

zzuf - multiple purpose fuzzer

HA!



HA!

**YOUR SOFTWARE IS FULL OF
FAIL!!1!**

input fuzz testing

- The idea
 - randomly alter a program's input
- Expose bugs
 - user-contributed data (intarweb, e-mail)
 - file parsers, interpreters...
 - security implications more than often
- No magical solution
 - only finds simple bugs, not all of them

what zzuf does

- Input corruption
 - randomly sets or unsets bits
 - uses a random seed
- Fully automated
 - can work on the fly
 - no costly file generation
 - works with DVDs, network, `stdin`
 - reproducible behaviour
 - on consecutive runs, with different programs

how zzuf works

- Controlling `zzuf` binary
 - forks tested program
 - checks `stdout`, exit value, signals...
- `LD_PRELOAD` mechanism
 - intercepts file reading functions
 - `open()`, `read()`, `fopen()`, `fread()`...
 - plans for zlib functions, too
 - also `malloc()` to check memory usage
 - more portable than `ptrace`, kernel land...

example #1

- Using `zzuf` and `cat`

trivial use of cat

```
16/02 1:41 sam@poukram /tmp% cat readme.txt
```

```
HELLO WORLD
```

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ abcdefghijklmnopqrstuvwxyz 0123456789
```

```
16/02 1:41 sam@poukram /tmp% 
```

use cat through zzuf

```
16/02 1:42 sam@poukram /tmp% zzuf -r0.001 cat readme.txt
```

```
HELLO WORLD x
```

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ abcdefghijklmnopqrstuvwxyz 0123456789
```

```
16/02 1:42 sam@poukram /tmp% █
```

**-r0.001 means “fuzz 0.1%
of the files we read”**

fuzz more bits (1%)

```
16/02 1:41 sam@poukram /tmp% zzuf -r0.01 cat readme.txt
```



```
ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789
```

```
16/02 1:41 sam@poukram /tmp% █
```

-r0.01 means “fuzz 1% of the files we read”

fuzz even more bits (3.8%)

```
16/02 1:43 sam@poukram /tmp% zzuf -r0.038 cat readme.txt
_0 ! _ _ _ _ _ ] A _0 0_ [ø b _ _ _ _ _ ] _ _ _ _ _ ! _ _ _ N
| \ | | 0 _ _ _ ~ } ~ | 0 } ( /ø _ \ X(X( / ' _ \ | ( _ \ | ` | d } _ \
| _ | | _ | $ | | | ` } | | \ < \ ' \ / | | ` |ø | _ ) | | | | }
| _ | | ! \ _ 0 | | _ _ _ | x ♠ | < \ \ \ ^ ` \ / | | % | [ < | | _ _ < $ tø | d
| _ | | _ _ ] ♠ ♠ } _ _ _ | ø ø ^ ^ \ _ _ _ / \ _ o \ ] / " | _ _ _ / | _ | \ 0 \ _ _ _ < _ _ ]ø /
σ$AC▷DøGH IJKD}NOPQRøTUVWXxZ abc$eggh)zkømnoPqzst}øwxyr$0123456789

16/02 1:43 sam@poukram /tmp% □
```

-r0.038 means “fuzz 3.8% of the files we read”

refuse non-printable bytes

```
16/02 1:46 sam@poukram /tmp% zzuf -r0.038 -R '\x00-\x20\x7f-\xff' cat readme.txt
```

```
|_0 ! _ _ _ _ _ ] A  _0  0_ [_ b  _ _ _ _ _ ] _ _ _ _ _ ! _ _ _ N  
| \ | | 0 _ _ _ ~ } ~ | 0 } ( / _ \ X(X( / ' _ \ | ( _ \ | ' | d } _ _ \  
| | | | _ | $ | | | | ' } | | \ \ \ / / | | | | _ ) | | | } |  
| _ | | ! \ _ 0 | | _ _ | x _ _ | < _ \ \ \ \ ^ \ V / | | | % | [ < | | _ _ < $ t _ | d  
| | | | _ ] _ _ } _ _ _ _ ^ ^ \ _ _ / \ _ o \ ] / " | _ _ / | | \ 0 \ _ _ _ _ < _ ] /
```

```
$ACCDEFGHIJKD}NOPQRSTUVWXYZ abc$eggh)zklmnoPqzst}vwxyr$0123456789
```

```
16/02 1:46 sam@poukram /tmp% □
```

