OSS-Fuzz：容器和云计算在模糊测试中的应用

韩寒
<hhan@redhat.com>
虚拟化测试工程师
2017-10-22

AGENDA

Fuzzing[1]

Container in testing

OSS-Fuzz[2]

How to contribute to oss-fuzz

Demo : add project to oss-fuzz

Open source software severe vulnerabilities

2017

- WPA KRACK[3] - *wpa_supplicant/hostapd*
- Bluetooth BlueBorne[4] - *kernel*
- The stack slash[5] - *sudo,at,ld...etc*

2016

- Dirty cow[6] - *kernel*

2015

- GHOST[7] - *glibc*

2014

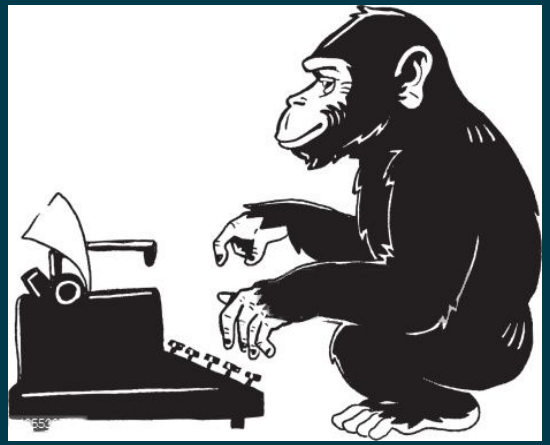
- Shellshock[8] - *bash*
- Heartbleed[9] - *openssl*

What could we do to contribute to OSS security?

Fuzzing（模糊测试）

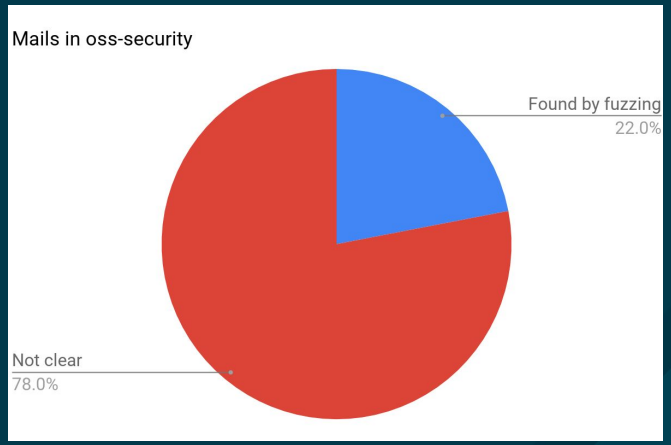
What is fuzzing

Fuzzing is an **automated** software **testing** technique that involves providing **invalid, unexpected,** or **random** inputs to a program.



Simple but powerful

Near **22%**(58/264 Jul. to Sep.) vulnerabilities are claimed found by fuzzing in **oss-security**[10] mail list.



Fuzzing
How to fuzzing

Advantages

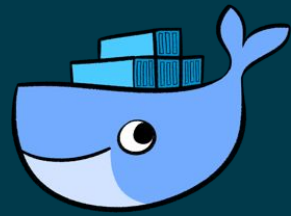
- Automatically executed.
- Powerful to find security issue.
- Simple to start.

Challenges

- Fuzzing ENVs(fuzzers)
- Execution ENV isolated.
- Resource control

Fuzzing in container

Advantages of container



- Fast & easy deployment
- Lightweight Virtualization
- Cgroup resource control
- Quick boot and shut-off

- Fuzzing ENVs(fuzzers)
- Execution ENV isolated.
- Resource control

Limitations: cannot provide complex environment for **system** or **integration** test.

What if
fuzzing + container + cloud ?

