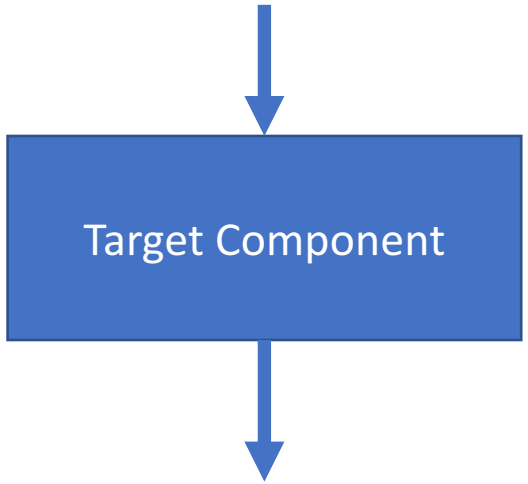
Integrate libFuzzer with the NetBSD Userland

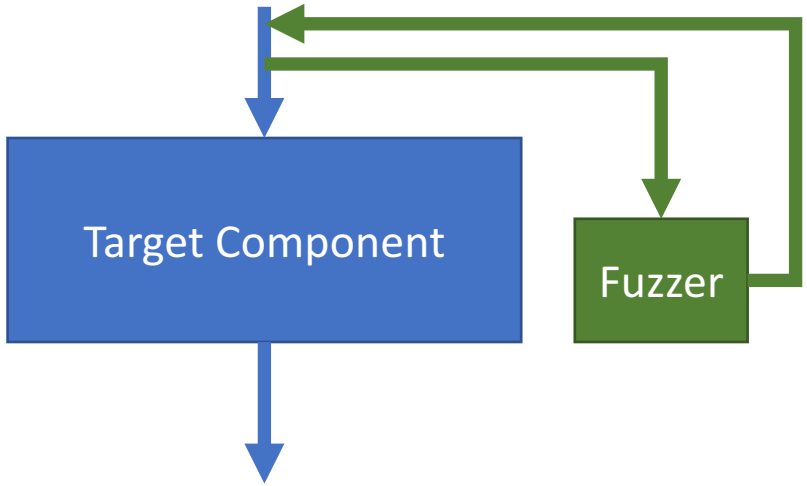
Yang Zheng

Shanghai Jiao Tong University

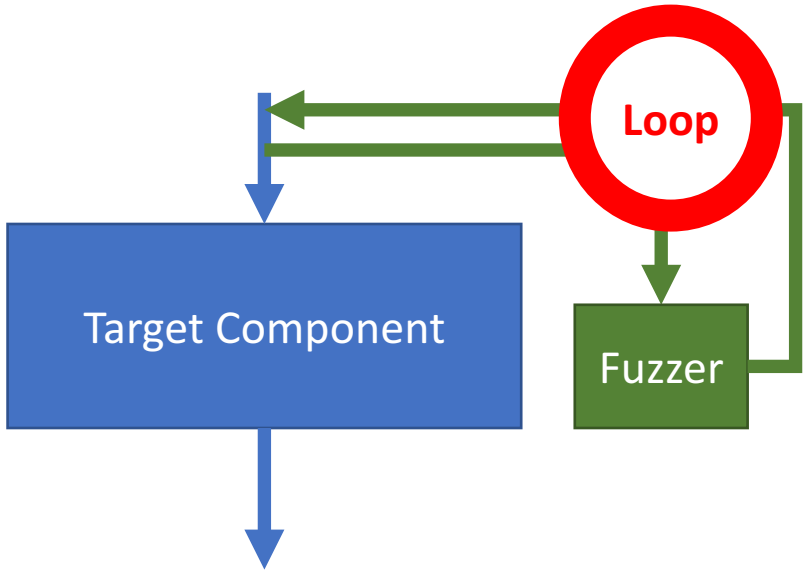
What is Fuzzing?



