Status Report on A New Linux OpenGL ABI

Andy Ritger, NVIDIA Corporation

October, 2014

Overview

At XDC 2013: presented a proposal for a new Linux OpenGL ABI

- Define new ABI between applications and OpenGL libraries:
  - Window System libraries: EGL, GLX
  - Client API libraries: OpenGL, OpenGL ES
- Allow multiple vendor implementations to co-exist on the file system.
- Allow multiple vendor implementations to co-exist within the same process.
- Focus is on Linux, but should be applicable to other UNIX and UNIX-like platforms.

In today's talk:

- Restate the proposal.
- Describe what work has been done, and what work is left.

New Library Organization: Summary

- Vendor-neutral Client API Libraries:
  - libOpenGL.so.1
  - libGLESv1_CM.so.1
  - libGLESv2.so.2
- Vendor-neutral window system libraries:
  - libGLX.so.1
  - libEGL.so.1
- Vendor-specific libraries:
  - libGLX_${VENDOR}.so.1
  - libEGL_${VENDOR}.so.1

New Library Organization: Vendor-Neutral Client API Libraries

- libOpenGL.so.1
  - Provides symbols for OpenGL 4.4 (Core and Compatibility Profiles).
  - Vendors can provide additional OpenGL entry points, through {egl,glX}GetProcAddress.
  - No EGL or GLX entry points provided by this library; should be used with lib{GLX,EGL}.so.1.
- libGLESv1_CM.so.1
  - Provides symbols for all OpenGL ES 1 common profile entry points.
  - No EGL or GLX entry points provided by this library; should be used with lib{GLX,EGL}.so.1.
- libGLESv2.so.2
  - Provides symbols for all OpenGL ES 2 and 3 entry points.
  - No EGL or GLX entry points provided by this library; should be used with lib{GLX,EGL}.so.1.

New Library Organization: Vendor-Neutral Window System Libraries

- libEGL.so.1
  - Provides symbols for all EGL 1.4 entry points.
  - Loads and dispatches to one or more vendor libraries.
- libGLX.so.1
  - Provides symbols for all GLX 1.4 entry points.
  - Provides symbols for the GLX_ARB_create_context extension.
  - Loads and dispatches to one or more vendor libraries.

New Library Organization: Vendor-Specific Libraries

- lib{EGL,GLX}_${VENDOR}.so.1
  - Provides initialization function that lib{EGL,GLX}.so.1 calls.
  - Pulls in the vendor's implementation of all the client APIs it supports.
  - Registers with the appropriate Client API library at MakeCurrent time.
  - Must not export symbol names that collide with:
    - EGL (^egl.*),
    - GLX (^glX.*), or
    - OpenGL (^gl.*).

Dispatching to Vendor Implementations

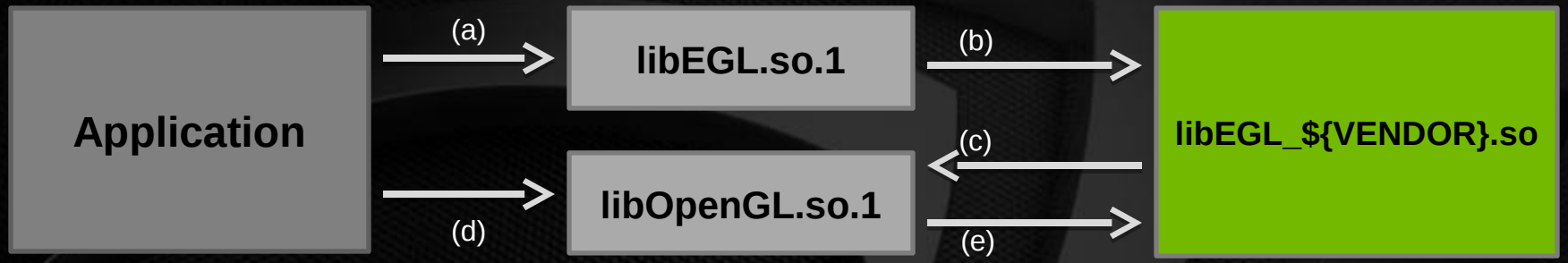
- The vendor-neutral libraries need to dispatch each entry point to a vendor.
- Easy for Client APIs:
  - MakeCurrent defines the vendor to use with the thread.
- Slightly harder for Window System APIs:
  - Many EGL,GLX entry points imply a vendor through their arguments.
  - Some EGL,GLX entry points dispatch based on the current context. E.g., eglWaitGL.
  - Some EGL,GLX entry points return current API state. E.g., glXGetCurrentContext.