-R \x00-\x20 means “do not generate characters in the range 0x00 to 0x20”

example #2

- Using `zzuf` and `file`

trivial use of file

```
16/02 2:07 sam@poukram /tmp% file /bin/ls
/bin/ls: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), for GNU/Linux
 2.4.1, dynamically linked (uses shared libs), stripped
16/02 2:07 sam@poukram /tmp% □
```

Using -d (debug)

```
16/02 2:09 sam@poukram /tmp% zzuf -d -r0.001 file /bin/ls
** zzuf debug ** libzzuf initialised for PID 27060
** zzuf debug ** fopen64("/etc/magic", "r") = [3]
** zzuf debug ** fgets(0xbfbea6ef, 8192, [3]) = 0xbfbea6ef
** zzuf debug ** fgets(0xbfbea6ef, 8192, [3]) = 0xbfbea6ef
** zzuf debug ** fgets(0xbfbea6ef, 8192, [3]) = 0xbfbea6ef
** zzuf debug ** fgets(0xbfbea6ef, 8192, [3]) = NULL
** zzuf debug ** fclose([3]) = 0
** zzuf debug ** open64("/usr/share/file/magic.mgc", 0) = 3
** zzuf debug ** mmap64(NULL, 1012224, 3, 2, 3, 0) = 0xb792b008 "\x1c\x04\x1e\xfb
1...
** zzuf debug ** close(3) = 0
** zzuf debug ** open64("/bin/ls", 0) = 3
** zzuf debug ** read(3, 0xb78e3008, 262144) = 77352 "\x7fELF...
** zzuf debug ** close(3) = 0
/bin/ls: ERROR: cannot happen: invalid relation `0'
16/02 2:09 sam@poukram /tmp% █
```

**Files in /etc and /usr/share
should not be fuzzed**

Using `-E` (exclude)

```
16/02 2:10 sam@poukram /tmp% zzuf -r0.001 -E/etc -E/usr/share file /bin/ls
/bin/ls: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), dynamically l
inked (uses shared libs), stripped
16/02 2:10 sam@poukram /tmp% █
```

**Files in `/etc` and `/usr/share`
are now properly ignored
(see also `-c` and `-I`)**

Using `-s` (seed)

```
16/02 2:30 sam@poukram /tmp% zzuf -s0:5 -r0.01 -E/etc -E/usr/share file /bin/ls
/bin/ls: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), dynamically l
inked (uses shared libs), corrupted section header size
/bin/ls: ELF 32-bit LSB executable, (SYSV), statically linked (uses shared libs)
, stripped
/bin/ls: data
/bin/ls: data
/bin/ls: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), bad note name
size 0x80000061, dynamically linked, stripped
16/02 2:30 sam@poukram /tmp% □
```

`-s0:5` means “use random seeds 0, 1, 2, 3 and 4”

Using `-r` with ranges

```
16/02 2:34 sam@poukram /tmp% zzuf -s0:5 -r0.001:0.05 -E/etc -E/usr/share file /bin/ls
/bin/ls: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), dynamically linked (uses shared libs), stripped
/bin/ls: data
/bin/ls: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), dynamically linked (uses shared libs), stripped
/bin/ls: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), corrupted program header size, stripped
/bin/ls: data
16/02 2:34 sam@poukram /tmp% □
```