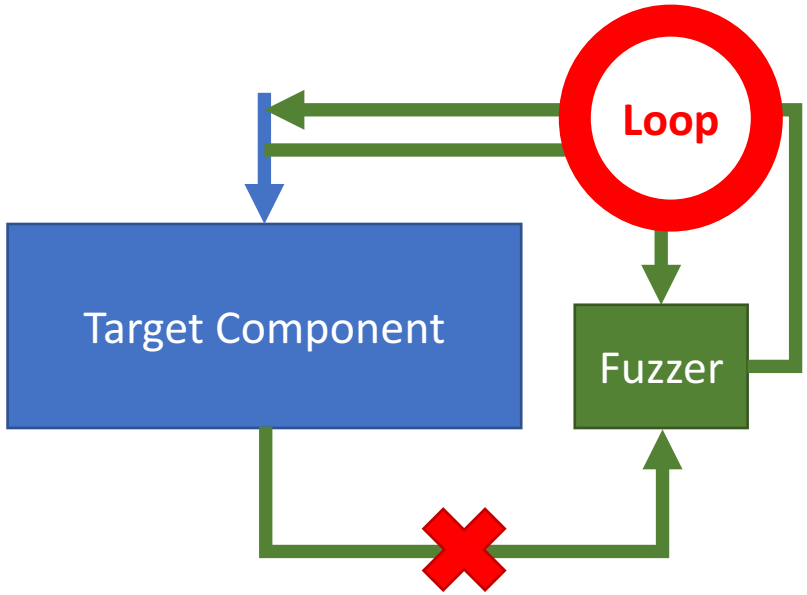
What is Fuzzing?



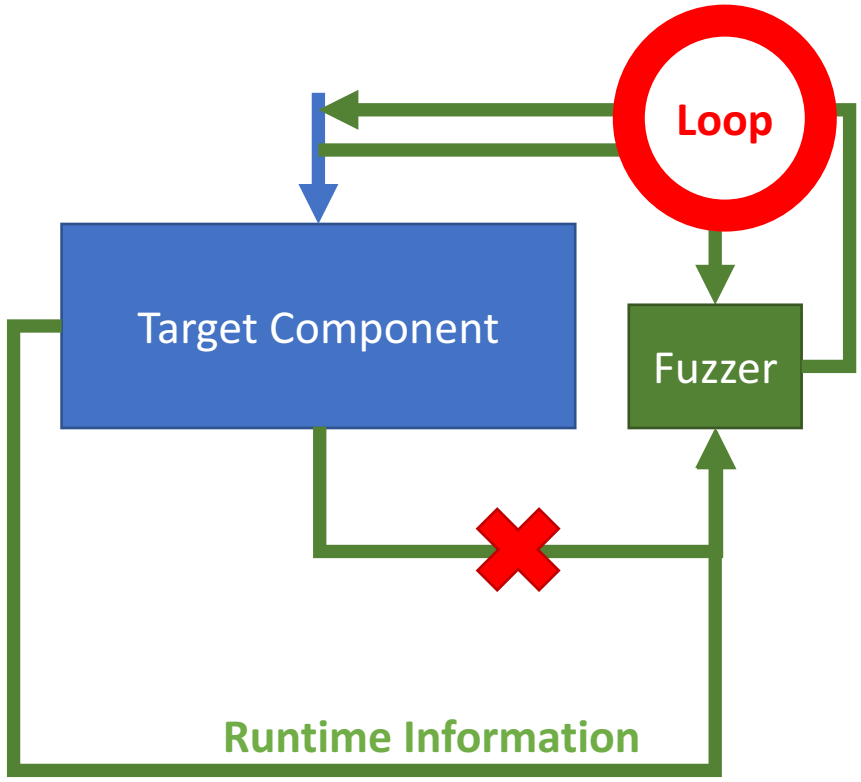
What is Fuzzing?