OSS-FUZZ

What is oss-fuzzing

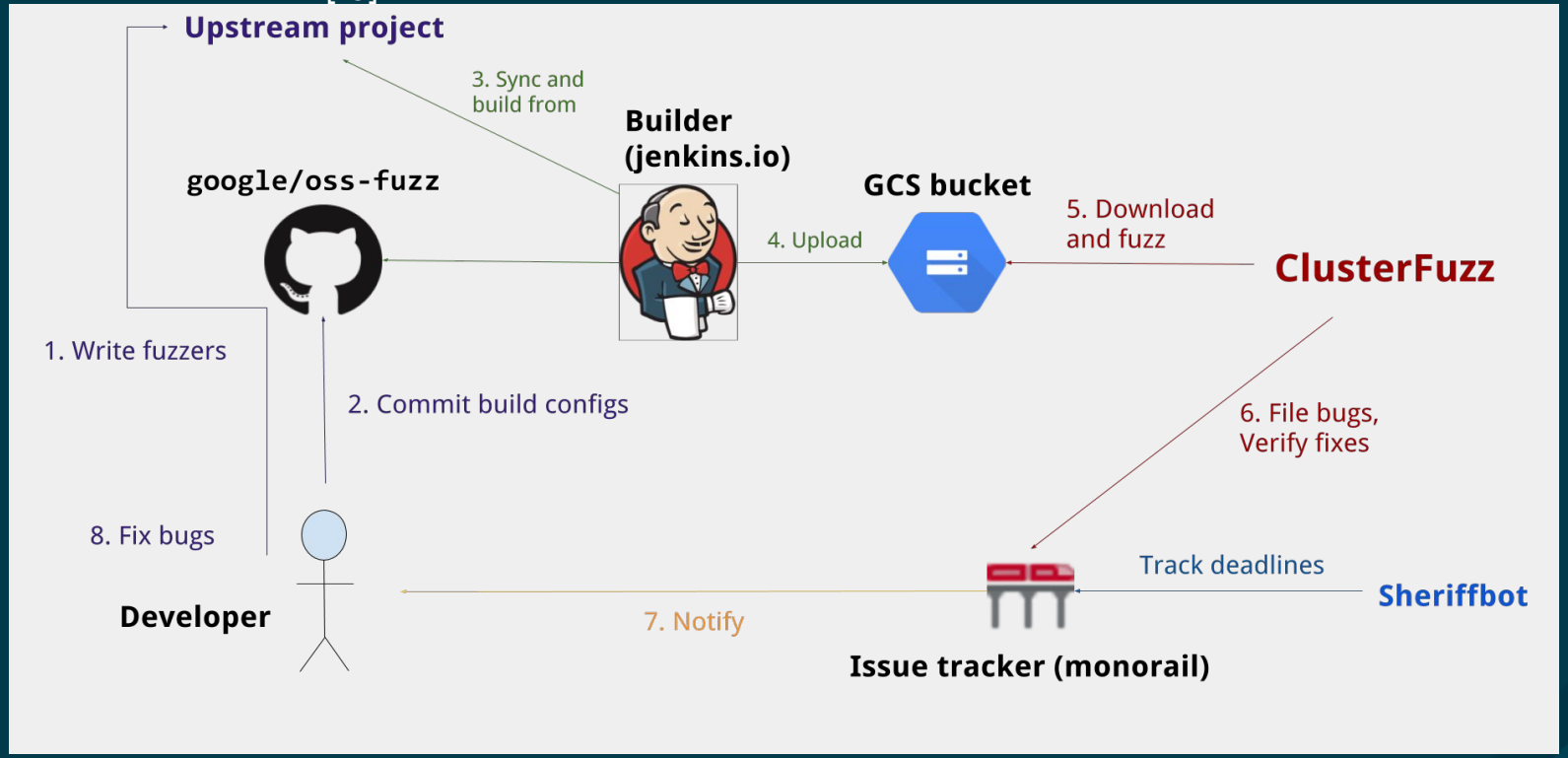
Continuous **fuzzing framework** for open source software in **C/C++**, aims to make common open source software more **secure and stable** by combining **modern fuzzing** techniques and **scalable distributed execution**.

Features:

- Multiple fuzzer engines: **libFuzzer**[11] , **afl**[12], **honggfuzz**[13].
- Multiple sanitizers: **address**[14]/**memory**[15]/**undefined**[16].
- Fuzzing execution and reporting by **ClusterFuzz**[17].
- Fuzzing under **docker**.

