Full Stack on Wine

Create a Win-Win between Wine and thousands of Win32 open source projects

Qian Hong
Survey

- Chromium Browser
- Chromium Embed Framework (CEF)
- Python
- Lua
- Dazhihui Financial Security Software
<table>
<thead>
<tr>
<th>Closed / Mixed source productions</th>
<th>Open source third party libraries</th>
</tr>
</thead>
<tbody>
<tr>
<td>Long Tail</td>
<td>Famous</td>
</tr>
</tbody>
</table>
Story – Hacking experience on Dazhihui

- CEF
- Python
- Lua
- Wine
<table>
<thead>
<tr>
<th>Mixed source production as a combined blackbox</th>
<th>Open source projects as independent components</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expensive</td>
<td>Relative Non-expensive</td>
</tr>
</tbody>
</table>
How could we make sure Dazhihui is compatible with Wine if we are even not sure how compatible is CEF / Python / Lua on Wine?
This makes me thinking...
Normal User / Dev knows two domains

Website User

Website Dev = Firefox User

Firefox Dev = libstdc++ User

libstdc++ Dev = Kernel User

Kernel Dev
Wine User / Dev knows everything...

IE 8 crashes!

Website User

Wine Dev

……
Structure of Dev / User community

Normal software communities: layer by layer

Wine communities: missing middle layers
Story – Nightmare of upgrading Valve Steam and Tencent QQ

- Chromium v44.0.2378.0 introduced a new feature …
- CEF v3.2357.1273 Inherit the new feature from Chromium …
- Valve Steam (steamwebhelper) upgrades its bundled CEF …
- Bomb!
Question – How to fix the nightmare of upgrading?
Good doctor vs Great doctor

Good doctor treats the disease after a patient sicks

Great doctor prevents the disease before the patient sicks

– Chinese Folk Tale
Good doctors survive
Great doctors starve

– WineConf2015
Summary - Our goals

- Save our Dev from blackbox debugging
- Save our QA from nightmare
- Fill the gap between end user and low level dev
- Catch bug before application update
Solution – Full Stack on Wine

FOR OUR PARTNERS

- Create a centralized Win32 compiling and deployment platform base on Wine.
- Combine Valgrind with Wine as additional value to Win32 developers.

FOR WINE

- Improve the compatibility of Win32 open source libraries on top of Wine, layer by layer.
- Coordinate release plan with famous Win32 open source projects.
What is a 'stack'?  

<table>
<thead>
<tr>
<th>Chromium / CEF</th>
</tr>
</thead>
<tbody>
<tr>
<td>gtk, libwebp, libogg, speedx, sqlite3, nspr, nss</td>
</tr>
<tr>
<td>cairo, pango, atk, gdk-pixbuf2, libxslt, ncurses, speexdsp</td>
</tr>
<tr>
<td>fontconfig, freetype, libpng, libjpeg, libtiff, glib2, gettext, libxml2, libgcrypt, libsysstre, ...</td>
</tr>
<tr>
<td>gcc-libs, gmp, libwinpthread, bzip2, zlib, xz, libiconv</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Wine</th>
</tr>
</thead>
</table>
What we did before...

Tencent QQ: Sorry, I'm a big black box...

Wine: WTF?
What we hope in the future...

<table>
<thead>
<tr>
<th>Tencent QQ: I'm still a black box, but smaller...</th>
</tr>
</thead>
<tbody>
<tr>
<td>CEF, LibFoo, LibBar, LibCool, LibBoring...</td>
</tr>
<tr>
<td>gtk, libwebp, libogg, speedx, sqlite3, nspr, nss</td>
</tr>
<tr>
<td>cairo, pango, atk, gdk-pixbuf2, libxslt, ncurses, speexdsp</td>
</tr>
<tr>
<td>fontconfig, freetype, libpng, libjpeg, libtiff, glib2, gettext, libxml2, libgcrypt, lib systre, ...</td>
</tr>
<tr>
<td>gcc-libs, gmp, libwinpthread, bzip2, zlib, xz, libiconv</td>
</tr>
</tbody>
</table>

Wine: Hello, folks!
Step by Step, Layer by Layer

• Step One (relative harder)
  – Manually test or run testsuite on Wine, fix test failures.