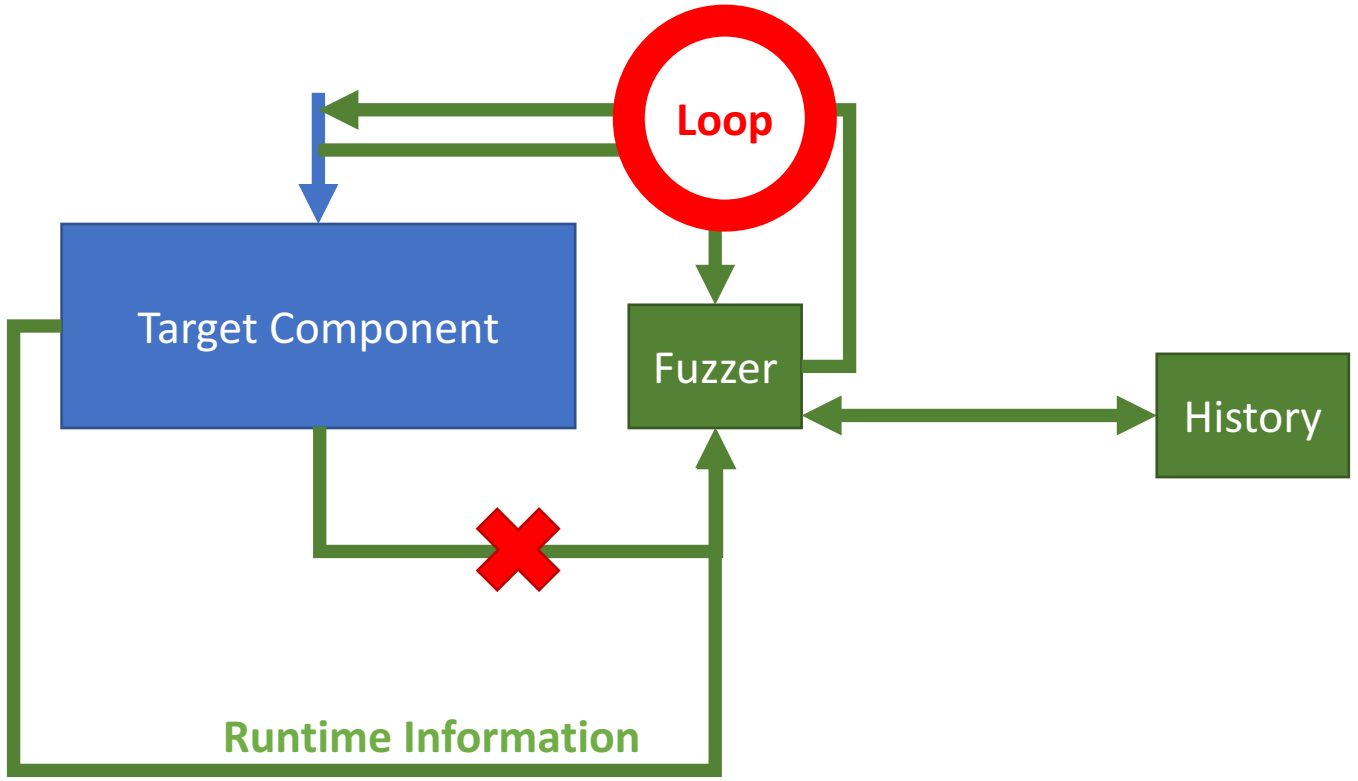
What is Fuzzing?



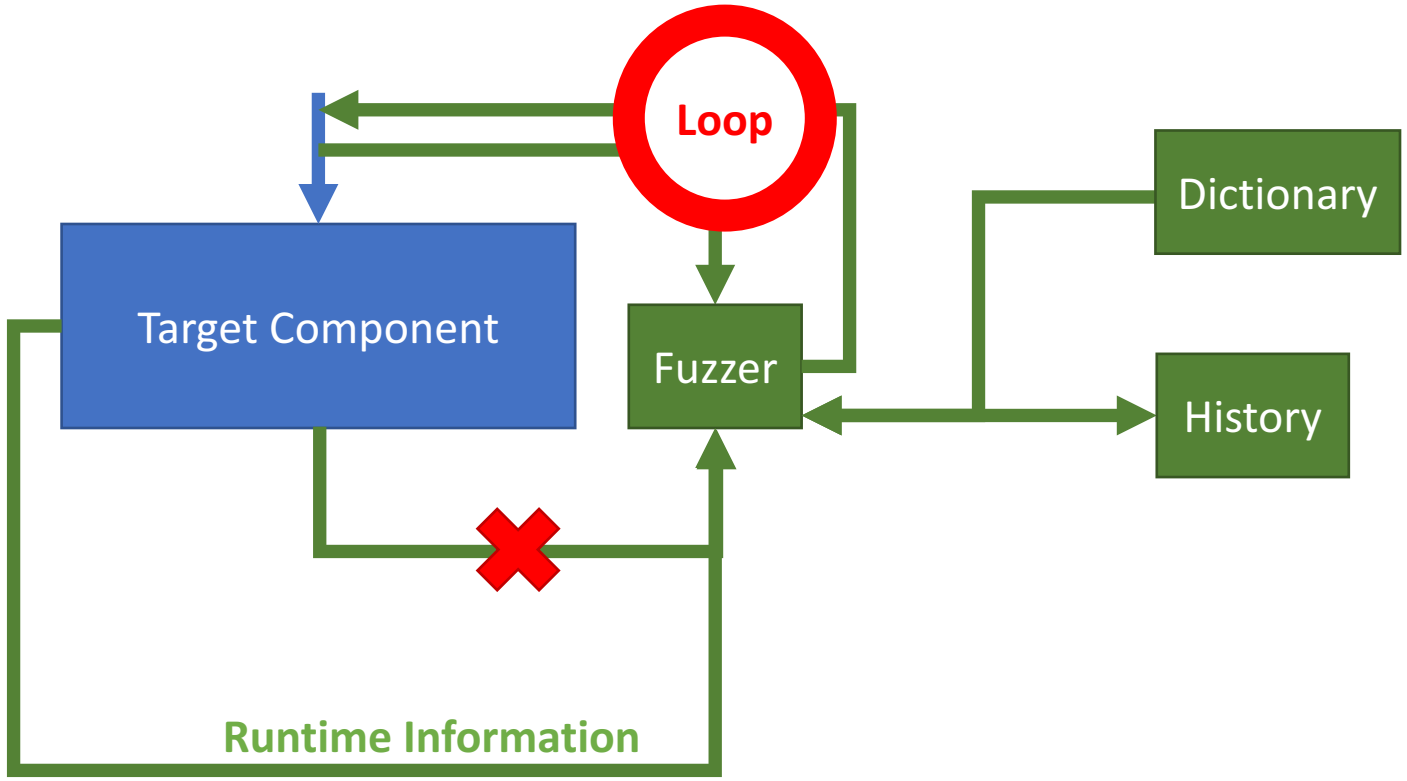
What is Fuzzing?



