Testsuite Validator

Isolated test-suite execution and coverage measurement

Thomas Lemberger
Joint work with Dirk Beyer

LMU Munich, Germany

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

SoSy-Lab Software Systems

The bug-o-rama trophy case

Yeah, it finds bugs. I am focusing chiefly on development and have not been running the fuzzer at a scale, but here are some of the notable vulnerabilities and other uniquely interesting bugs that are attributable to AFL (in large part thanks to the work done by other users):

IJG jpeg 1	libjpeg-turbo 1 2	libpng 1
libtiff 1 2 3 4 5	mozjpeg 1	PHP 1 2 3 4 5 6 7 8
Mozilla Firefox 1 2 3 4	Internet Explorer 1 2 3 4	Apple Safari 1
Adobe Flash / PCRE 1 2 3 4 5 6 7	sqlite 1 2 3 4...	OpenSSL 1 2 3 4 5 6 7
LibreOffice 1 2 3 4	poppler 1 2...	freetype 1 2
GnuTLS 1	GnuPG 1 2 3 4	OpenSSH 1 2 3 4 5
PuTTY 1 2	ntpd 1 2	nginx 1 2 3
bash (post-Shellshock) 1 2	tcpdump 1 2 3 4 5 6 7 8 9	JavaScriptCore 1 2 3 4
pdfium 1 2	ffmpeg 1 2 3 4 5	libmatroska 1
libarchive 1 2 3 4 5 6 ...	wireshark 1 2 3	ImageMagick 1 2 3 4 5 6 7 8 9 ...
BIND 1 2 3 ...	AFL-fuzz	lcms 1

About Us
- About Center
- Our Team
- News
- Partners
- Contacts

Projects
- Linux Kernel Space Verification
- LSB Infrastructure
- Testing Technologies
- Tests and Frameworks
- Portability Tools

Results
- Contribution
  - Problems in Linux Kernel
  - Problems in Libraries
  - Problems in Standards
- Publications
- Events

Problems in Linux Kernel

This section contains information about problems in Linux kernel found within Linux Driver Verification program as well as within KEDR and Linux File System Verification projects.

Click on a problem number for detailed description. Click on a column header to change the sorting order.

No.	Type	Brief	Added on	Accepted	Status
L0296	Crash	ASoC: samsung: i2s: Null pointer dereference on samsung_i2s_remove	2017-09-18	https://patchwork.kernel.org/patch/9897495/ commit	Fixed in kernel 4.14-rc4
L0297	Crash	ASoC: samsung: i2s: Null pointer dereference on samsung_i2s_remove	2017-09-18	https://patchwork.kernel.org/patch/9897495/ commit	Fixed in kernel 4.15-rc1
L0295	Crash	serial: sccnxp: sccnxp_probe() returns zero, while device is not properly initialized	2017-09-18	https://patchwork.kernel.org/patch/9935805/ commit	Fixed in kernel 4.14-rc4
L0294	Contradiction	Input: ucb1400_ts - fix suspend and resume handling	2017-08-31	https://patchwork.kernel.org/patch/9933247/ commit	Fixed in kernel 4.14-rc1
L0293	Data race	dmaengine: rcar-dmac: initialize all data before registering IRQ handler	2017-08-28	https://patchwork.kernel.org/patch/9911633/ commit	Fixed in kernel 4.14-rc1
L0292	Use after free	video: fbdev: udlfb: Fix use after free on dlfb_usb_probe error path	2017-08-28	https://patchwork.kernel.org/patch/9895789/ commit	Fixed in kernel 4.14-rc1
L0291	Contradiction	USB: Gadget core: fix inconsistency in the interface usb_add_gadget_udc_release()	2017	https://patchwork.kernel.org/patch/9906907/	Fixed in kernel 4.14-rc1

LDV

afl-generated, minimized image test sets (partial)

These very compact, synthetic corpora were generated with afl-fuzz for some of the image formats supported in modern web browsers. They exercise a remarkable variety of features in common image parsers and are a superior starting point for manual testing or targeted fuzzing work. The test cases are selected for optimal edge coverage and a wide range of coarse hit counts for every branch, as culled with *afl-cmin*. There are also **-edges-only* variants that do not factor in hit counts.

