



# The curl Test Suite

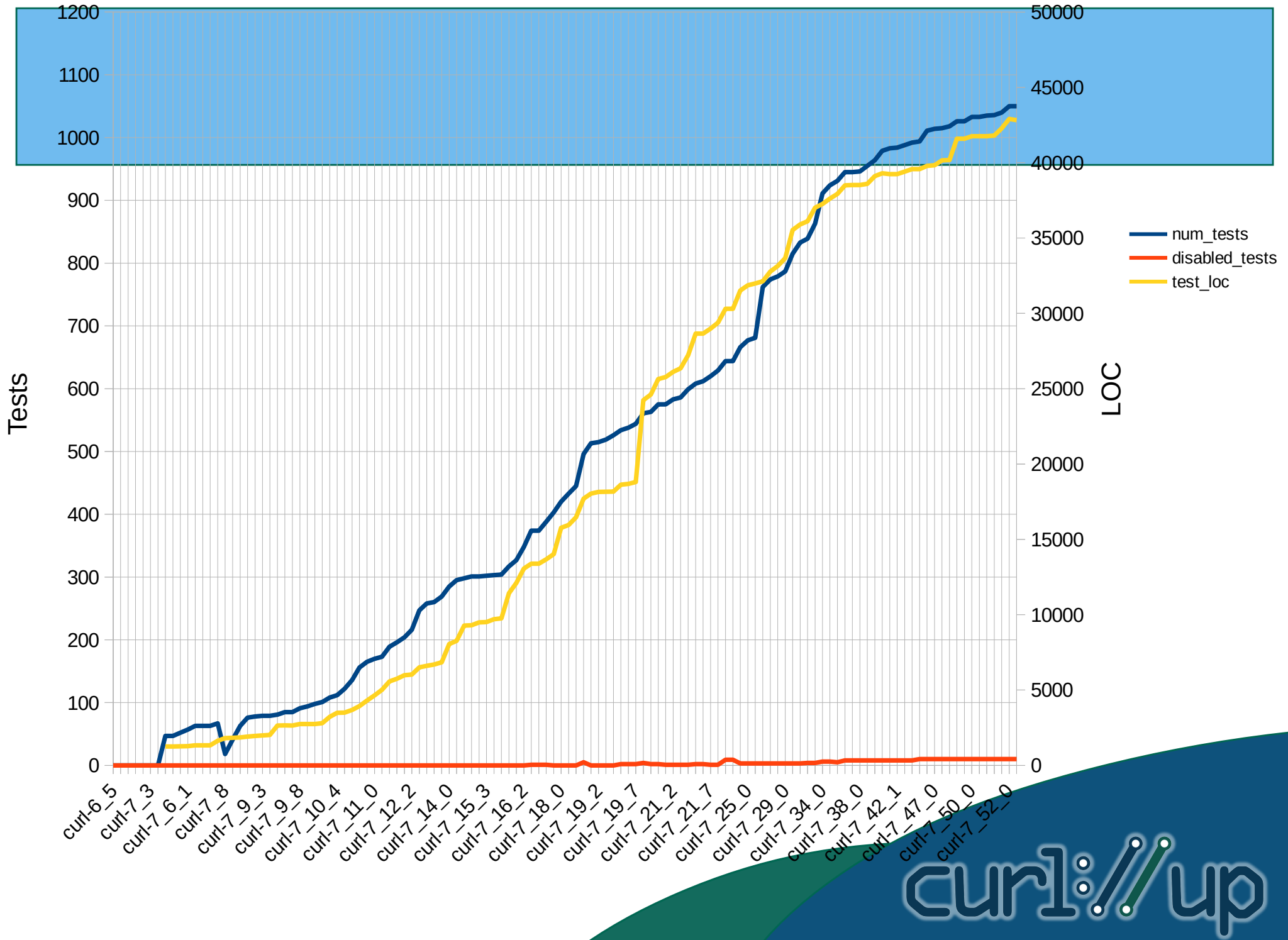
Dan Fandrich  
March 2017



# What is it?

- Takes inputs, runs commands, generates outputs
- Validates expected behaviour
- Bug fixed → New test





# How Effective?

- 7.16.3 (2007) with 355\* of 373 tests
  - 84% function coverage
  - 71% line coverage
  - 67% block coverage
- \* Missing Kerberos, more?