  In nearly all cases, the vendor for a window system entry point can be inferred.
- We are not trying to address server-side GLX (yet), so in practice GLX can only have one vendor at a time, per X server, for now.

Example: libGLX.so.1 and libOpenGL.so.1

a) Application calls any GLX entry point.
b) libGLX.so.1 queries the X server, to map X screen to vendor.
c) libGLX.so.1 loads and dispatches to libGLX_${VENDOR}.so.
d) During glxMakeCurrent, libGLX_${VENDOR}.so registers with libOpenGL.so.1; sets up dispatch table.
e) Application calls OpenGL entry point.
f) libOpenGL.so.1 jumps through dispatch table to OpenGL implementation registered by libGLX_${VENDOR}.so.

Example: libEGL.so.1 and libOpenGL.so.1



a) Application calls eglInitialize.
b) libEGL.so.1 uses configuration magic to select, load, and dispatch to libEGL_${VENDOR}.so.
c) During eglMakeCurrent, libEGL_${VENDOR}.so registers with libOpenGL.so.1; sets up dispatch table.
d) Application calls OpenGL entry point.
e) libOpenGL.so.1 jumps through dispatch table to OpenGL implementation registered by libEGL_${VENDOR}.so.

Backwards Compatibility

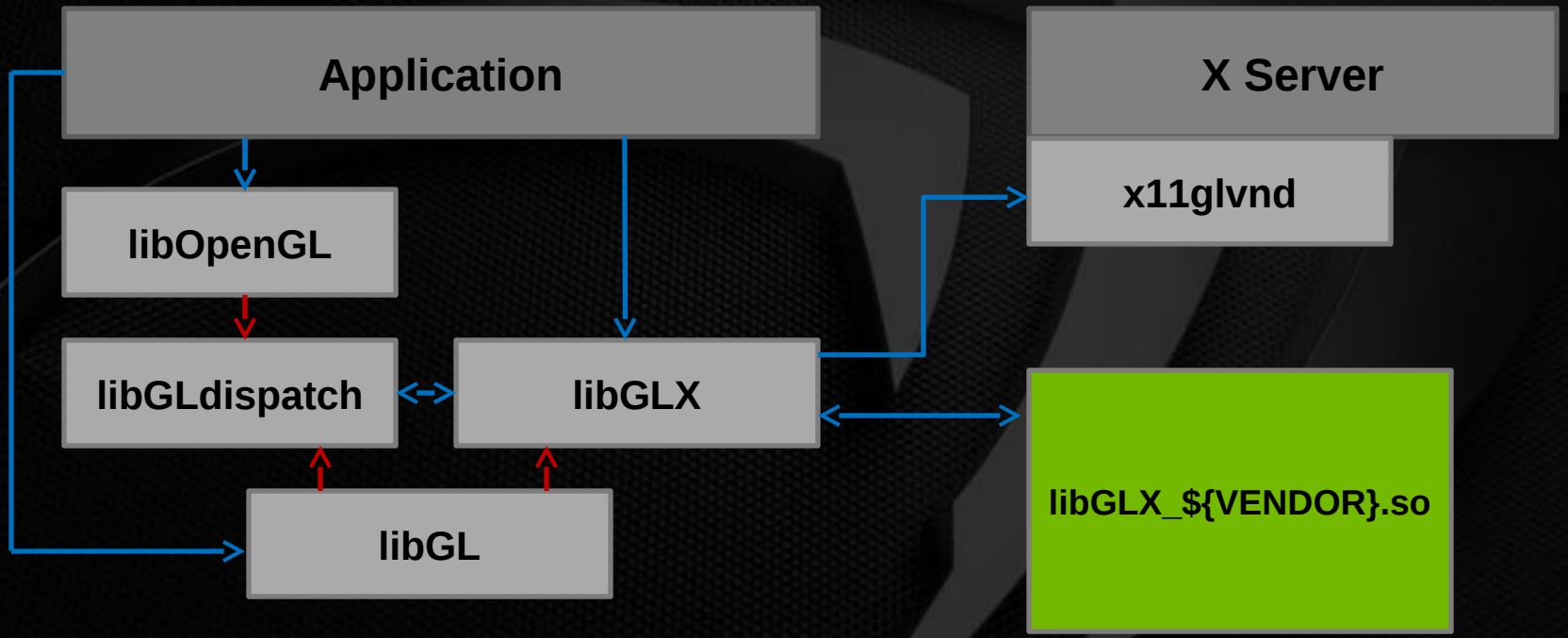
- There will be a libGL.so.1 provided with the vendor-neutral libraries.
- This exports all symbols from all current vendors' libGL.so.1's.
- For symbols provided by libGLX.so.1 or libOpenGL.so.1:
  - use ELF DT_FILTER to resolve libGL.so symbols with libGLX.so.1, libOpenGL.so.1.
- For symbols not provided by libGLX.so.1 or libOpenGL.so.1:
  - use libGLX.so.1's glXGetProcAddress to call from libGL.so.1 to libGLX.so.1, libOpenGL.so.1.
- Existing applications *should* be unaffected.