Format	Parsing library	Instrumented tool	Browsers	Preview link	S
JPEG #1	IJG jpeg9a	djpeg	All	click here	I
JPEG #2	libjpeg-turbo 1.3.1	djpeg	All	click here	I
GIF #1	giflib 5.1	gif2rgb[1]	All	click here	I
GIF #2	ImageMagick 6.8.9	convert	All	click here	I
PNG	libpng 1.6.16	readpng	All	click here	I
BMP	ImageMagick 6.8.9	convert	All	click here	I
ICO	ImageMagick 6.8.9	convert	All	click here	I
WebP	libwebp 0.4.2	dwebp	Chrome	click here	I
TIFF	libtiff CVS 2014/12/24	tiff2rgba[1]	IE, Safari	click here	I
JPEG XR	jxrlib 1.1	JxrDecApp[1]	IE	click here	I

[1] With some ad-hoc security fixes incorporated into the utility.

[2] Due to the sheer number of exploitable bugs that allow the fuzzer to jump to arbitrary addresses.

You can also grab a **downloadable archive** containing all of the above.

Note that some of this may crash your browser or make it use up 100% of CPU time (and let's not even mention trying to open this in any desktop software).

Additional sets are probably coming in the near future. This may include:

afl-generated, minimized image test sets (partial)

These very compact, synthetic corpora were generated with afl-fuzz for some of the image formats supported in modern web browsers. They exercise a remarkable variety of features in common image parsers and are a superior starting point for manual testing or targeted fuzzing work. The test cases are selected for optimal edge coverage and a wide range of coarse hit counts for every branch, as culled with *afl-cmin*. There are also **-edges-only* variants that do not factor in hit counts.

Format	Parsing library	Instrumented tool	Browsers	Preview link	S
JPEG #1	IJG jpeg9a	djpeg	All	click here	I
JPEG #2	libjpeg-turbo 1.3.1	djpeg	All	click here	I
GIF #1	giflib 5.1	gif2rgb[1]	All	click here	I
GIF #2				re	I
PNG				re	I
BMP				re	I
ICO	ImageMagick 6.8.9	convert	All	click here	I
WebP	libwebp 0.4.2	dwebp	Chrome	click here	I
TIFF	libtiff CVS 2014/12/24	tiff2rgba[1]	IE, Safari	click here	I
JPEG XR	jxrlib 1.1	JxrDecApp[1]	IE	click here	I

Note that some of this may crash your browser or make it use up 100% of CPU time (and let's not even mention trying to open this in any desktop software).

[1] With some ad-hoc security fixes incorporated into the utility.

[2] Due to the sheer number of exploitable bugs that allow the fuzzer to jump to arbitrary addresses.

You can also grab a **downloadable archive** containing all of the above.

Note that some of this may crash your browser or make it use up 100% of CPU time (and let's not even mention trying to open this in any desktop software).

Additional sets are probably coming in the near future. This may include:

We introduce...

Testsuite-Validator

- Isolated test execution
- Coverage measurements
- Test-suite reduction (`WIP`)

- Simple, xml-based format for test specification

Concrete Example

```
#include <stdio.h>
#include <unistd.h>
extern char __VERIFIER_nondet_char();

int main() {
  char x = __VERIFIER_nondet_char();
  if (x == 'a') {
    while (1)
      fork();
  } else {
    remove("important.txt");
    if (access("important.txt", F_OK) != -1) {
      return 1;
    }
  }
}
```

