The Fuzzing Project
https://fuzzing-project.org/

Hanno Böck

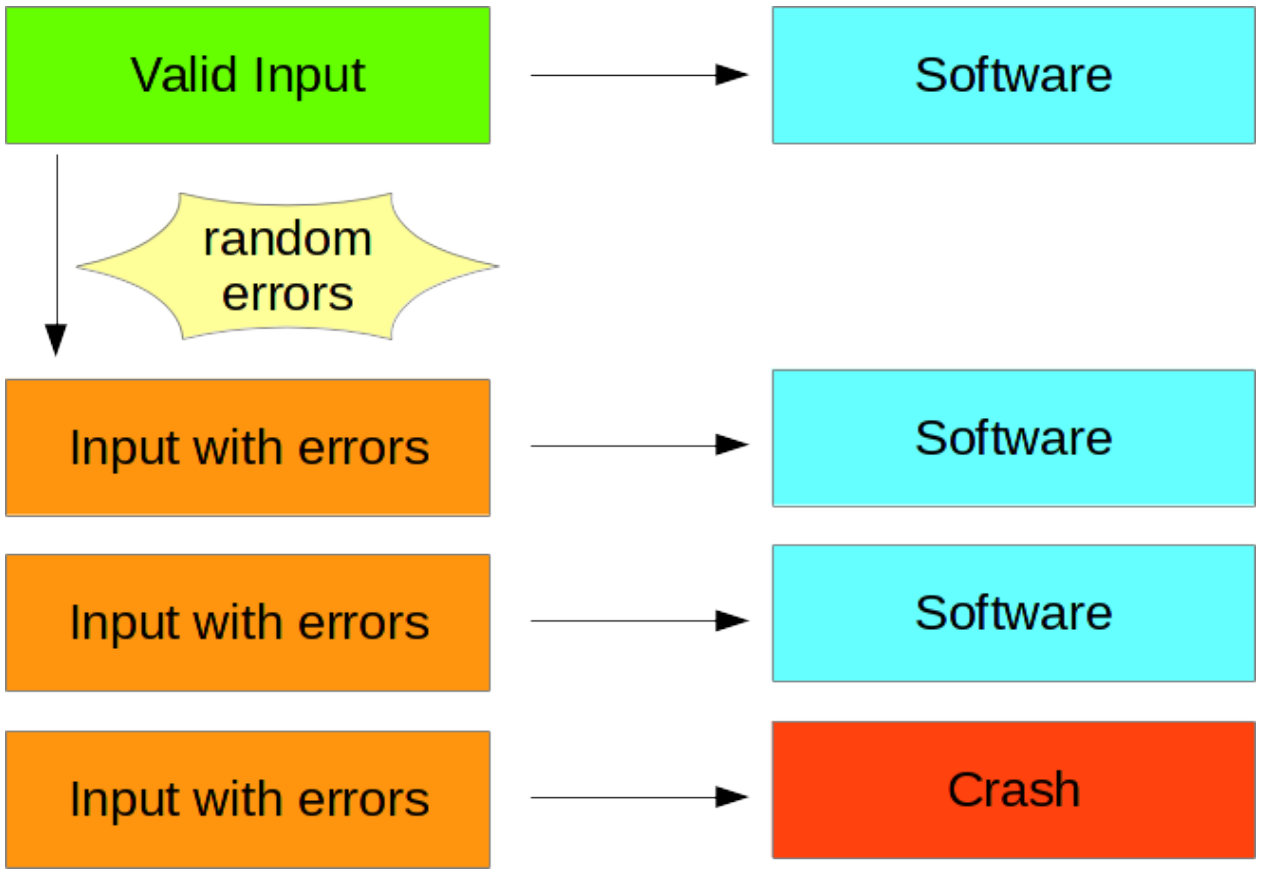
CC 0 ZERO

Motivation

- Do you use tools like strings, less, file, convert, ldd, unzip, ...?
- Would you use these tools on untrusted input?

CC 0 ZERO

Fuzzing



CC 0 ZERO

C Memory Bugs

- Buffer Overflow, Stack Overflow, Heap Overflow, Use-after-Free, Out-of-bounds, Memory Corruption, Off-by-1, ...
- Summarize: Software reads or writes the wrong memory
- Many security vulnerabilities are bugs in C memory handling

CC 0 ZERO

Invalid memory access example

```
int main() {
int a[2] = {3, 1};
int b = a[2];
}
```

CC 0 ZERO

Example: binutils

- October 2014: Michal Zalewski reports a crash in strings
- strings is part of binutils and parses executables (ELF, PE and others) - did you know that?
- Followup: Various people started fuzzing binutils (nm, ld, objdump, readelf, ...) and found hundreds of memory corruption issues - and we're still not done
- binutils 2.25: strings doesn't parse executables by default any more

CC 0 ZERO

Example: less

- less pipes input through lesspipe, a script that calls other applications depending on the filetype
- unzip, cpio, lha, antiword, catdoc, unrtf, rpm, msgunfmt, dpkg, identify (ImageMagick), cabextract, readelf (binutils!), isoinfo, ...
- Many of these tools have or had memory corruption bugs that are trivial to find via fuzzing
- less itself has unfixed memory access issues (CVE-2014-9488)

