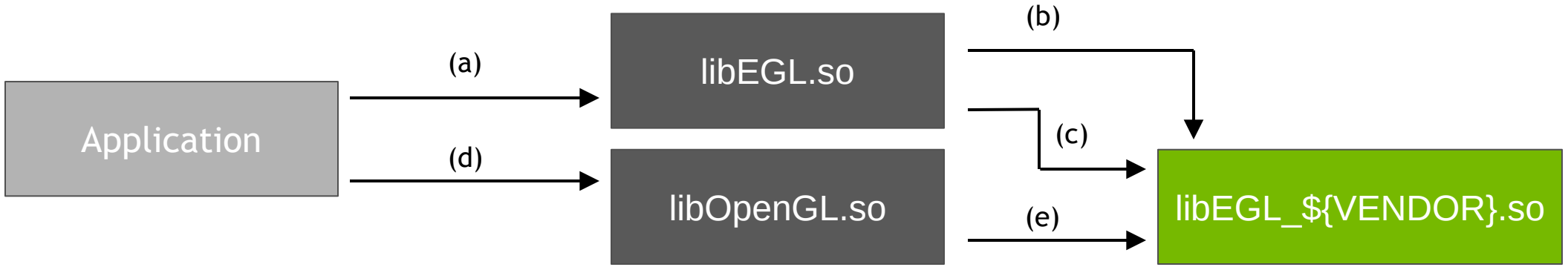
c) libGLX.so loads and dispatches to libGLX_${VENDOR}.so.

d) During glxMakeCurrent, libGLX.so calls libGLX_${VENDOR}.so to construct dispatch table used by libOpenGL.so.

e) Application calls OpenGL entry point.

f) libOpenGL.so.1 jumps through dispatch table to OpenGL implementation registered by libGLX_${VENDOR}.so.

glvnd Example: libEGL.so and libOpenGL.so



a) Application calls eglGetPlatformDisplay() or eglGetDisplay().

b) libEGL.so loads all libEGL_${VENDOR}.so's, asking which recognizes the display_id. The resulting EGLDisplay is associated with that vendor.

c) During eglMakeCurrent, libEGL.so calls libEGL_${VENDOR}.so to construct dispatch table used by libOpenGL.so.

d) Application calls OpenGL entry point.

e) libOpenGL.so.1 jumps through dispatch table to OpenGL implementation registered by libEGL_${VENDOR}.so.

glvnd Status: API Support

- libGLX.so: exports GLX 1.4 entry points, plus glXGetProcAddress.
- libEGL.so: exports EGL 1.5 entry points, plus eglGetProcAddress.
- libOpenGL.so: exports OpenGL 4.5 entry points.
- libGLESv1_CM.so: exports OpenGLES CM 1.0 entry points.
- libGLESv2.so: exports OpenGLES 2.0, 3.0, 3.1, and 3.2 entry points.
- libGL.so: exports every entry point we could find in any vendor's libGL.so, for backwards compatibility.

Vendors can advertise additional entry points and new API versions at run time.

glvnd Status: Platform Support

NVIDIA is currently using glvnd on Linux x86, x86_64 and ARMv7.

aarch64 is almost done.

ppc64le is in progress.

Needs porting to other UNIX platforms and CPU architectures:

- Porting to other UNIX platforms should be straight-forward (requires pthreads, currently uses dlsym(RTLD_DEFAULT, …))
- Porting to new CPU architectures requires online generated assembly for dispatch stubs.

glvnd Status: NVIDIA

The NVIDIA driver supports using glvnd.

- Choose at install time whether to use glvnd.
- Use either distro's glvnd or glvnd snapshot in NVIDIA driver package.

NVIDIA Driver Release status:

361.45.11: initially shipped glvnd OpenGL and GLX, disabled by default.

364.19: glvnd OpenGL and GLX, enabled by default.

375.xx (future): glvnd EGL, disabled by default.

378.xx (future): glvnd EGL, enabled by default.

glvnd Status: Mesa

Mesa has optional support for using glvnd.

- OpenGL: support enabled at build-time (disabled by default) in Mesa 12.0.0.
- GLX: support enabled at build-time (disabled by default) in Mesa 12.0.0.
- EGL: support enabled at build-time (disabled by default) patches in review.

Use ‘--enable-libglvnd’ when building Mesa.

glvnd Implementation: libGLdispatch.so

- Handles all dispatching of OpenGL functions within glvnd.
- Applications don't link against this directly.
  - Instead, application-facing libGL.so, libGLES*.so, libOpenGL.so use libGLdispatch.so.
- libGLdispatch.so dispatch stubs:
  - Based on GLAPI from Mesa.
  - Stubs look up function pointers from dispatch table in TLS.
  - Known entry points get stubs at compile-time.
  - Future entry points (e.g., new GL extensions) get run-time generated stubs.

glvnd Implementation: libGLX.so

- New in ecosystem, added by glvnd.
- Only exports GLX entry points (unlike libGL.so).
- Dispatches to vendor based on function arguments.
  - For glvnd-known GLX entry points: glvnd interprets function arguments.
  - For glvnd-unknown entry points: vendor libraries provide dispatch stubs for each GLX extension function.