Isolated test execution

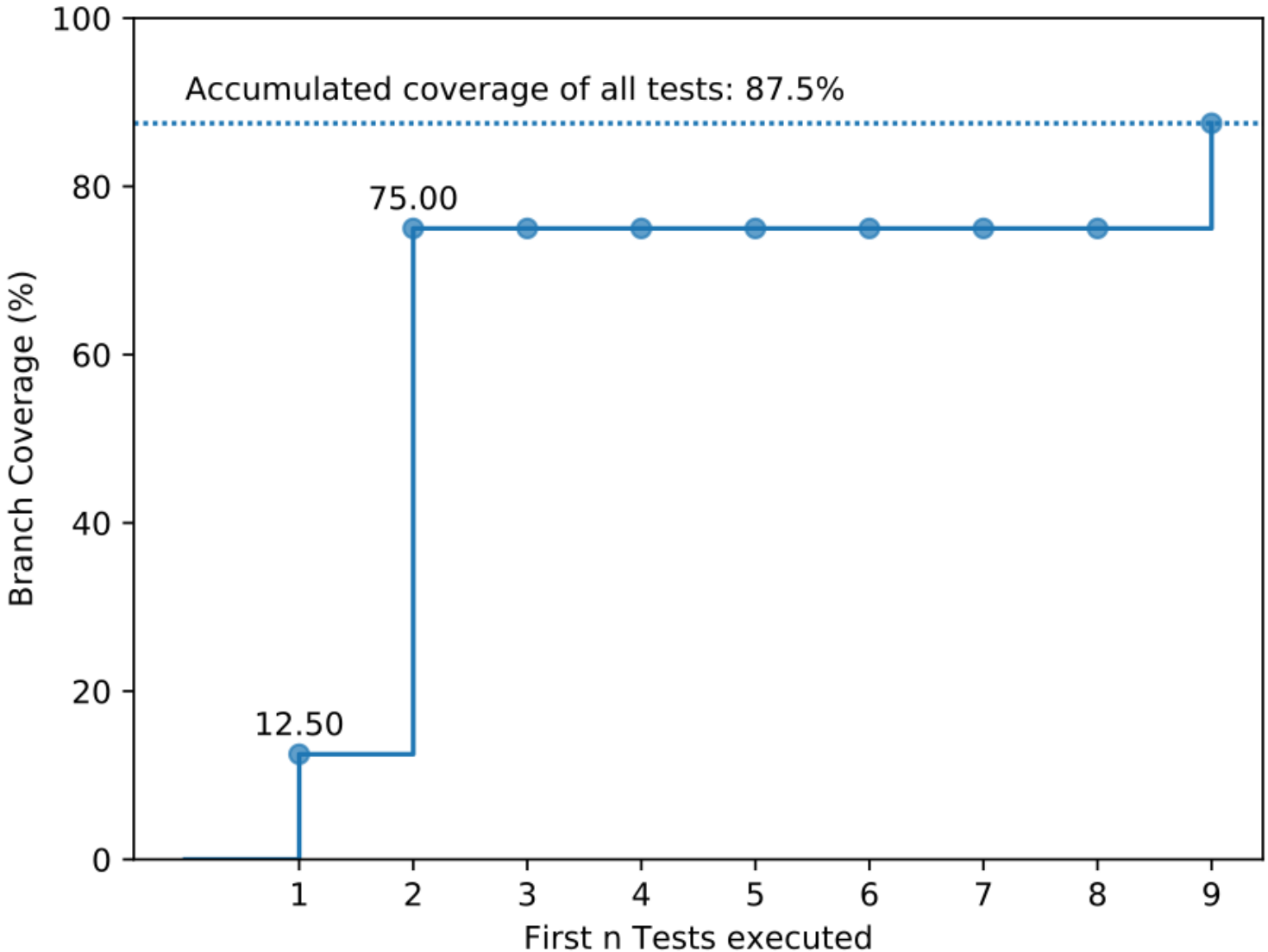
- Malicious influences:
  - Resource exhaustion
  - File system modifications
  - Spoofing
- Solutions:
  - CGroups
  - Containers
- Both provided by BenchExec