# How Effective?

- 7.52.1 (2017) with 1008\* of 1040 tests
  - 87% function coverage
  - 73% line coverage
  - 70% block coverage
- \* Missing HTTP/2, Kerberos, IDN, RTMP, Metalink, PSL
- Unit tests likely not counted

# Code Churn

- 21355 commits
- 3498 bug fixes
- 727 named changes
- 74 documented platforms
- 1050 tests

\* from curl 6.5 through 7.52.1



# Test Cases

- Single file
- XML-like format
- Sometimes C driver or perl script



# Test File Format

- `<testcase>`
  - `<info>...</info>`
  - `<reply>...</reply>`
  - `<client>...</client>`
  - `<verify>...</verify>`
- `</testcase>`

# Example: Test 1

```
<testcase>  
# The first test  
<info>  
<keywords>  
HTTP  
HTTP GET  
</keywords>  
</info>
```

```
<reply>  
<data>  
HTTP/1.1 200 OK  
Date: Thu, 09 Nov 2010 14:49:00 GMT  
Last-Modified: Tue, 13 Jun 2000 12:10:00 GMT  
Content-Length: 6  
Connection: close  
Content-Type: text/html
```

```
-foo-  
</data>  
</reply>
```

```
<client>  
<server>  
http  
</server>  
<name>  
HTTP GET  
</name>  
  
<command>  
http://%HOSTIP:%HTTPPORT/1  
</command>  
</client>
```

```
<verify>  
<strip>  
^User-Agent:.*  
</strip>  
<protocol>  
GET /1 HTTP/1.1  
Host: %HOSTIP:%HTTPPORT  
Pragma: no-cache  
Accept: */*  
  
</protocol>  
</verify>  
</testcase>
```

# Running the Tests

- `make test`
- `cd tests; ./runtests.pl`
- A few examples
  - `./runtests.pl -v -t 573 to 575`
  - `./runtests.pl -n -k 'HTTP PUT' !HTTPS`
  - `make test TEST_Q='-a -p -n !flaky'`

# Test Flow

- Select test number
- Read test configuration
- Start needed server(s)
- Write <servercmd> into file
- Write <file> into file
- Run <precheck>
- Run <command> or <tool>
- Run <postcheck>
- Verify with <verify>

# Server Test Flow

- Receives connection
- Reads <servercmd> from file (if it exists)
- Receives request from curl over socket
- Loads testNNNN configuration file
- Responds accordingly to request

# Types of Tests

- Normal curl command-line
- libtest—C linked to libcurl
- unittest—C statically linked to libcurl
- Perl script