OSS-FUZZ

Process Overview[18]



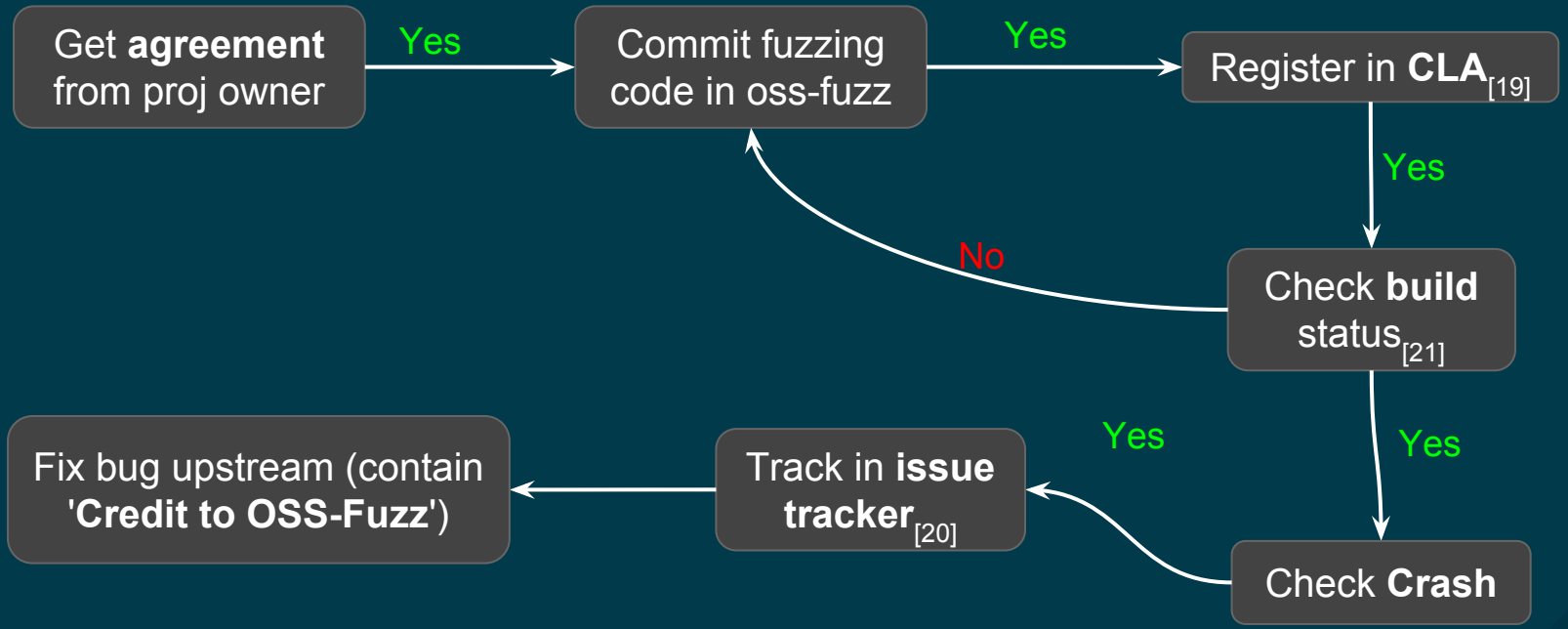
OSS-FUZZ
Achievement

From Dec,2016 to Sep,2017:

- **92** open source software projects integrated.
- **1843** non-security bugs found and fixed.
- **585** security bugs found and fixed.

Contribute to oss-fuzz

Developer workflow



Demo : add project to oss-fuzz
Code structure

Project code is located in *projects/<project_name>*

Mandatory files:

- **project.yaml**: project metadata file.
- **Dockerfile**: Provides docker env preparation.
- **build.sh**: Build project and its fuzzer.

Optional files(Or provide in upstream repository):

- **Fuzzer.cc**: Provides fuzzer code.

Demo : add project to oss-fuzz

Project metadata file

Attributes:

- **homepage:** Project's homepage.
- **primary_contact, auto_ccs:** Primary contact and CCs list.
- **sanitizers (optional)** - List of sanitizers to use.
  - address: detect memory issues like **mem-leak**, **buffer overflow**, **use-after-free**, **double-free**
  - memory: detect **uninitialized reads**.
  - undefined: detect **misaligned or null pointer**, **int or float overflow**

Example:

```
homepage: "https://curl.haxx.se/"
primary_contact: "daniel@haxx.se"
auto_ccs:
  - "daniel.haxx@gmail.com"
sanitizers:
  - address
```

Demo : add project to oss-fuzz

Docker env

Steps:

1. Pull image with clang toolchain and fuzzer engines.
2. Set **maintainer** for this file.
3. Install **required packages** for building project.
4. Pull upstream codes.
5. (Optional)Copy build script and fuzzer codes to container.