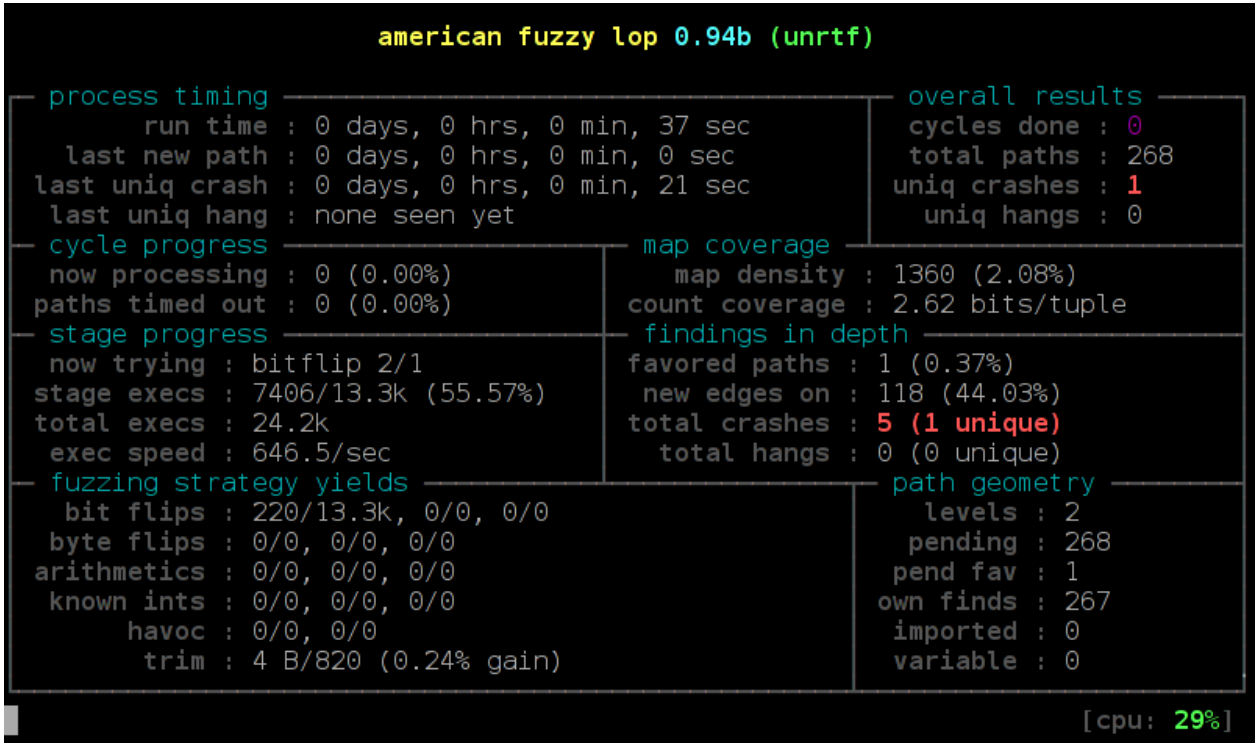
CC 0 ZERO

Let's start fuzzing

- Fuzzing finds real security vulnerabilities
- It's easy!
- If you take a random piece of software that parses complex data chances are very high that you will find crashes within minutes
- We should just fuzz everything and fix this

CC 0 ZERO

American Fuzzy Lop (afl)



CC 0 ZERO

American Fuzzy Lop (afl)

- Currently most powerful free tool for fuzzing
- Adds compile time instrumentation and identifies promising code paths
- Developed by Michal Zalewski (lcamtuf), found some of the post Shellshock Bash bugs and issues in gnupg, openssh, libjpg, libpng, ...

CC 0 ZERO

Address Sanitizer (asan)

- Not every invalid memory access causes a crash
- Addressf Sanitizer: Compile time feature to add additional bounds checks (clang, gcc - CFLAGS="-fsanitize=address")
- afl/asan combination is currently the gold standard of fuzzing

CC 0 ZERO

Make fuzzing part of development

- Ideally free software projects should integrate fuzzing into their development process
- Make software fuzzing friendly!
- Should not break with Address Sanitizer
- Provide simple command line tools with libraries to expose parsers

CC 0 ZERO

Deprecate C

- Shouldn't we deprecate C and rewrite everything in [some other programming language]?
- Answer: Moving away from C is good for new projects
- Projects like miTLS, Servo (browser engine), MirageOS are valuable
- But: We won't deprecate C any time soon

CC 0 ZERO

Fix C

- Shouldn't we use mitigations like ASLR because we can't fix all buffer overflows?
- Answer: Yes! Unfortunately state right now is sad. Most Linux distributions don't enable position independent executables by default and have weak ASLR. Better exploit mitigations (Levee) are coming.
- Exploit mitigations are either incomplete or too expensive for real applications - fixing bugs still reduces attack surface
- `http://oss-security.openwall.org/wiki/exploit-mitigation`

CC 0 ZERO

Not everything is bad!

- In most cases upstream developers were happy about reports and fixed them quickly, many start fuzzing themselves
- Many people right now flood upstream devs with fuzzing-related bug reports
- Some projects that didn't have releases for a long time were revived (unrtf, cabextract)
- bc/dc had last stable release in 2000, will soon have a new release with fixes for fuzzing-related bugs

CC 0 ZERO

Hall of shame

- less: developers didn't answer, new releases didn't fix reported issues
- poppler: several unfixed open bugs, no visible activity on them
- unzip: Public forum has information about memory corruption issues posted several years ago, unfixed in current release
- Dead projects are a problem (no development but active use - e. g. procmail)

CC 0 ZERO

The Fuzzing Project

- Tutorial for beginners (Fuzzing is easy!)
- Software list, the good and the bad
- File samples (if you want to fuzz a Microsoft Works importer and don't have an input sample at hand)

CC 0 ZERO

Takeaway messages

- Fuzzing is easy - everyone involved in software development should use it
- We have powerful free software tools (american fuzzy lop, address sanitizer)
- If your software is listed on the Fuzzing Project webpage and has no green "OK" - do something about it!

`https://fuzzing-project.org/`
`http://lcamtuf.coredump.cx/afl/`
`https://code.google.com/p/address-sanitizer/`

CC 0 ZERO