GDP - Feature #102

Easier GDP Deployment: shared library issues

06/10/2017 10:50 AM - Anonymous

Status:	New	**Start date:**	06/10/2017
Priority:	Low	**Due date:**	
Assignee:		**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:			

Description

This issue is for discussion about possible easier deployment strategies for the GDP. I've created an issue because I can't see how to be sure that Edward and others see the discussion using the forum. I did not find a gdp developers mailing list, so I'm using the issue tracker. For reference, see Mailing list for GDP Users (https://gdp.cs.berkeley.edu/redmine/issues/33)

One of the issues that happened to Edward yesterday is that when he tried to run the GDP inside Ptolemy, libevent was not installed and he received a somewhat mysterious message. One workaround would be to attempt to detect this and prompt the user to install libevent, but this is quite a bit of platform-dependent work and likely to cause problems. It would be better to ship a libgdp with a minimum of shared library dependencies.

Below are the details.

For Java and Node, we require a GDP shared library that can be loaded at run time.

The problem is that the gdp shared library uses other shared libraries that must be installed.

Under Mac OS X, libgdp uses two libevent2 shared libraries and openssl

```
bash-3.2$ otool -L $PTII/lib/libgdp.0.7.dylib
/Users/cxh/src/ptII11.0.devel/lib/libgdp.0.7.dylib:
        libgdp.0.7.dylib (compatibility version 0.0.0, current version 0.0.0)
        /usr/local/opt/libevent/lib/libevent-2.1.6.dylib (compatibility version 7.0.0, current ver
sion 7.2.0)
        /usr/local/opt/libevent/lib/libevent_pthreads-2.1.6.dylib (compatibility version 7.0.0, cu
rrent version 7.2.0)
        /opt/local/lib/libcrypto.1.0.0.dylib (compatibility version 1.0.0, current version 1.0.0)
        /usr/lib/libSystem.B.dylib (compatibility version 1.0.0, current version 1238.51.1)
```

So, libevent2 must be installed for this to work, or we need to include the libevent2 shared libraries with the Java and Node installers. This is not impossible, just a bit more work.

libcrypto1.0.0 itself uses libz:

```
bash-3.2$ otool -L /opt/local/lib/libcrypto.1.0.0.dylib
/opt/local/lib/libcrypto.1.0.0.dylib:
        /opt/local/lib/libcrypto.1.0.0.dylib (compatibility version 1.0.0, current version 1.0.0)
        /opt/local/lib/libz.1.dylib (compatibility version 1.0.0, current version 1.2.11)
        /usr/lib/libSystem.B.dylib (compatibility version 1.0.0, current version 1238.0.0)
```

I would like to avoid shipping libcrypto and libz. Under Mac OS X, /usr/lib/libcrypto.0.9.7.dylib is present. One issue is that libevent2 seems to require libcrypto.1.0.0.dylib, which seems to mean we would need to include libcrypto.1.0.0.dylib

I'm a little foggy about this, but I believe that under Linux, I was not able to share a common libgdp between RHEL and Ubuntu because of OpenSSL issues and glibc. I'd like to be able to drop RHEL, but that is what the build machine is running. I'm not sure of the details.

Building a libgdp shared library that includes everything is one possibility. I tried to do this quite a while ago and failed. My notes at

https://www.icyphy.org/accessors/wiki/Notes/GDPWithAccessorsNotes#GDPBrowserIssues say:

> Building a gdp shared library that includes the dependent shared libraries is not possible under RHEL because libevent was not compiled with -fPIC

One step would be to work further on building a gdp shared library that included the dependent shared libraries.

Under RHEL, libgdp has the following dependencies.

```
[cxh@terra ~]$ ldd /home/cxh/src/ptII11.0.devel/lib/linux-x86-64-rhel/libgdp-0.7.so
        linux-vdso.so.1 =>  (0x00007ffd03278000)
        libevent-2.0.so.5 => /usr/local/lib/libevent-2.0.so.5 (0x00007f74dfaca000)
        libevent_pthreads-2.0.so.5 => /usr/local/lib/libevent_pthreads-2.0.so.5 (0x00007f74df8c700
0)
        libcrypto.so.10 => /usr/lib64/libcrypto.so.10 (0x00007f74df4c8000)
        libavahi-client.so.3 => /usr/lib64/libavahi-client.so.3 (0x00007f74df2b8000)
        libavahi-common.so.3 => /usr/lib64/libavahi-common.so.3 (0x00007f74df0ab000)
        libc.so.6 => /lib64/libc.so.6 (0x00007f74ded17000)
        librt.so.1 => /lib64/librt.so.1 (0x00007f74deb0f000)
        libdl.so.2 => /lib64/libdl.so.2 (0x00007f74de90a000)
        libz.so.1 => /lib64/libz.so.1 (0x00007f74de6f4000)
        libdbus-1.so.3 => /lib64/libdbus-1.so.3 (0x00007f74de4b3000)
        libpthread.so.0 => /lib64/libpthread.so.0 (0x00007f74de295000)
        /lib64/ld-linux-x86-64.so.2 (0x0000003611200000)
[cxh@terra ~]$
```

If the gdp is built with "make all_noavahi", then libavahi-client and -common are not required. However, we have the same issues with libevent, libevent_pthreads and libcrypto.