What is Fuzzing?



What is Fuzzing?



Why We Need libFuzzer?

- libFuzzer: coverage-guided, evolutionary fuzzing engine
- Automatically generated test cases
- Robustness
- Correctness
  - Sanitizers

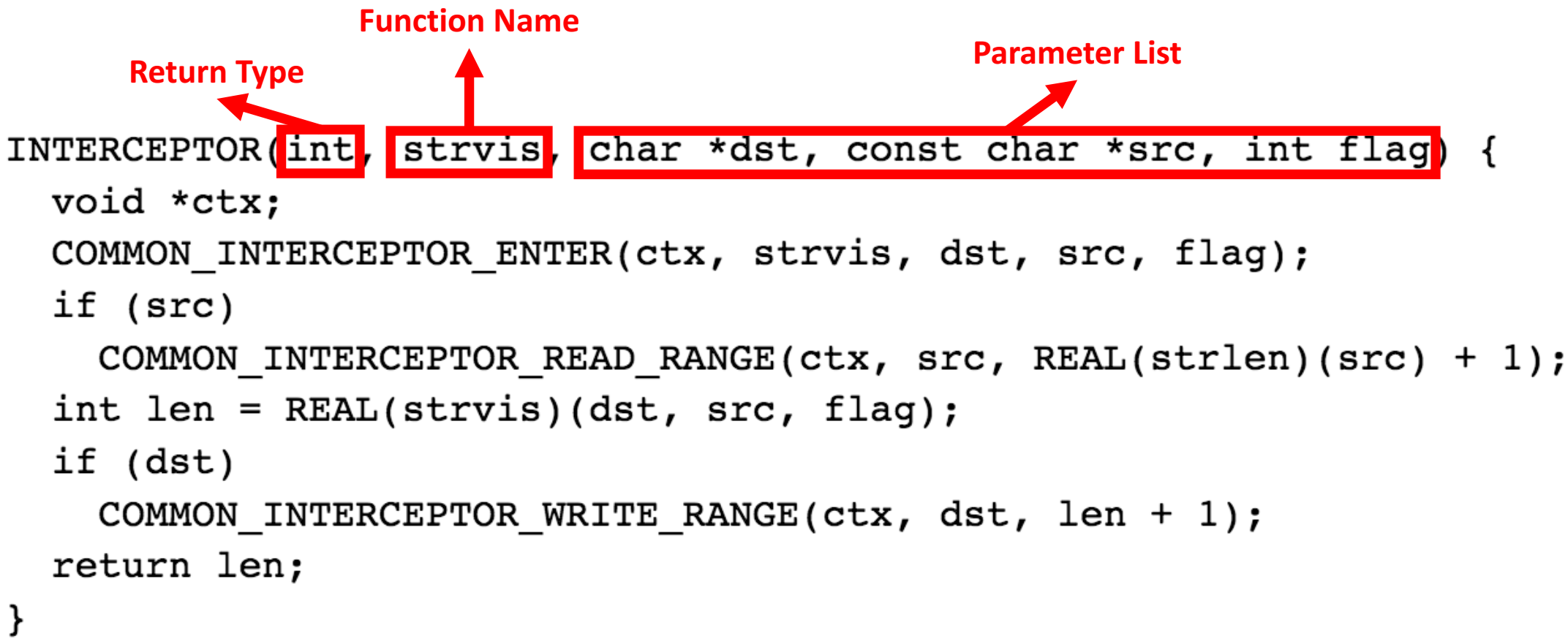
Sanitizer Integration

- Runtime tools (in LLVM)
  - AddressSanitizer (ASan)
  - ThreadSanitizer (TSan)
  - MemorySanitizer (MSan)
  - UndefinedBehaviorSanitizer (UBSan)
  - DataFlowSanitizer (DFSan)
  - LeakSanitizer (LSan)
- Compiler instrumentation

