WineTestBot

A Wine gatekeeper and test farm for Wine developers

An ongoing test automation odyssey

François Gouget
Conformance tests

- Microsoft defines the Win32 API
- But the documentation is incomplete, e.g.
  
  ```
  found = PathResolve(basename, dirs, flags);
  notepad.exe    -> c:\windows\notepad.exe
  notepad        -> ???
  ```

⇒ Write tests in typical blackbox testing fashion

- Keep the tests to verify the Wine implementation
- Wrote a tool to run all the tests and a site to collect the results
Got Windows?

- Wine developers use Unix / Mac OS X
  - They usually lack easy access to Windows
  - Conformance tests get written on Wine!

- Which Windows version to test on?
  - More than one
  - Also test different locales, user privileges, etc.

- Wine developers need a Windows test farm
The gatekeeper

- How to keep incorrect tests out?
- Patch reviewers cannot guess the API behavior
  - Automatically run all test patches on Windows

- How to motivate developers to fix their tests?
  - Verify them before commit
Consequences

- Runs untrusted code
- A lot of Windows version and configurations to test
  - Use virtual machines for sandboxing
  - Revert them to a clean state after each test
  - Only let individual tests run for 2 minutes
Architecture

WineTestBot Server

Test Engine

DB + Files

WineTest Update Cron Batch

WebPatch Handler Batch

Web Frontend

Patches Website

Notification Email

Patch or Exe

Notification Email

Status Info

Jobs & Tasks

Status Query & Update

Notifications

Revert / Stop VMs

Patch

Test Exes + Logs

Test Exe

Logs

VM Host 1

Build VM

Win VM 1

Win VM 2

VM Host n

Win VM n
Comparison with BuildBot

- **TestBot**: The Server starts and sends work to the VMs
  - **BuildBot**: Slaves ask the Master for work
    - Probably easier to have ‘third-party’ slaves

- **TestBot**: Monolithic, new tasks require changes in the DB
  - **BuildBot**: Very modular, for tasks but also reporting, etc.
    - But geared towards testing patches post-commit
    - Needs Python on the Slaves -> quite a lot of dependencies

- **TestBot**: Reverts to running VM snapshots
- **BuildBot/LibVirtBuildSlave**: Clones VMs
  - Cloning + booting takes longer than our jobs
  - Can duplicate a VM if it has a lot of jobs (license permitting)
Status

- Switched from VMware ESX to libvirt and QEmu / KVM
- Test drive the new WineTestBot at: http://newtestbot.winehq.org
- Three VMs: Win 7 64, Win 8 32 & 64
- Configuring the VMs so the tests pass / trying to figure out why the tests fail
Basic VM configuration

- Disable the screensaver
- Disable Windows update
- Disable ‘Search Indexing’
- Disable ‘Restauration Points’
- Typically clean Windows install with no extra package
Future improvements (1)

- Add more VM configurations:
  - Windows versions
  - Multi-core, multi-network card VMs, etc.
  - Various locale / language combinations
  - Run tests in non-administrator accounts
  - See Wine-Testbot bugs for more...

- But limit the number of VMs (and licenses)
  - Combine some configurations
  - Only test the latest service pack?
    - Decision to be made by the community
Future improvements (2)

- Use multiple snapshots per VM
  - Tweak TestBot to only run one snapshot / VM at a time
  - Tests are serialized -> try to avoid bottlenecks

- Improve VM management and status display

- Solve the ‘VBScript’ test bug : i.e. which tests should be re-run when a patch modifies a non C test file

- Pending patches (e.g. TestLauncher fix)
Testing Wine (1)

- WineTestBot reduced test failures on Windows
  - Now it’s Wine’s turn

- Will be a lot more intensive:
  - Every patch needs testing
  - A kernel32 bug could impact any test
    - => re-run them all?
  - Any unreliable test will generate spam

  - Maybe start simple:
  - Only run the dll’s tests
Testing Wine (2)

- Start with Unix platforms
  - Linux
  - Hopefully FreeBSD and Solaris too
- Mac OS X will present some challenges
  - For licensing reasons it seems the VM should be run on Mac hardware
  - No Qemu on Mac?
  - Libvirt support on Mac?
Beyond VMs

- Virtual graphics cards are useless for Direct3D
- Virtual sound cards are not so good either
- Impacts both Windows tests and future Wine ones

⇒ Run the tests on real hardware
  - But still start from a clean state
  - Still isolate the code

⇒ Explore KVM’s graphics card pass-through and/or Grub network boot (more on bug 31786)
Win 7 Results 1

`ddraw:ddrawmodes.c:480`: Test failed: Expected 0 modes got 14 (solved with QXL driver)

`kernel32:console`: Timeout (QEmu graphics is just too slow)

`gdi32:dc.c:134`: Test failed: rects are not equal: (0,0-100,100) - (0,0-100,29)
`gdi32:dc.c:158`: Test failed: rects are not equal: (0,0-50,50) - (0,0-50,29)
`gdi32:dc.c:169`: Test failed: rects are not equal: (0,0-100,100) - (0,0-100,29)
-> Only happens sometimes 😊

`mmdevapi:render.c:1003`: Test failed: Position 6336 too far after playing 100ms

`ntdll:exception.c:621`: Test failed: B0 flag is not set in Dr6
`ntdll:exception.c:629`: Test failed: BS flag is not set in Dr6
`ntdll:exception.c:637`: Test failed: B0 flag is not set in Dr6
`ntdll:exception.c:646`: Test failed: BS flag is not set in Dr6

`rpcrt4:rpc.c:823`: Test failed: GUID does not appear to contain a MAC address
Win 7 Results 2

```
urlmon:url.c:1498: Test failed: hres = 1, expected 0
...

wintrust:softpub.c:502: Test failed: Expected S_OK, got 00000001
wintrust:softpub.c:515: Test failed: Expected cert to be self-signed
wintrust:softpub.c:517: Test failed: Expected CERT_CONFIDENCE_SIG | CERT_CONFIDENCE_TIMENEST, got 00100000

ws2_32:sock.c:4794: Test failed: Local socket address is different 0.1.0.0 != 127.0.0.1
-> Happens on other VMs too

ws2_32:sock.c:5892: Test failed: failed to get completion status 0
ws2_32:sock.c:5893: Test failed: Last error was 995
ws2_32:sock.c:5895: Test failed: Number of bytes transferred is 0
ws2_32:sock.c:5897: Test failed: Internal status is c0000120
ws2_32:sock.c:5958: Test failed: Last error was 995
ws2_32:sock.c:5962: Test failed: Internal status is c0000120
```