Coverage measurements

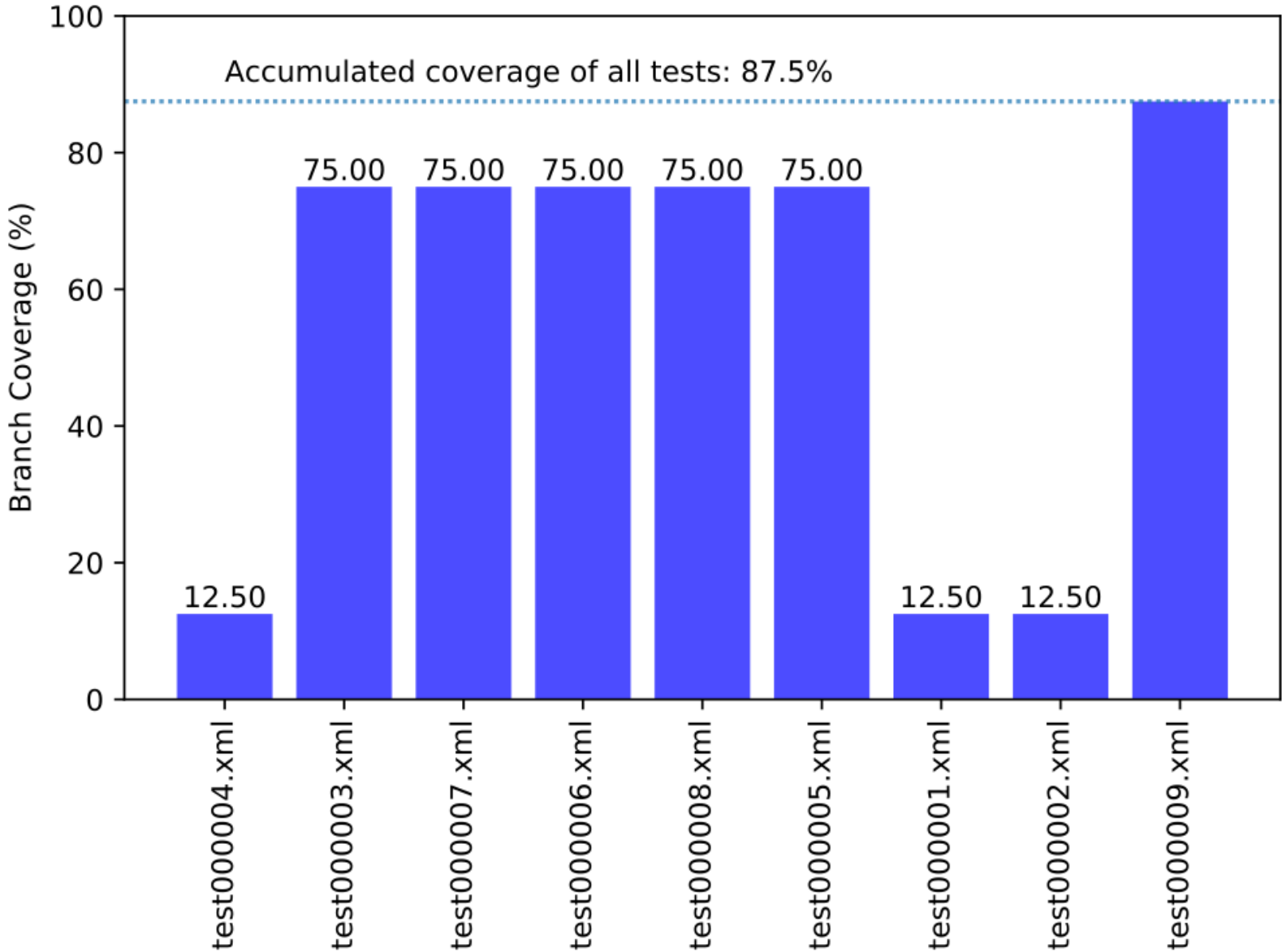
- Measurement through `lcov` and `llvm-cov`
  - Provides line- and condition-coverage
- Manual branch coverage computation based on `lcov` data

- Produced data:
  - Individual test coverage
  - Accumulated test coverage (after each execution)

Coverage Plots: Accumulated Coverage



Coverage Plots: Individual Coverage



Test-suite Reduction

- Naive approach only

- Test does not increase accumulated coverage $\Rightarrow$ remove
- Order of tests determined by test file names

Test Format

- Two components:
  1. `metadata.xml`
  2. Test-XMLs
- Handled as zip archives
- Builder library provided in python:

https://gitlab.com/sosy-lab/software/test-format/tree/master/python_modules/tfbuilder

Metadata

```
<?xml version="1.0"?>
<!DOCTYPE test-metadata PUBLIC "+//IDN sosy-lab.org//DTD test-format te
<test-metadata>
  <sourcecodelang>C</sourcecodelang>
  <producer>Testsuite Validator v2.0</producer>
  <specification>CHECK(FQL(cover EDGES(@CONDITIONEDGE)))</specificatio
  <programfile>example.c</programfile>
  <programhash>eeecda9cbf27c43c9017fa00dd900c19a5ec18d46303f59a6e0357db78
  <entryfunction>main</entryfunction>
  <architecture>32bit</architecture>
  <inputtestsuitefile>original-suite.zip</inputtestsuitefile>
  <inputtestsuitehash>11911d658dcfbf8501390bf0faa96eb193b11bb1</inputtestsui
  <creationtime>2019-06-19T14:17:34Z</creationtime>
</test-metadata>
```

Test Case

```
<?xml version="1.0"?>
<!DOCTYPE testcase PUBLIC "+//IDN sosy-lab.org//DTD test-format testcase
<testcase>
  <input type='char'>'b'</input>
  <input  variable='x'>10</input>
  <input type='int'>0x0f</input>
  <input>Hello World</input>
</testcase>
```

Usage

- Used in Test-Comp'19
- 9 participants support test format:

Future Work

- Fix branch coverage
- Speed up test execution with mentioned features
- Different test-suite reduction strategies