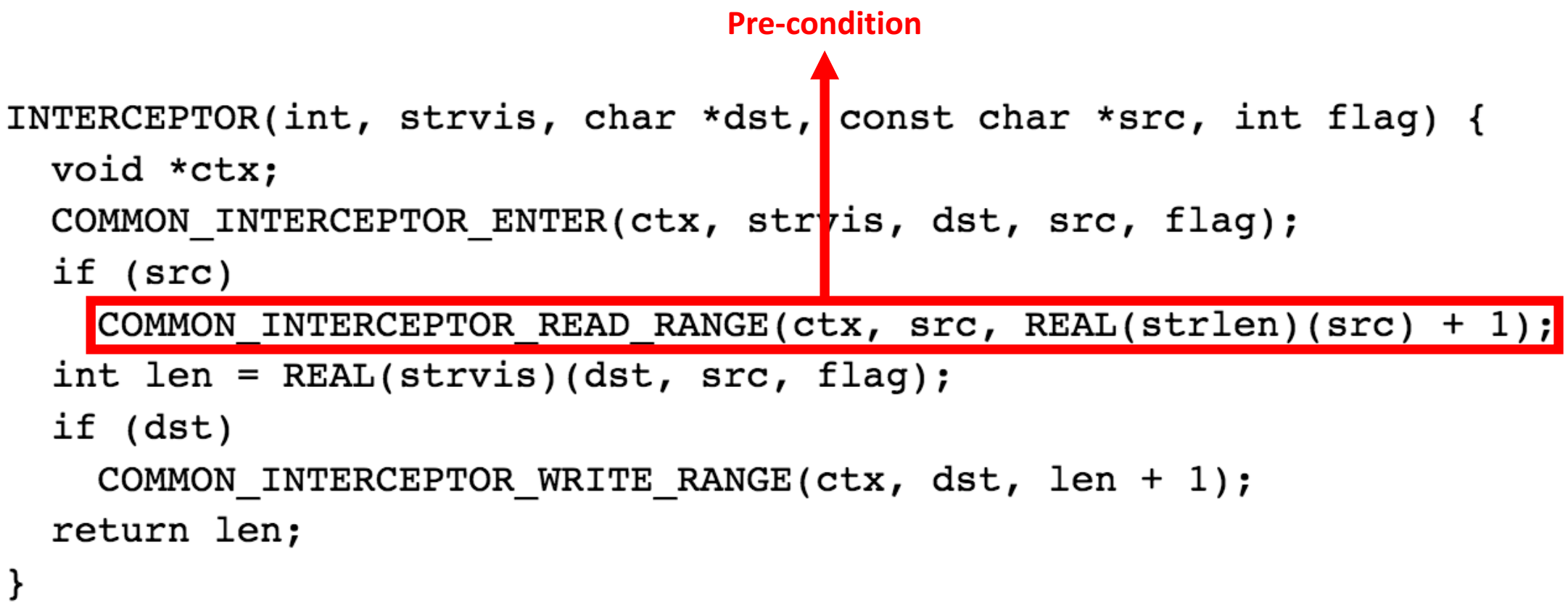
Interceptor

- Not all source can be instrumented
  - Libraries
    - libc
    - libm
    - libpthread
  - Syscalls
- Interceptors
  - Manually provide information for uninstrumented interfaces
- Added several interceptors to enable sanitizers

Interceptor: Example

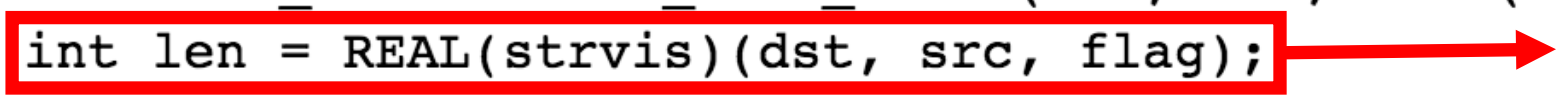


Interceptor: Example



Interceptor: Example

```
INTERCEPTOR(int, strvis, char *dst, const char *src, int flag) {
  void *ctx;
  COMMON_INTERCEPTOR_ENTER(ctx, strvis, dst, src, flag);
  if (src)
    COMMON_INTERCEPTOR_READ_RANGE(ctx, src, REAL(strlen)(src) + 1);
  int len = REAL(strvis)(dst, src, flag);
  if (dst)
    COMMON_INTERCEPTOR_WRITE_RANGE(ctx, dst, len + 1);
  return len;
}
```



