Containerized Curl Testing

Daniel Fandrich
April 2018

curl’s Test Suite

- Both unit and integration tests
- Self-contained, mostly internal servers
- perl is main additional dependency
- Easy to run: make test
- 1194 test cases (April 2018)
- Auto-detects most compiled-in curl features
- Auto-detects most environment requirements

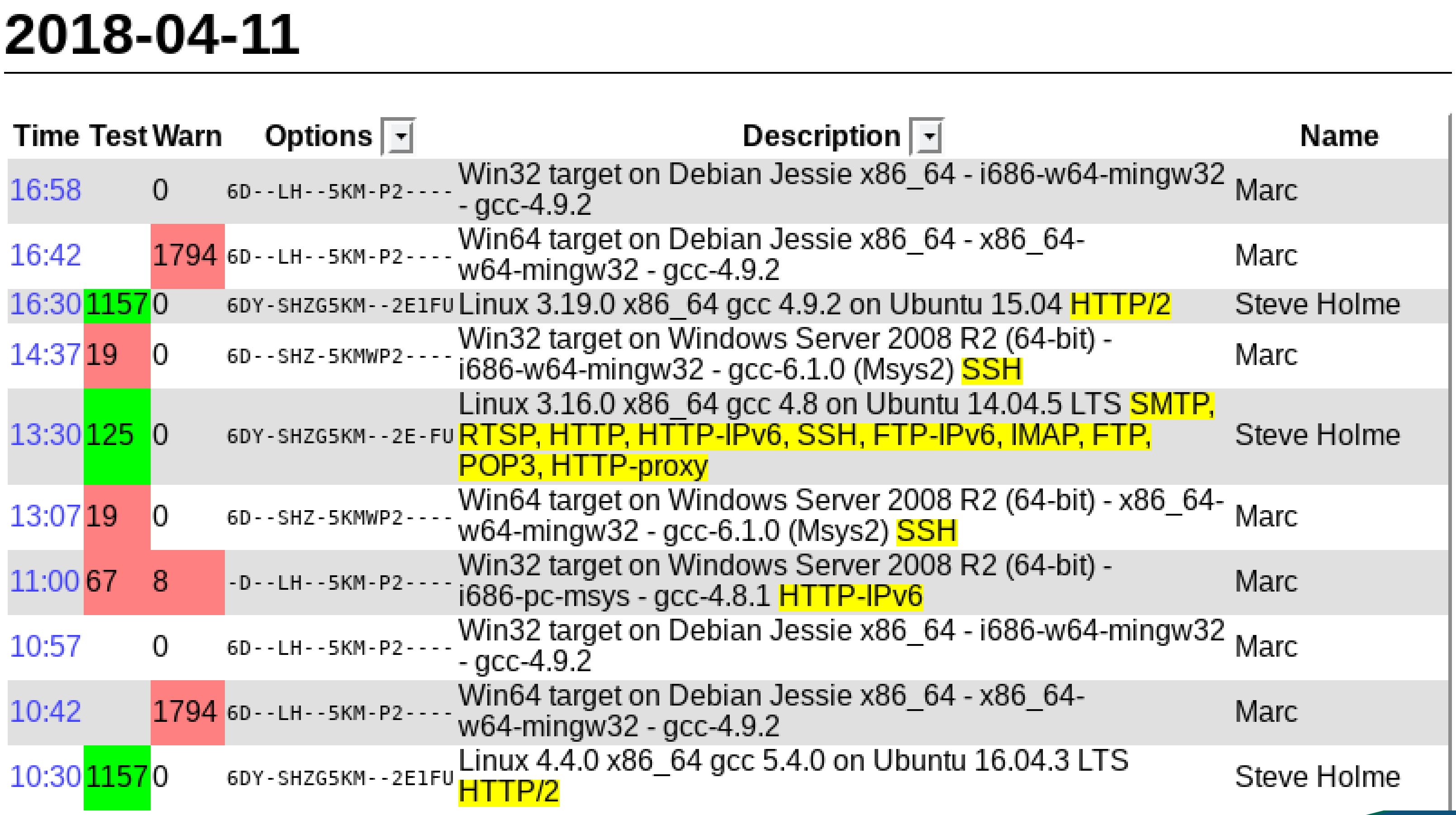
git-integrated Builds

- AppVeyor (Windows)
- Travis CI (Linux, Mac OS)
- Coveralls (code coverage)
- Coverity (static analysis)
- Automatically build changed code

curl Autobuilds

- https://curl.haxx.se/dev/builds.html
- Users can run test suite and upload results
- Slightly more setup
- Even more setup to automate regular builds
- 5 people contribute majority of builds
- 4 OSes covered
- 43 unique build configurations