Status of Implementation

Linux OpenGL Vendor Neutral Dispatch Library (libglvnd) implementation:

https://github.com/NVIDIA/libglvnd

libglvnd Implementation Diagram



LEGEND:
A → B: module A calls into module B
A → B: module A is (logically) a filter library on module B (symbols exported by A are resolved by symbols in B)

libGLdispatch

- Implements core OpenGL dispatching and TLS.
- Acts as a thin wrapper around glapi (taken from Mesa):
  - Provides dispatch table management.
  - Requests vendor proc addresses from the vendor libraries.
  - Tracks making current to a given context + dispatch table.
- Separate library rather than statically linked into libGLX:
  - Current dispatch tables will eventually be shared between GLX and EGL
  - Similar to how glapi operates when Mesa is compiled with the --shared-glapi option.
- Not application-visible.

x11glvnd

- X extension “XGLVendor”
- Tracks XID -> screen, and screen -> vendor mappings.
- libGLX calls this extension to determine the correct vendor to use.

Implementation Progress

- We have a working functional prototype for GLX at https://github.com/NVIDIA/libglvnd
- Testing:
  - Unit tests included with libglvnd
  - Prototyped NVIDIA plugging into libglvnd.

Existing Issues

- ELF symbol filtering exposes glibc loader bug:
  - https://sourceware.org/bugzilla/show_bug.cgi?id=16272
    - dlopen() of a DSO with a DT_FILTER causes the loader to crash
  - ELF symbol filtering: proposed to route libGL symbols to libOpenGL
- Possible performance problems with the current glvnd/vendor ABI that could be fixed.
  - E.g., currently the library calls GetProcAddress() from a vendor one entrypoint at a time, rather than retrieving all the vendor's entrypoints at once.
- Memory management and OOM handling could be improved.

Next Steps

- Work to improve robustness of libGLX against various use cases (multithreading, fork recovery, library load/unload, etc.)
- Solicit feedback on Mesa mailing list.
- Start on libEGL.
  - Need to work out correct vendor selection scheme.
  - Need to work out how libglvnd's libEGL will interact with EGL_EXT_device_base.
- Other OpenGL implementors can start experimenting with plugging into libglvnd, and providing feedback.

Migration

- Most likely, deploy with NVIDIA first:
  - NVIDIA driver package would include a snapshot of the vendor-neutral libraries.
  - If vendor-neutral libraries not already present on system, NVIDIA package installs its copies.
  - Get broader testing.
- Hopefully get feedback from Mesa, AMD on how well it works to plug into vendor-neutral libraries.
- Once we have confidence in it, lock down the ABI between vendor-neutral libraries and vendor libraries.
- Encourage distros to start packaging the vendor-neutral libraries

Thank You

All the implementation work so far has been done by Brian Nguyen
(brnguyen at nvidia.com)

References

- These slides: https://github.com/aritger/linux-opengl-abi-proposal/blob/master/presentation-xdc2014.pdf
- Last year's slides: https://github.com/aritger/linux-opengl-abi-proposal/blob/master/presentation-xdc2013.pdf
- Proposal: https://github.com/aritger/linux-opengl-abi-proposal/blob/master/linux-opengl-abi-proposal.txt
- Current implementation: http://github.com/NVIDIA/libglvnd

Questions