A side issue about all of this is that currently, the Node gdp module provides shared libraries for the Darwin, RHEL and possibly Ubuntu platforms. A better Node installer would include the gdp source code and build it from sources. I'm not sure how to handle the requirement for libevent in the installer. Do we tell the user to install libevent? Do we include the sources? Research would need to be done.

I see Node as a very large community where the GDP (and CapeCode) could have great impact, hence the importance of supporting Node.

Anyway, the above are my thoughts about issues surrounding deploying the GDP client. These issues are why I see the RESTful interface or WebSockets to be a better way forward than providing a C-based gdp client.

Does anyone have any thoughts or suggestions how we could reduce the shared library dependencies?

History

#1 - 06/16/2017 03:14 PM - Eric Allman

I have a couple of observations here:

First, isn't this really just an issue that we haven't "product-tized" the libraries yet? For example, we can produce a Debian package that includes the dependencies, so that an install of the GDP library pulls in the dependencies. However, Debian is our only "Tier 1" platform, and we simply don't have enough people or time to do this for all the officially unsupported platforms, which include RHEL and MacOS. It's convenient that they work at all, but you have to expect that you will need special effort to install on an unsupported platform.

It is true that we haven't been producing binary Debian packages on a regular basis for the two platforms we seem to be using (SwarmBox and BeagleBoneBlack). We could do that, but frankly those installations seem to be fairly static, so there isn't a lot of bang for the buck.

I wish I had a build management team that could do proper QA on multiple platforms and produce binary releases on a weekly (or at least monthly) basis, but the reality is that we have three people to do the research, development, documentation, testing, packaging, distribution, and maintenance of the servers that we make available to the community. I would like this to be easier for users too, but I note that if you compile from source (as you require with Ptolemy) then the build scripts take care of these installations. If for some reason we are missing a dependency I'm happy to fix it.

Your comment about the RESTful interface is quite apropos. It is slower and less secure than a direct interface, but it does have the advantage of requiring no software installation at all on client machines. In the longer term, I hope we will have proper packages for all the significant platforms that

people want, but for now it's really no harder than installing Ptolemy.

#2 - 06/16/2017 03:46 PM - Anonymous

Thanks for the reply. I was hoping that there would be some magic bullet that would create a gdp shared library that had the functionality required by the other shared libraries in it or something like that.

One big issue is that to build the GDP from source under Linux requires that libevent2 be installed, which requires running sudo, which typically requires the user to enter their password. This makes automatic building, say in node, a bit tricky. For example, if I want to deploy a Swarmlet that uses the GDP to an arbitrary Linux box, the I need to have root access on that remote box. Do you have any suggestions about how to over come this limitation?

MacOS is a bit easier because brew can be installed without root access.

At this point, I think it is better to look at REST for appending to logs and WebSockets or something else for subscription.

#3 - 06/16/2017 03:54 PM - Eric Allman

So I'm curious why libevent is such an issue. libgdp also requires openssl, jansson, libdb, etc., all of which require sudo, but these haven't been flagged. And of course, make install requires sudo.

#4 - 06/16/2017 03:56 PM - Eric Allman

Ah, one other thing. If you want to build a "mega-library" that includes all dependencies but are stuck on a missing -fPIC, have you considered compiling libevent from source? It seems to be simple to do.

#5 - 06/16/2017 04:04 PM - Anonymous

The reason I'm focused on libevent is because libgdp has dependencies on it. I can build libgdp without avahi by running "make all_noavahi" and the resulting shared library needs neither jansson nor libdb. openssl is almost always installed, so it is not a huge problem, except for the issue between old RHEL 6 and Ubuntu that I discussed in the initial text for this issue.

Under Darwin, libgdp has the following dependencies:

```
bash-3.2$ otool -L $PTII/lib/libgdp.0.7.dylib
/Users/cxh/src/ptII11.0.devel/lib/libgdp.0.7.dylib:
        libgdp.0.7.dylib (compatibility version 0.0.0, current version 0.0.0)
        /usr/local/opt/libevent/lib/libevent-2.1.6.dylib (compatibility version 7.0.0, current version 7.2.0)
        /usr/local/opt/libevent/lib/libevent_pthreads-2.1.6.dylib (compatibility version 7.0.0, current versio
n 7.2.0)
        /opt/local/lib/libcrypto.1.0.0.dylib (compatibility version 1.0.0, current version 1.0.0)
        /usr/lib/libSystem.B.dylib (compatibility version 1.0.0, current version 1238.51.1)
```

I think I started to look at compiling libevent from source, but I think it turned in to a nightmare. I'm look for an easy solution to this issue, and consider recompiling libevent to be out of scope for me.