2021 - All about

COOL days

Collabora Online

Sanitizers, fuzzing and string-vectors

By Miklos Vajna

Software Engineer Collabora Online

vmiklos@collabora.com

About Miklos

From Hungary

- More details: https://www.collaboraoffice.com/about-us/

Google Summer of Code 2010 / 2011

- Rewrite of the Writer RTF import/export

Then a full-time LibreOffice developer for SUSE

Now a contractor at Collabora

2021 - All about COOL days Collabora Online

Sanitizers

Collabora Online

ubsan, asan and others

Clang provides several sanitizers, we use two:

- UndefinedBehaviorSanitizer (detects e.g. signed integer overflow)
- AddressSanitizer (detects e.g. stack-use-after-return and heap-use-after-free)

Environment

- core.git make check already passes with these sanitizers
- Now online.git make check (c++ tests) also pass
  - Cypress?
- Use LODE as the environment, as sanitizers have lots of config options, easy to hit non-interesting problems

2021 - All about COOL days Collabora Online

Fuzzing

Collabora Online

Admin fuzzer

Tests the incoming websocket traffic of the admin console

- Simple file format: one websocket message / line
- Found 6 problems so far

```
Admin& admin = Admin::instance();
auto handler = std::make_shared<AdminSocketHandler>(&admin);

std::string input(reinterpret_cast<const char*>(data), size);
std::stringstream ss(input);
std::string line;
while (std::getline(ss, line, '\n'))
{
  std::vector<char> v(line.data(), line.data() + line.size());
  handler->handleMessage(v);
}
```

Client session fuzzer

Initially this was “the fuzzer”, i.e. the first one:

- Tests what is incoming on the websocket from editing clients
- Found 11 problems so far

Fuzzer environment

- Same as sanitizers, i.e. ubsan+asan
- online.git configure gets an --enable-fuzzers
- Only uses Online as a library, i.e. the build produces no loolwsd binary
- The fuzzer is an executable, and it has to link all Online code statically

HTTP response fuzzer

Introduced as part of the async save work

- Tests what is a reply for a HTTP request
- Found 3 problems so far

Fuzzing-as-a-service

- All 3 fuzzers run 7/24 as a Jenkins job
- They run for a week: if they don’t find anything, then they quit
  - Then pull, build, and start again
- Mail notification when they find something:
  - The server creates a reproducer (expensive)
  - A local environment can reproduce the produced crash sample (cheap)

String-vectors

Fuzzing found a pattern:

- If we have a vector of strings, it’s easy to forget checking the array bounds before accessing the nth string
- If we are at it: allocating a null-terminated string for each token shows up on profiles

Solution: StringVector

- Similar to std::vector<std::string>, but it has a single underlying string
- Tokens only have offset + length “pointers” into that
- Safe API: if we would read past the end of the array, return an empty string
- Clang AST matcher to find all uses of v[0] == “foo”

Summary

Sanitizers: to make sure tests don’t only pass by accident

- Have a tinderbox for this

Then fuzz it:

- Invent fake file formats to stress-test API that handles untrusted user input
- Do it as a CI job, so it finds badness before others do
- When the crash samples show a pattern, introduce safe APIs around unsafe ones

This makes Online a safer choice for everyone!