Example:

```
FROM gcr.io/oss-fuzz-base/base-builder
MAINTAINER YOUR_EMAIL
RUN apt-get update && apt-get install -y ...
RUN git clone <git_url> <checkout_dir>
WORKDIR <checkout_dir>
COPY build.sh fuzzer.cc $SRC/
```

Demo : add project to oss-fuzz
Fuzzer

Build fuzzing code with fuzzing engine(e.g libFuzzer)

Code example(libFuzzer)

```
bool FuzzMe(const uint8_t *Data, size_t DataSize) {
  return DataSize >= 3 &&
      Data[0] == 'F' &&
      Data[1] == 'U' &&
      Data[2] == 'Z' &&
      Data[3] == 'Z';  // :-<
}
extern "C" int LLVMFuzzerTestOneInput(const uint8_t *Data, size_t Size) {
  FuzzMe(Data, Size);
  return 0;
}
```

Demo : add project to oss-fuzz
Build script

To build project and fuzzer.

build.sh example:

```
#!/bin/bash -eu
# configure scripts usually use correct environment variables.
./configure --enable-static  # build static libraries
make clean
make -j$(nproc) all
$CXX $CXXFLAGS -std=c++11 -I src/  $SRC/parse_fuzzer.cc -o
OUT/parse_fuzzer \  -lFuzzingEngine .libs/libXX.a # build fuzzer with static libraries
```

Demo : add project to oss-fuzz

Testing locally(in oss-fuzz repo dir)

- Build docker image(you may need to set **proxy** first)

python infra/helper.py build_image $PROJECT_NAME

- Build fuzzer

python infra/helper.py build_fuzzers --sanitizer <address/memory/undefined> $PROJECT_NAME

- Run fuzzer

python infra/helper.py run_fuzzer $PROJECT_NAME <fuzz_target>

- Run coverage

python infra/helper.py coverage $PROJECT_NAME <fuzz_target>

Demo : add project to oss-fuzz
ClusterFuzz

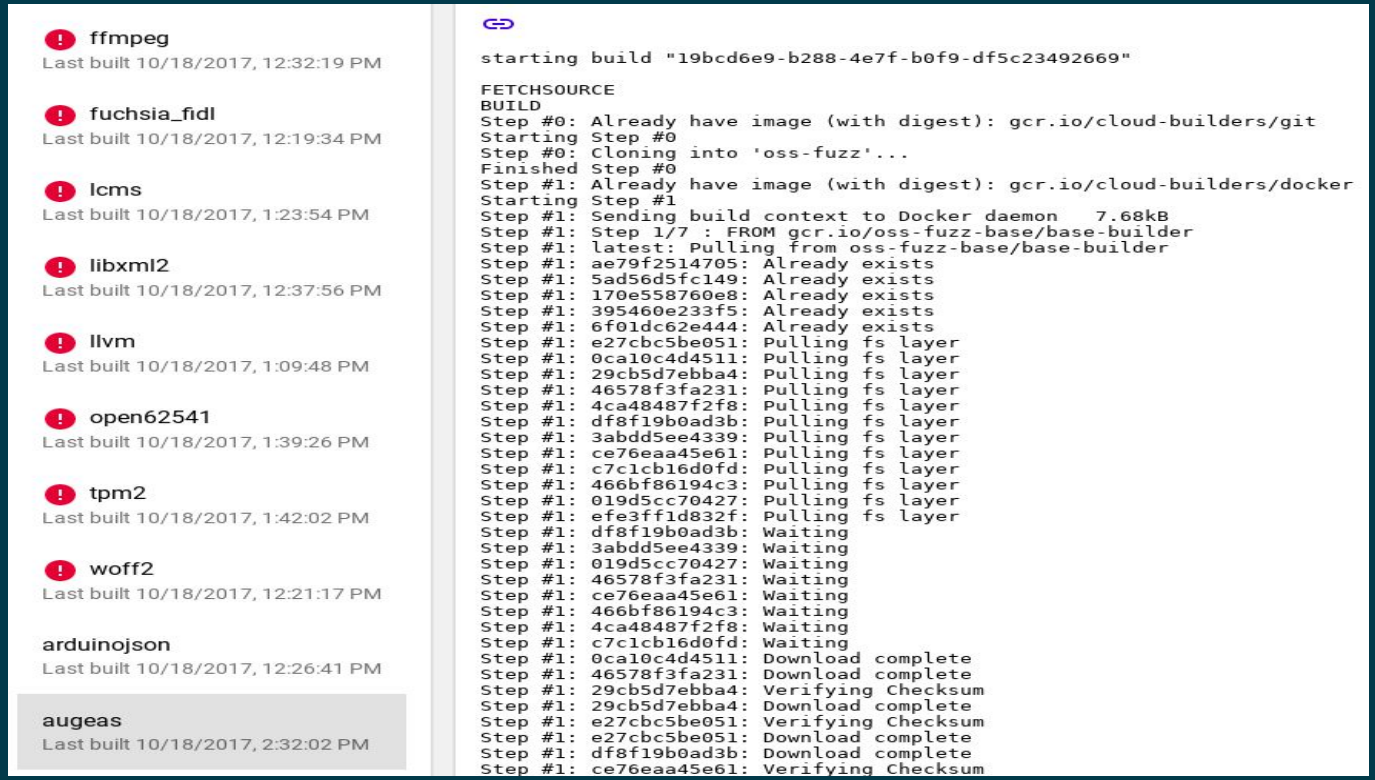
ClusterFuzz is the **distributed** fuzzing execution and reporting infrastructure behind OSS-Fuzz.