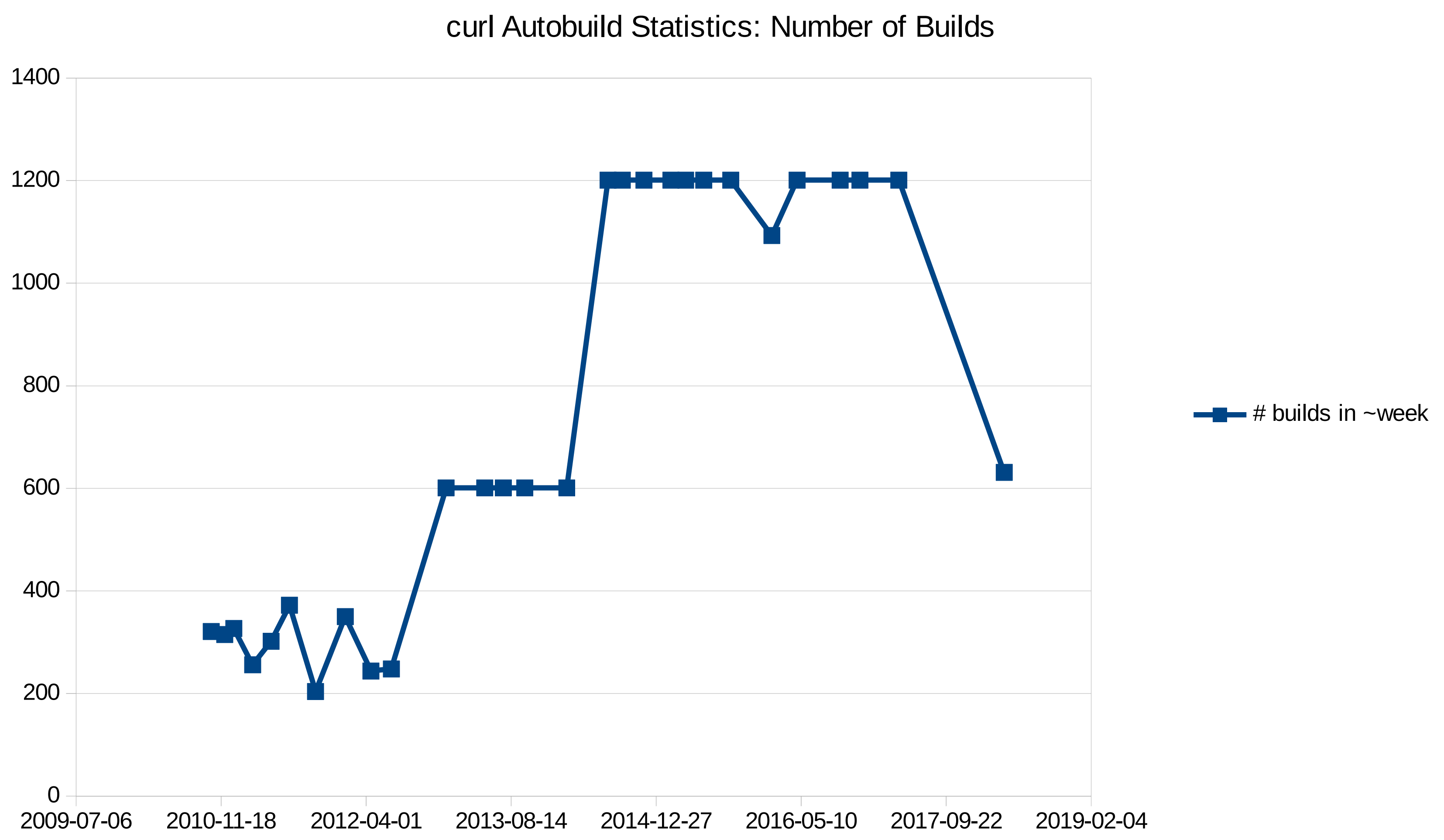
curl Autobuilds



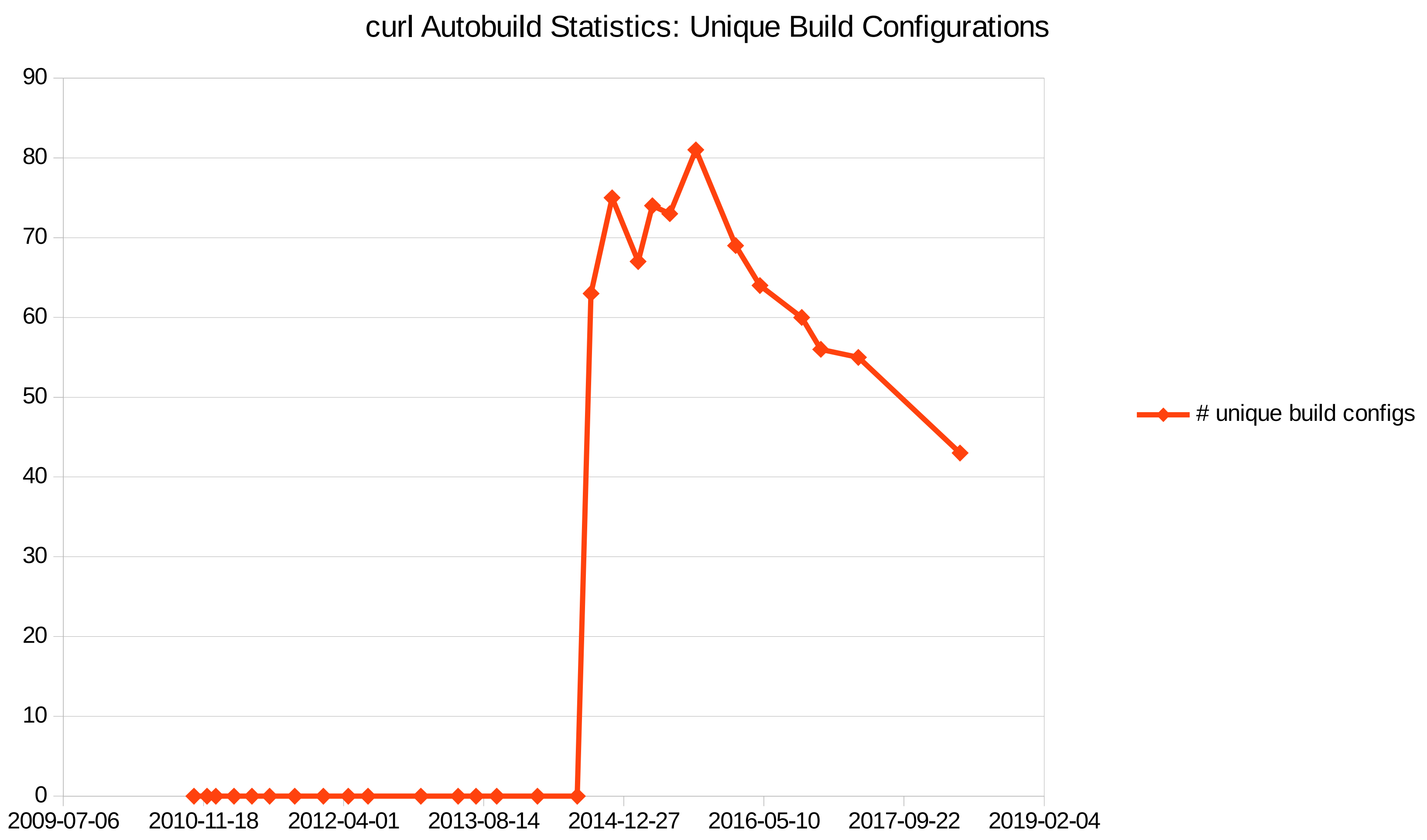
2018-04-11

Time	Test	Warn	Options	Description	Name
16:58		0	6D--LH--5KM-P2----	Win32 target on Debian Jessie x86_64 - i686-w64-mingw32 - gcc-4.9.2	Marc
16:42		1794	6D--LH--5KM-P2----	Win64 target on Debian Jessie x86_64 - x86_64-w64-mingw32 - gcc-4.9.2	Marc
16:30	1157	0	6DY-SHZG5KM--2E1FU	Linux 3.19.0 x86_64 gcc 4.9.2 on Ubuntu 15.04 HTTP/2	Steve Holme
14:37	19	0	6D--SHZ-5KMWP2----	Win32 target on Windows Server 2008 R2 (64-bit) - i686-w64-mingw32 - gcc-6.1.0 (Msys2) SSH	Marc