**`-r0.001:0.05` means
“choose fuzzing ratios in
the 0.1% - 5% range”**

example #3

- Finding real bugs

giftopnm

```
16/02 9:13 sam@poukram /tmp% zzuf -q -s0:1000 -r0.001:0.1 giftopnm image.gif
zzuf[s=19,r=0.001:0.1] signal 11 (SIGSEGV)
[1] 5328 exit 1 zzuf -q -s0:1000 -r0.001:0.1 giftopnm image.gif
16/02 9:13 sam@poukram /tmp% zzuf -s19 -r0.001:0.1 < image.gif > fuzzed.gif
16/02 9:13 sam@poukram /tmp% giftopnm fuzzed.gif
[1] 5389 segmentation fault giftopnm fuzzed.gif
16/02 9:13 sam@poukram /tmp%
```

Use `-q` to avoid flooding

Use the same `-r` and `-s` flags to generate a file that reproduces the behaviour

antiword

```
16/02 9:06 sam@poukram /tmp% zzuf -C10 -q -s0:10000 -r0.001:0.02 -M1000 antiword
worddocument.doc
*** glibc detected *** double free or corruption (!prev): 0x0807a020 ***
zzuf[s=19,r=0.001:0.02]: signal 6 (SIGABRT)
zzuf[s=98,r=0.001:0.02]: signal 11 (SIGSEGV)
zzuf[s=109,r=0.001:0.02]: signal 11 (SIGSEGV)
*** glibc detected *** double free or corruption (out): 0x0807a020 ***
zzuf[s=140,r=0.001:0.02]: signal 6 (SIGABRT)
*** glibc detected *** double free or corruption (out): 0x0807a020 ***
zzuf[s=188,r=0.001:0.02]: signal 6 (SIGABRT)
zzuf[s=214,r=0.001:0.02]: signal 9 (memory exceeded?)
*** glibc detected *** double free or corruption (!prev): 0x0807a020 ***
zzuf[s=256,r=0.001:0.02]: signal 6 (SIGABRT)
zzuf[s=269,r=0.001:0.02]: signal 11 (SIGSEGV)
zzuf[s=270,r=0.001:0.02]: signal 9 (memory exceeded?)
zzuf[s=283,r=0.001:0.02]: signal 9 (memory exceeded?)
[1] 2818 exit 1 zzuf -C10 -q -s0:10000 -r0.001:0.02 -M1000 antiword wordd
ocument.doc
16/02 9:06 sam@poukram /tmp%
```

Many different ways to crash

mpplayer/ffmpeg

```
16/02 9:40 sam@poukram /tmp% zzuf -q -s0:1000 -r0.001:0.1 -S -c -b 1000000-1200000
00 mplayer -- -ao null -vo null -benchmark -fps 1000 video.wmv
zzuf[s=1,r=0.001:0.1]: signal 11 (SIGSEGV)
[1] 10231 exit 1 zzuf -q -s0:1000 -r0.001:0.1 -S -c -b 1000000-1200000 mp
layer -- -ao null -vo
16/02 9:40 sam@poukram /tmp% zzuf -s1 -r0.001:0.1 -b 1000000-1200000 < video.wmv
> fuzzed.wmv
16/02 9:40 sam@poukram /tmp% □
```

Use `-b` to only fuzz parts of the input

Innocent-looking videos can cause a DoS or worse

other features

- Fork and parallelise (-F)
- Detect stuck processes
 - Set maximum memory allocation (-M)
 - Set maximum running time (-T)
 - Set maximum `stdout` output (-B)
- Fuzz network input (-n)
- Fuzz `stdin` (-i)

zzuf's future

- Context-dependent fuzzing
 - ignore or recompute CRCs
 - divert the zlib library, too
- Finish the Windows® port
 - help needed
- Make it easier to test GUI applications
- Attach to a debugger

<http://sam.zoy.org/zzuf/>

**THANKS A LOT FOR
COMING EARLY AND
LISTENING**



**NOW YOU
CAN ENJOY YOUR
SANDWICH**