Interceptor: Example

```
INTERCEPTOR(int, strvis, char *dst, const char *src, int flag) {
  void *ctx;
  COMMON_INTERCEPTOR_ENTER(ctx, strvis, dst, src, flag);
  if (src)
    COMMON_INTERCEPTOR_READ_RANGE(ctx, src, REAL(strlen)(src) + 1);
  int len = REAL(strvis)(dst, src, flag);
  if (dst)
    COMMON_INTERCEPTOR_WRITE_RANGE(ctx, dst, len + 1);
  return len;
}
```

Post-condition

Interceptor: Unsolved Issues

- FILE structure
  - Implementations vary lot on different Oses
- mount(2) interface
  - Parameters vary a lot for different file systems
- getchar(3)/putchar(3) interfaces
  - Complicated definitions with macros

libFuzzer Usage

- Interfaces
  - LLVMFuzzerTestOneInput
  - LLVMFuzzerInitialize
  - LLVMFuzzerCustomMutator
  - LLVMFuzzerCustomCrossOver

```
#include <stdint.h>
#include <stddef.h>
extern "C" int LLVMFuzzerTestOneInput(const uint8_t *data, size_t size) {
  if (size > 0 && data[0] == 'H')
    if (size > 1 && data[1] == 'I')
       if (size > 2 && data[2] == '!')
       __builtin_trap();
  return 0;
}
```

libFuzzer Usage

- Compilation
  - clang++ -fsanitize=fuzzer,address test.cc
- Fuzzing

```
INFO: Seed: 1523017872
INFO: Loaded 1 modules (16 guards): [0x744e60, 0x744ea0),
INFO: -max_len is not provided, using 64
INFO: A corpus is not provided, starting from an empty corpus
#0    READ units: 1
#1    INITED cov: 3 ft: 2 corp: 1/1b exec/s: 0 rss: 24Mb
#3811 NEW    cov: 4 ft: 3 corp: 2/2b exec/s: 0 rss: 25Mb L: 1 MS: 5 ChangeBit-ChangeByte-ChangeBit-
#3827 NEW    cov: 5 ft: 4 corp: 3/4b exec/s: 0 rss: 25Mb L: 2 MS: 1 CopyPart-
#3963 NEW    cov: 6 ft: 5 corp: 4/6b exec/s: 0 rss: 25Mb L: 2 MS: 2 ShuffleBytes-ChangeBit-
#4167 NEW    cov: 7 ft: 6 corp: 5/9b exec/s: 0 rss: 25Mb L: 3 MS: 1 InsertByte-
==31511== ERROR: libFuzzer: deadly signal
...
artifact_prefix='./'; Test unit written to ./crash-b13e8756b13a00cf168300179061fb4b91fefbed
```

Fuzzing Whole Program with libFuzzer

- *expr(1)*
- *sed(1)*
- *sh(1)*
- *file(1)*
- *ping(8)*

Fuzzing *expr(1)*

```
$ expr 1 + 1
2
```

- *expr(1)*
  - Evaluate arguments as an expression
- Transform buffer into *argv* vector
- Dictionary file

```
min="-92233720368547758O8"
max="9223372036854775807"
zero="0"
one="1"
negone="-1"
div="/"
mod="%"
add="+"
sub="-"
or="|"
add="&"
```

Fuzzing *expr(1)*

- Bugs (reproduced/found)
  - Get *SIGFPE* when fed "*-9223372036854775808 / -1*"
  - Integer overflow detected by the UBSan
    - "*9223372036854775807 * -3*"

Fuzzing *sed(1)*

- *sed(1)*
  - Stream editor
  - *sed ‘s/abc/def/g’ some-file.txt*
- Transform buffer into commands and text

```
command #1                          s/hello/hi/g
command #2
...                                 hello, world!
command #N
    // an empty line
text strings
```

Fuzzing *sed(1)*

- *exit(3)* issue
  - libFuzzer treat *exit(3)* as errors
  - Replacing it with a return statement?
  - Exceptions (C++)
  - *setjmp(3)*/*longjmp(3)*