• Step Two (relative easier)
  – Continuous Integration Testing platform based on Wine
Introduce MSYS2

- Derivative of Cygwin
- Replacement of MSYS
- Pacman
- POSIX build environment: msys2_shell
- Win32 build environment: mingw32_shell / mingw64_shell
- 300+ MSYS2 packages, 700+ MinGW packages
- MSYS2 = a Win32 distribution, Full Stack on Windows
- Wine + MSYS2 = Full Stack on Wine
Work on Wine + Cygwin / MSYS2

- Several months of work
- Fixed 30+ bugs, patches in Wine and Wine Staging
- Reported 5 upstream bugs, one is detected by Valgrind
- Strong support from Cygwin Devs / MSYS2 Devs / XZ Devs
- Impossible without Wine Staging
Status of Wine + Cygwin / MSYS2

- Almost work out of box (Wine Staging 1.7.51)
  - Export STAGING_WRITECOPY=1
- Manual testing
  - gcc / clang / gdb / strace / coreutils / binutils / make / git / autoconf / xz ...
- Integration testing
  - xz: 100% pass (9/9)
  - libpng: 100% pass (40/40)
  - gettext: 93.8% pass (361/385)
Wine + MSYS2 as Win32 compiling platform

• Latest Wine Staging + several local hacks:
  – GCC (MSYS2) 4.9.2, 8+ hours, 3 stagings
  – Qt (MinGW) 4.8.7, 20+ hours, 200+M source code
  – 300+ MSYS2 packages
  – 50+ MINGW packages
    • Boost, Binutils, Cairo, ffmpeg, Inkscape, Gimp, Qt, QtCreator, SQLite3, Tor, WineD3D, Wget, WxWidgets, etc
  – All above done by self-compiled MSYS2 runtime using Wine MSYS2
Next plan

• Build a centralized continuous compiling and continuous deployment platform based on Wine
  – Drone CI / Travis CI / Buildbot / Jenkins
• Invite Win32 open source developers
• Integrate Valgrind
• Work with Win32 partner projects to fix Wine failures
Why maintain a centralized build service?

- Self controllable environment
  - Kernel, libc, filesystem, wine, etc
- Quick bug fix
- Build connections with open source win32 developers, build a developers' community
- Required by next next plan
Future Dream (next next plan)

- **Plan A**
  - Provide commercial service to Windows software vendors

- **Plan B**
  - Provide commercial technical support to mainstream CI vendors
    - Travis CI, Appveyor CI, Drone CI, Circle CI, Snap CI, etc
Advantage, Disadvantage

• We are good at
  – Unlimited parallel builds, License free
  – Save hardware resource
    • Wine + Docker vs Windows inside VM
  – Valgrind

• We can't guarantee
  – Zero false positive
Theory - the more the better

• Dream
  – Convince Top 1000 open source win32 projects to use our solution
  – Build a Wine/Win32 developer community like a Linux distribution
Real world cases

- LibreOffice (FOSDEM 2013)
- Firefox
- Git for Windows
- Bitcoin
- llvm
- Search in github: filename:..travis.yml wine
Summary – Our plan

- Continuous Compiling / Deployment service
- Valgrind support
- Fix test failures, layer by layer, expect a long term job
- Always welcome new contributors...
- Seek for business chance
Call for help – contribute time

- Contribute time
- Upstream existent wine staging patches
- Continue improve remain hacks
- Test Win32 open source libraries / applications
- Fix Wine bugs
- Improve Valgrind symbol support
- Improve WineDBG / GDB symbol support
Call for help – contribute connections

• Do you have connections to:
  – GCC, Clang, GDB, Git, etc
  – Python, Perl, PHP, Lua, OpenJDK, Rust, Go, Node.js, etc
  – CEF, NSIS, etc
  – Cocos2dx, Godot, Minetest, Blender, OpenRA, MonoGame, etc
  – LibreOffice, Firefox, Gimp, Inkscape, etc

• Please invite them to join us
  – Investigate chance to Win-Win
  – Co-mentor future GSoC projects
Call for help – contribute ideas

• Any combination similar to Wine + Valgrind
  – Useful for Win32 application developers
  – Available only on unix
  – Possible case: LibFuzzer, Klee, Reverse Debugging, etc
Proposal
Build consensus focus on Win32 open source software

• Top to Bottom
  – Steam folks and QQ folks
    • Could we co-work on CEF?
    • Binary scanning tool to discover common used open source libraries?

• Bottom to Top
  – Cocos2dx
    • What games are developed by cocos2dx?
Proposal
Coordinate release plan with our partner project

- Wine side: Milestone / Stable Release
  - Could each of 'Fixed all gcc/clang/gdb/git test failures' suggest a milestone?
  - Could 'Fixed all Chromium 39 / CEF 3.2 test failures' suggest a stable release?

- Partner side: wait for Wine!
  - Could we ask CEF / cocos2dx / etc reserve a reasonable time for Wine community to test their software before final release?
Welcome to TeaConf2018 @China
Thank You!