Web interface of ClusterFuzz:

- Testcase reports
- Fuzzer stats
- Coverage reports
- Performance analyzer
- Crash stats

Demo : add project to oss-fuzz

ClusterFuzz: build status



Demo : add project to oss-fuzz
Cluster fuzz: Crash report

CREATE ISSUE | REDO TASK

OVERVIEW

Crash State:	NULL				
Crash Type:	Use-of-uninitialized-value	Fuzzer:	libFuzzer_augeas_escape_name_fuzzer	Security:	**YES (Medium)**
Crash Address:	---	Job Type:	libfuzzer_msan_augeas	Reproducible:	**NO**
Issue:	None	Platform:	linux	Fixed:	NA
Created:	Wed, Oct 18, 2017, 8:36 PM	Sanitizer:	memory (MSAN)		
Project:	augeas	Deletion:	Will be auto-deleted on 10/25/2017 if flaky crash no longer seen		

Minimized Testcase: NA Unminimized Testcase: (0 B) Re-upload testcase:

You can reproduce this crash painlessly with our reproduce tool. For Googlers, install the required libraries and run `prodaccess && /google/data/ro/teams/clusterfuzz-tools/releases/clusterfuzz reproduce 6442776221712384` . For non-Googlers, see the installation section. Report any issues at clusterfuzz-dev@chromium.org.

FIXED REVISION RANGE	REGRESSION REVISION RANGE
NA	NA

Demo : add project to oss-fuzz

ClusterFuzz: Code coverage

File	Coverage
Files with zero coverage are not shown.	
/src/augeas/src/augeas.c	027%
/src/augeas/src/builtin.c	007%
/src/augeas/src/errcode.c	015%
/src/augeas/src/info.c	022%
/src/augeas/src/internal.c	006%
/src/augeas/src/internal.h	029%
/src/augeas/src/memory.c	073%
/src/augeas/src/pathx.c	044%
/src/augeas/src/ref.c	075%
/src/augeas/src/syntax.c	021%
/src/augeas_escape_name_fuzzer.cc	090%
/usr/local/include/c++/v1/memory	100%
/usr/local/include/c++/v1/string	100%

Demo : add project to oss-fuzz
ClusterFuzz:Performance

date	perf_report	logs	tests_executed	total_crashes	new_crashes	known_crashes	edge_cov
Oct 18, 2017	Performance	Logs	676,631,973	0	0	0	26.99% (1561/5783)

func_cov	cov_report	corpus_sizeSize of the minimized corpus generated based on code coverage (number of testcases and total size on disk)	corpus_backup	avg_exec_per_sec	new_tests_added	slowest_test_time_sec	peak_mem_mb	oom_count
42.19% (154/365)	Coverage	501 (69 KB)	Download	392.178	15,616	0	135	0