13:30	125	0	6DY-SHZG5KM--2E-FU	Linux 3.16.0 x86_64 gcc 4.8 on Ubuntu 14.04.5 LTS SMTP, RTSP, HTTP, HTTP-IPv6, SSH, FTP-IPv6, IMAP, FTP, POP3, HTTP-proxy	Steve Holme
13:07	19	0	6D--SHZ-5KMWP2----	Win64 target on Windows Server 2008 R2 (64-bit) - x86_64-w64-mingw32 - gcc-6.1.0 (Msys2) SSH	Marc
11:00	67	8	-D--LH--5KM-P2----	Win32 target on Windows Server 2008 R2 (64-bit) - i686-pc-msys - gcc-4.8.1 HTTP-IPv6	Marc
10:57		0	6D--LH--5KM-P2----	Win32 target on Debian Jessie x86_64 - i686-w64-mingw32 - gcc-4.9.2	Marc
10:42		1794	6D--LH--5KM-P2----	Win64 target on Debian Jessie x86_64 - x86_64-w64-mingw32 - gcc-4.9.2	Marc
10:30	1157	0	6DY-SHZG5KM--2E1FU	Linux 4.4.0 x86_64 gcc 5.4.0 on Ubuntu 16.04.3 LTS HTTP/2	Steve Holme