Fuzzing *ping(8)*

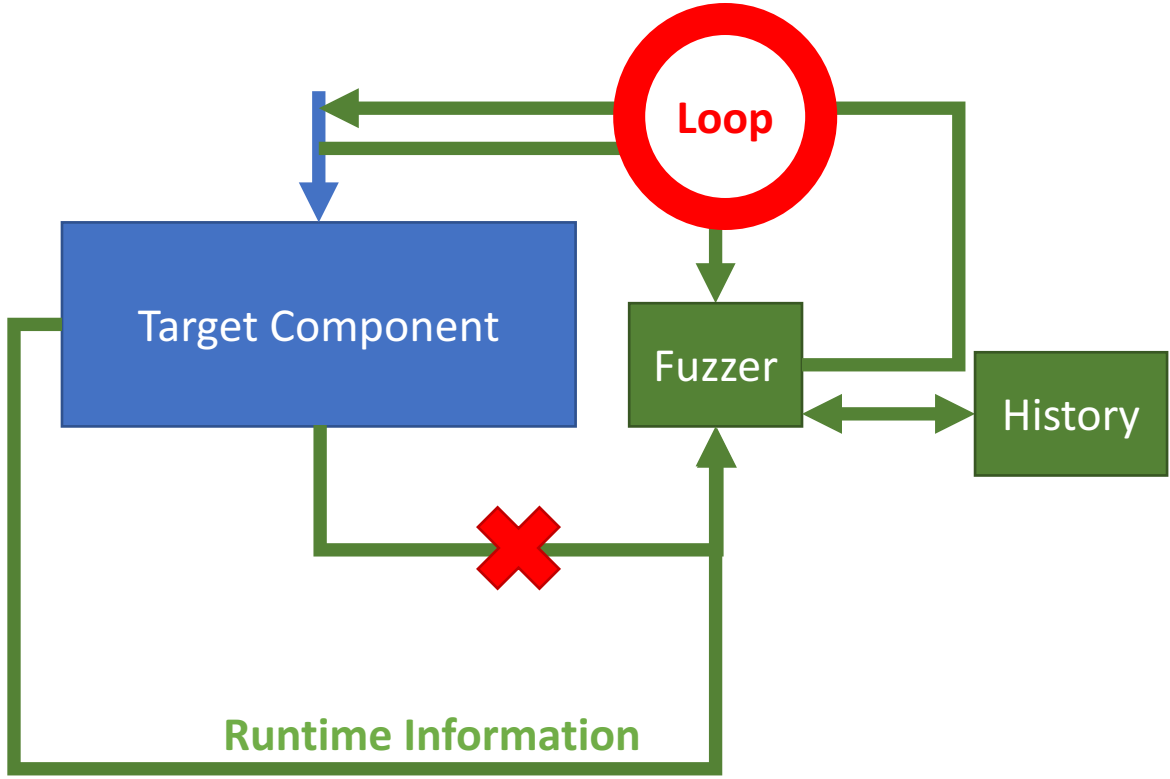
- *ping(8)*
  - send ICMP ECHO_REQUEST packets to network hosts
  - *ping localhost*
- Transform buffer into network packets
  - Implement network interfaces
    - *socket(2)*
    - *recvfrom(2)*
    - *sendto(2)*
    - *poll(2)*
    - *…*

Other Fuzzers

- American Fuzzy Lop (AFL)
  - With compile-time instrumentation and genetic algorithms
- honggfuzz
  - Feature-rich fuzzer
- Radamsa
  - Without compile-time instrumentation

Fuzzing with AFL/honggfuzz

- Compilers
  - AFL: *afl-clang, afl-clang++, afl-gcc, afl-g++*
  - Honggfuzz: *hfuzz-clang, hfuzz-clang++, hfuzz-gcc, hfuzz-g++*
- Input source
  - Standard input
  - File
  - No modifications for *sed(1), sh(1), file(1)*



Fuzzing *expr(1)* with AFL/honggfuzz

- *expr(1)* gets input from command line (*argv* vector)
  - Modify the *main* function to read file/*STDIN*
  - Reuse the modification from libFuzzer
- Trying to add features to fuzz command line input
  - Leverage *_libc_init* interface to replace command line
  - Use *exec* to execute program with fuzzed parameters
    - Missing fuzzing context with *exec* interface

```
% expr FUZZME / -1
```

Fuzzing *ping(1)* with AFL/honggfuzz

- Reuse the “fake” network interfaces from libFuzzer
- *LD_PRELOAD* and *HF_ITER* combo for honggfuzz
  - Shadow network interfaces with *LD_PRELOAD*
  - How to get the fuzzed data?
    - *HF_ITER* interface!