Demo : add project to oss-fuzz

Issue tracker

	ID ▾	Type ▾	Component ▾	Status ▾	Proj ▾	Reported ▾	Owner ▾	Summary + Labels ▾
☆	112	Bug	----	New	libchewing	----	----	libchewing: record ClusterFuzz Reproducible
☆	209	Bug	----	New	libchewing	----	----	libchewing: pgdata->chiSymbolBufLen >= pgdata->chiSymbolCursor ClusterFuzz Reproducible
☆	226	Bug	----	New	expat	----	mmoroz@google.com	Direct-leak in addBinding ClusterFuzz Reproducible
☆	359	Bug	----	New	libarchive	----	----	libarchive: Out-of-memory in libarchive_fuzzer ClusterFuzz Reproducible
☆	373	Bug	----	New	libchewing	----	----	libchewing: pgdata->choiceInfo.oldChiSymbolCursor <= pgdata->chiSymbolBufLen ClusterFuzz Reproducible
☆	570	Bug	----	New	libchewing	----	----	libchewing: chiSymbolCursor < ARRAY_SIZE(pgdata->preeditBuf) ClusterFuzz Reproducible
☆	572	Bug	----	New	libprotobuf-mutator	----	----	libprotobuf-mutator: Direct-leak in xz_error ClusterFuzz Reproducible
☆	589	Bug	----	New	grpc	----	----	grpc: Undefined-shift in gpr_stack_lockfree_push ClusterFuzz Reproducible
☆	602	Bug	----	New	libass	----	----	libass: Integer-overflow in parse_tag ClusterFuzz Reproducible
☆	603	Bug	----	New	libass	----	----	libass: Integer-overflow in ass_lazy_track_init ClusterFuzz Reproducible
☆	613	Bug	----	New	tpm2	----	----	tpm2: Undefined-shift in ObjectAllocateSlot ClusterFuzz Reproducible
☆	621	Bug	----	New	tpm2	----	----	tpm2: Undefined-shift in TPMI_DH_CONTEXT_Unmarshal ClusterFuzz Reproducible
☆	623	Bug	----	New	tpm2	----	----	tpm2: Undefined-shift in TPMI_DH_CONTEXT_Unmarshal ClusterFuzz Reproducible
☆	624	Bug	----	New	libass	----	----	libass: Integer-overflow in parse_tag ClusterFuzz Reproducible
☆	632	Bug	----	New	libtsm	----	----	libtsm: Crash in tsm_screen_tab_left ClusterFuzz Reproducible

Reference

1. **Fuzzing**: https://en.wikipedia.org/wiki/Fuzzing
2. **oss-fuzz**: https://github.com/google/oss-fuzz
3. **Krack**: https://www.krackattacks.com/
4. **BlueBorne**: https://www.armis.com/blueborne/
5. **Stack slash**: https://www.qualys.com/2017/06/19/stack-clash/stack-clash.txt
6. **Dirty cow**: https://dirtycow.ninja/
7. **GHOST**: https://access.redhat.com/articles/1332213
8. **Shellshock**: https://en.wikipedia.org/wiki/Shellshock_(software_bug)
9. **Heartbleed**: http://heartbleed.com/
10. **oss-security**: http://www.openwall.com/lists/oss-security/
11. **libFuzzer**: https://llvm.org/docs/LibFuzzer.html
12. **Afl**: http://lcamtuf.coredump.cx/afl/
13. **honggfuzz**: https://github.com/google/honggfuzz
14. **AddressSanitizer**: https://clang.llvm.org/docs/AddressSanitizer.html
15. **MemorySanitizer**: https://clang.llvm.org/docs/MemorySanitizer.html
16. **UndefinedBehaviorSanitizer**: https://clang.llvm.org/docs/UndefinedBehaviorSanitizer.html
17. **ClusterFuzz**: https://github.com/google/oss-fuzz/blob/master/docs/clusterfuzz.md
18. **Process Overview**: https://github.com/google/oss-fuzz#process-overview
19. **CLA(Contributor License Agreement)**: https://cla.developers.google.com/
20. **Issue tracker**: https://bugs.chromium.org/p/oss-fuzz/issues/list
21. **ClusterFuzz build status**: https://oss-fuzz-build-logs.storage.googleapis.com/index.html

Summary

- What is fuzzing
- Advantages of fuzzing with container and cloud
- What is oss-fuzz
- How to contribute to oss-fuzz

Q&A

redhat.

THANK YOU

plus.google.com/+RedHat

facebook.com/redhatinc

linkedin.com/company/red-hat

twitter.com/RedHatNews

youtube.com/user/RedHatVideos