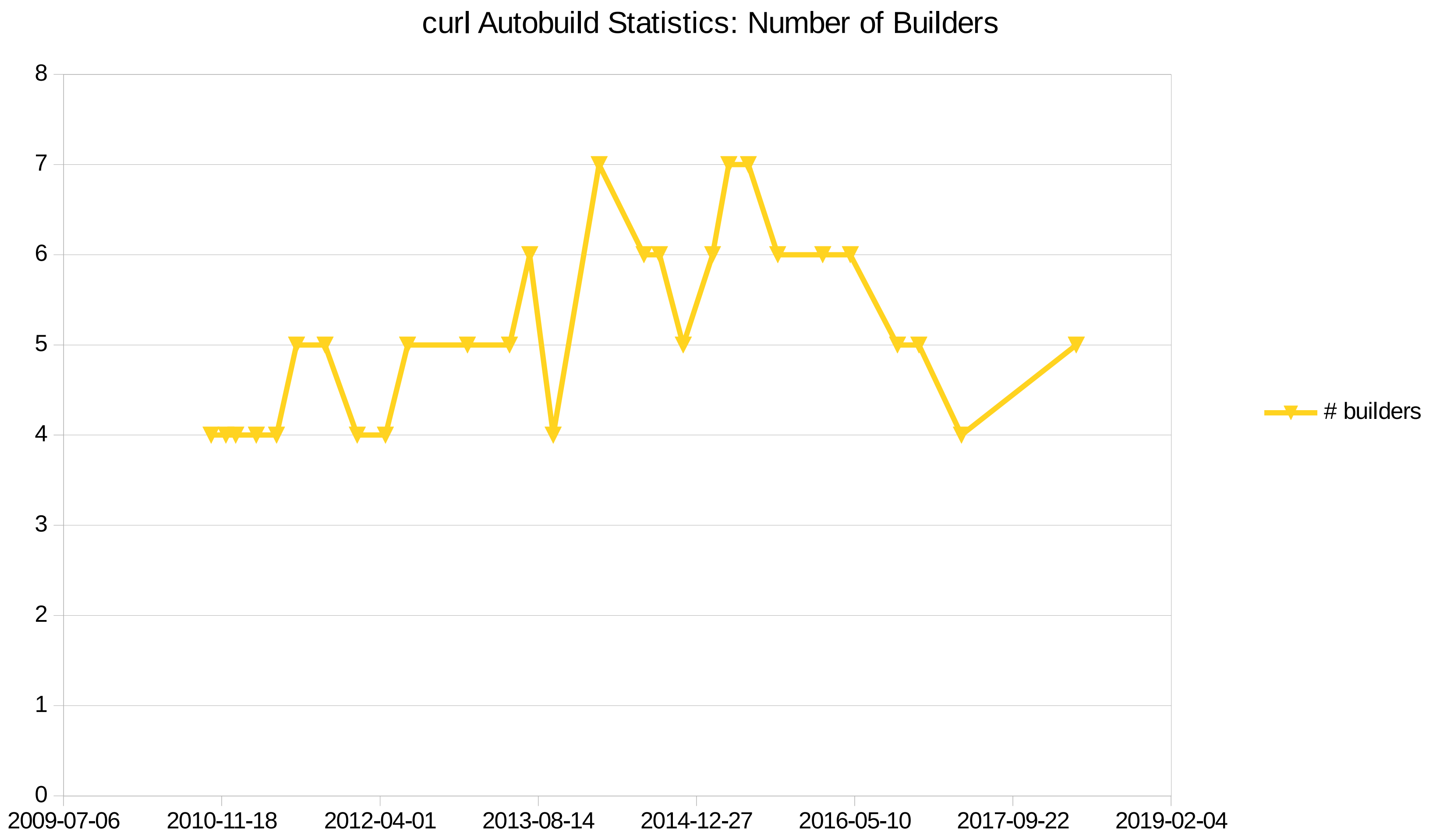
curl Autobuilds



curl Autobuilds



curl Autobuilds



Compatibility

- curl is highly portable
- INSTALL.md mentions 70 compatible systems
- “Linux” alone comes in infinite variations
- Architectures, hardware, kernel versions, compilers, libraries can all differ substantially

Testing Variations: One Machine

- Install multiple libraries, configure curl for one at a time (possible for e.g., TLS)
- Installing multiple *versions* of libraries a lot harder to manage
- Install multiple compilers, configure curl for one at a time
- Few systems support many compilers

Testing Variations: VMs

- Install different OSes in VMs
- Different OSes possible
- Heavyweight solution
- Need infrastructure to manage them
- Effort equivalent to keeping N different servers up-to-date
- Few alternatives for testing multiple OSes

Testing Variations: Containers

- Build/test environment can be completely separate from main system
- Images are smaller, faster to start, easier to update
- People maintain ready-to-use images
- Easy to customize images (e.g., to add other libraries)
- Mostly Linux only

Containers: lxc

- My goals were mostly security & isolation
- R/O filesystem except for /tmp RAMdisk
- Restricted view of filesystem
- No external network (just `lo`)
- Capability dropping, limited RAM, processes, etc.
- Development environment is otherwise same as host system

Containers: Docker

- More functionality—early versions ran on lxc
- Provides easier management for completely separate environments