glvnd Implementation: libGL.so

- Provided for backward compatibility.
- Is a wrapper over libGLX.so and libGLdispatch.so.
- Exports all symbols we could find exported by any libGL.so.

glvnd Implementation: GLX vendor selection

- New GLX extension, GLX_EXT_libglvnd, lets glvnd query what vendor to use per X screen.
- For local X server:
  - Returns a valid vendor name (e.g., "nvidia", or "mesa").
  - This is used to select vendor libraries such as "libGLX_nvidia.so" or "libGLX_mesa.so".
- For remote X server:
  - The server might not provide GLX_EXT_libglvnd.
  - The server-reported vendor may not exist on the client.
  - glvnd needs to fall back to vendor-neutral indirect rendering client, libGLX_indirect.so.
  - For now, libGLX_indirect.so is just a symlink to another vendor library, but should be glvnd-provided (selection of the symlink is arbitrary).
- Cannot really have screen granularity until we have server-side glvnd.

glvnd Implementation: libEGL.so

- Exports EGL entry points.
- Dispatches to vendor EGL implementation, based on EGL function arguments.
- Unlike GLX, loads all EGL vendor implementations up front.
  - Needed for EGL_EXT_device_enumeration.
  - Requires vendor EGL implementations to register at install time.
  - Modeled after Vulkan ICD.

glvnd Implementation: EGL vendor selection

Most EGL entry points take an EGLDisplay.

We associate a vendor with each EGLDisplay.

For:

```
EGLDisplay eglGetPlatformDisplay(EGLenum platform, void *native_display, ...)
```

call each vendor EGL library until one succeeds.

For:

```
EGLDisplay eglGetDisplay(EGLNativeDisplayType display_id)
```

guess platform enum from display_id, then behave same as eglGetPlatformDisplay().

glvnd Implementation: EGL client extensions

- When adding new EGL client extensions (ones that are independent of EGLDisplay), glvnd will need to be updated.
  - Thankfully, EGL client extensions are rare.
- EGL_KHR_debug must be implemented by every EGL vendor.
  - EGL_KHR_debug lets the application register a callback function
  - Callback function must be called any time an EGL error is generated, on any EGLDisplay.
  - No good way to handle this per-vendor, so just require it from everyone.

glvnd: Consistency of Function Pointers

An OpenGL entry point can be looked up through any of:

- -lGL and call glFoo
- -lOpenGL and call glFoo
- -lGLES* and call glFoo
- {glX,egl}GetProcAddress("glFoo")
- dlsym("glFoo")

All route to the same function.

GLX and EGL can be used in same process.

glvnd: Exposing Application Bugs

glvnd exposed a variety of application bugs:

- Applications called OpenGL entry points without a current context.
  - glvnd-based OpenGL would crash: NULL pointer to dispatch table.
  - NVIDIA's non-glvnd OpenGL initialized TLS to point to noop dispatch table (on first GLX function call).
  - Resolution: glvnd initializes a noop dispatch table on first GLX function call.
- glvnd dispatch table patching was expensive, slowing down cases where applications called glXMakeCurrent frequently.
  - Resolution: Avoid dispatch table patching when not necessary (only need to patch when switching vendors).
- We haven't seen application bugs caused by changing the set of exposed symbols (well, because libGL.so exports everything…)

glvnd Status: What Next?

- More testing with glvnd+Mesa.
  - What else does Mesa need from glvnd before glvnd could be enabled by default?
- Get glvnd EGL shipping and used by both Mesa and NVIDIA.
- Encourage other OpenGL vendors to port to glvnd.
- Once Mesa is ready to enable glvnd support by default, distros can start packaging glvnd and shipping glvnd-enabled Mesa.
- Server-side GLX glvnd.

glvnd: Call to Action for OpenGL Vendors

Make your OpenGL+GLX implementation glvnd-aware:

- Provide libGLX_${VENDOR}.so
- Implement glvnd interface defined in glvnd/libglxabi.h

Make your OpenGL+EGL implementation glvnd-aware:

- Provide libEGL_${VENDOR}.so
- Implement glvnd interface defined in glvnd/libeglabi.h
- Implement these EGL extensions:
  - EGL_KHR_client_get_all_proc_addresses
  - EGL_EXT_platform_base
  - EGL_KHR_debug

Mesa patches are an example of how to do this.

glvnd: Call to Action for Linux Distributors

- Once Mesa is ready, package glvnd and glvnd-enabled Mesa.
- If you package the NVIDIA driver, use the glvnd-enabled libraries.

glvnd: Call to Action for OpenGL Developers

- Test glvnd + glvnd-enabled Mesa and/or NVIDIA.
- As glvnd propagates through the ecosystem, start using libGLX + libOpenGL instead of libGL.
- Only call OpenGL entry points while a context is current.
- Use __GLVND_APP_ERROR_CHECKING=1 environment variable during development.
  - So far, only reports OpenGL entry points called without a context.
  - Hopefully add more checking here.

glvnd: Thank You

- Thank you to Kyle Brenneman (kbrenneman ‘at’ nvidia.com) for all the work on glvnd.

- https://github.com/NVIDIA/libglvnd

- https://devtalk.nvidia.com/default/topic/915640/unix-graphics-announcements-and-news/multiple-glx-client-libraries-in-the-nvidia-linux-driver-installer-package/