# Sample libtest: test1530

test1530

lib1530.c

Uses callbacks

# Sample libtest: test556

test556

lib556.c

Tests functions used by command-line tool



# Sample unittest: test1602

unit1602.c

Internal unit tests

# Sample perl test: test1140

test1140

Test written in perl

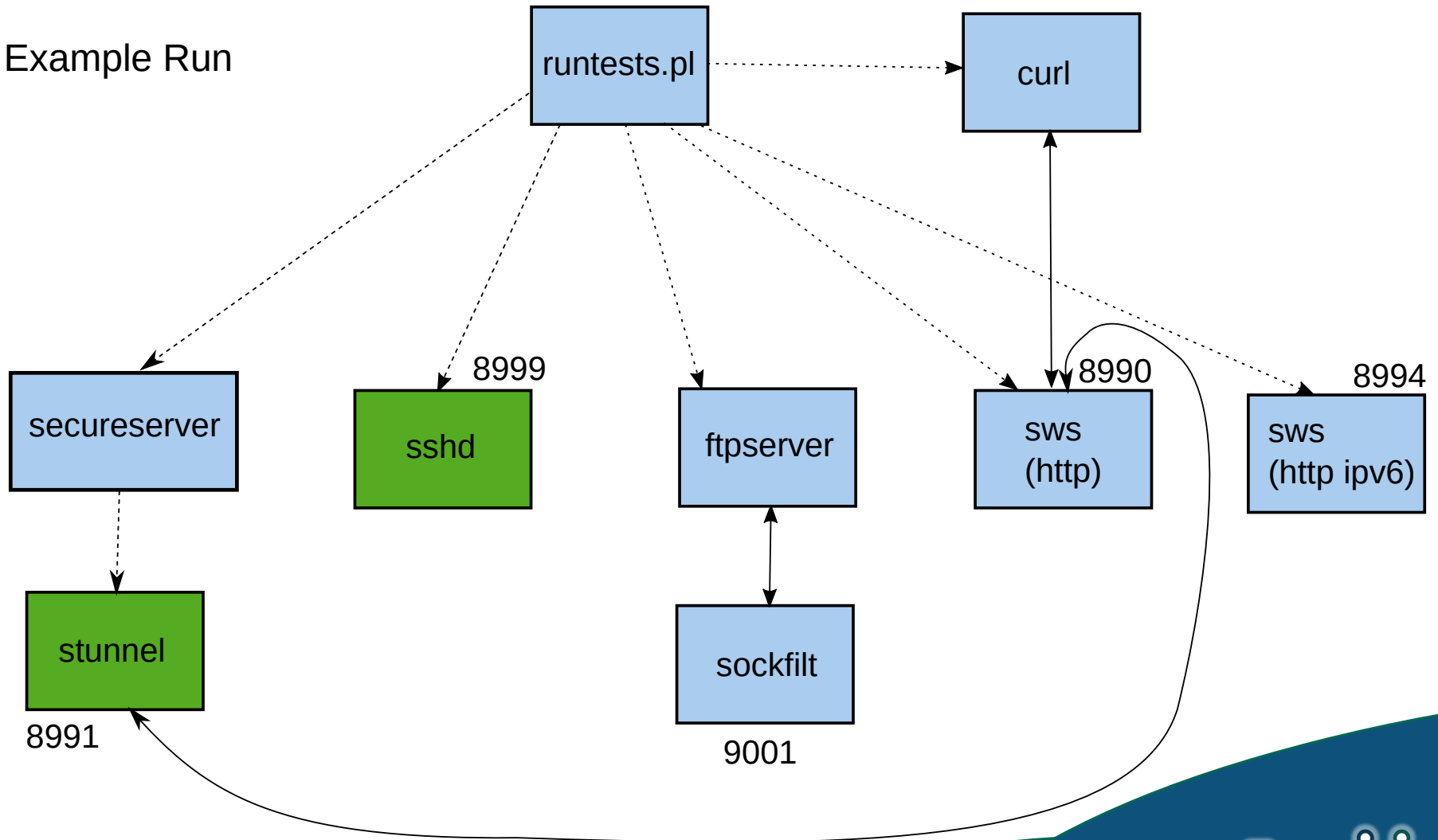
# Sample normal test: test888

test888

Test using <precheck>

# Test Suite Process Architecture

Example Run



# More Tricks

- gdb support
- memory leak detection
- torture tests
- Valgrind support
- keyword filters (and negations)
- test ranges
- permanently-disabled tests

# Potential Future Work

- Speed-ups
  - parallel testing, by protocol
- Generic test server
- Limitations, untested areas, untested protocols
- Get rid of DISABLED tests
- Move autotools tests into test suite
- Switch to true XML
- Docker image registry
- Better UI tracking features

# Questions

This presentation is © 2017 Daniel Fandrich and available under the [CC BY 4.0](https://creativecommons.org/licenses/by/4.0/) license