- Base image with overlay filesystem for ephemeral files
- By default, container can't see any host files—must supply everything (libc, busybox, etc.)
- Image is configured from a single Dockerfile

Containers: Docker

- Docker Hub provides community-supplied base images
- Base system with coreutils/busybox, base libraries, and usually a package manager
- Can extend these images to make new ones yourself by installing new packages
- Many existing images are ready for download with one command

Containers: Docker

- Alpine: MUSL based; checks glibc assumptions
- NixOS: Unusual package system with symlinks everywhere
- CentOS: Old library compatibility
- Debian: Everything enabled
- uClibc: Another non-glibc libc

curl://up

Containers: Setting One Up

- Find out how the package manager works
- Install needed packages (e.g., gcc, -dev)
- Don’t bother with git, just install nightly tarball
- Can’t assume curl is available to download source
- Include a small script to do the above in the custom Docker image
- Periodically rebuild the autobuild base images to pick up updates

What You Can Do

- Docker approach is pretty reproducible—too reproducible
- Find a base image that's not getting autobuilds
- Figure out how to use it
- Start a curl autobuild using it
- Next level: create a new public Docker base image first

What You Can Do

- Set up VM based images to build curl on other OSes
- Which ones? Are there any interesting ones left?
- If you make the effort, curl.haxx.se will host your build logs
- Better utilize the build logs

Questions