LLVMFuzzerTestOneInput v.s. *HF_ITER*

- Push v.s. Pull

```
extern "C" int LLVMFuzzerTestOneInput(const uint8_t *Data, size_t Size) {
  DoSomethingInterestingWithMyAPI(Data, Size);
  return 0;  // Non-zero return values are reserved for future use.
}
```

```
extern HF_ITER(uint8_t** buf, size_t* len);

int main(void) {
        for (;;) {
                size_t len;
                uint8_t *buf;

                HF_ITER(&buf, &len);

                TestAPI(buf, len);
        }
}
```

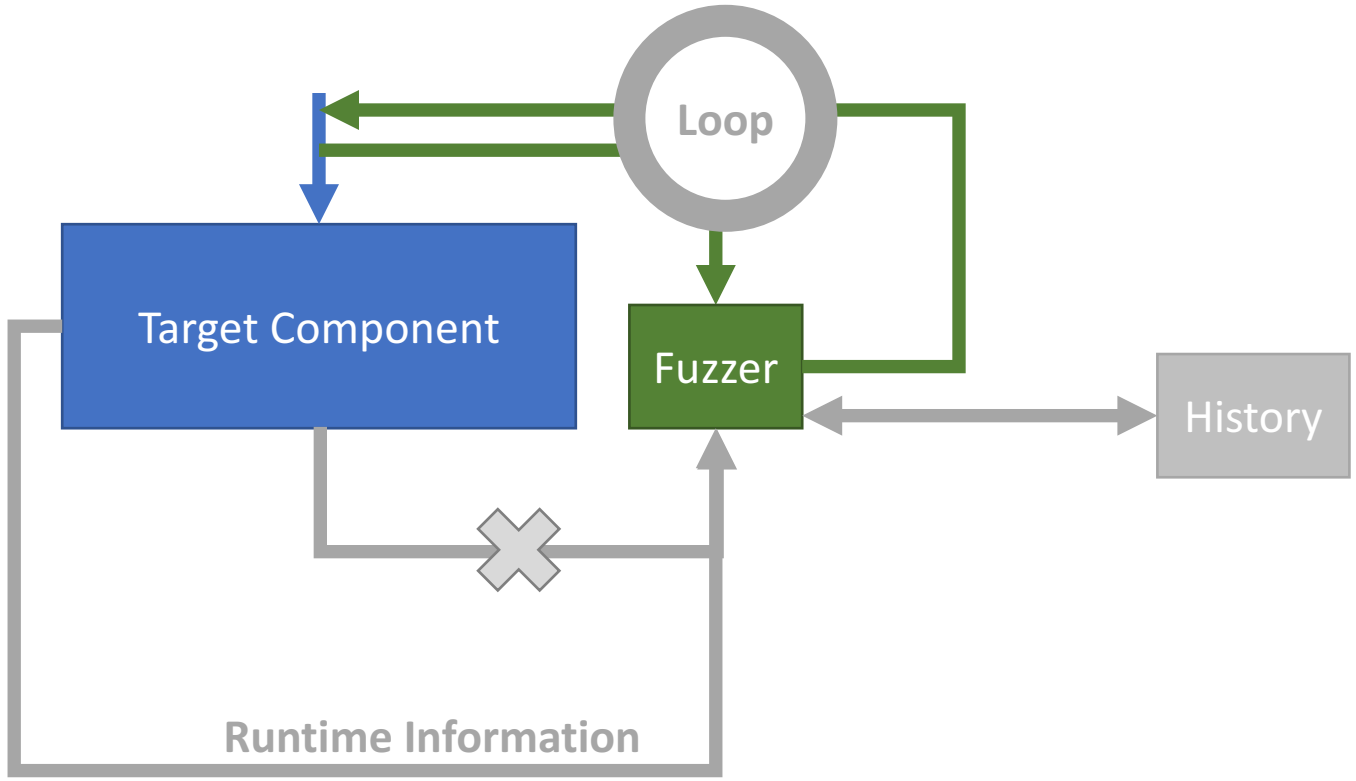
Fuzzing *ping(1)* with AFL/honggfuzz

- Reuse the “fake” network interfaces from libFuzzer
- *LD_PRELOAD* and *HF_ITER* combo for honggfuzz
  - Shadow network interfaces with *LD_PRELOAD*
  - How to get the fuzzed data?
    - *HF_ITER* interface!
  - No modification!

Fuzzing Programs with Radamsa

- Independent with fuzzed programs

```
bash-3.2$ echo "EuroBSDCon" | radamsa
EuroBSBSBSDCon
```

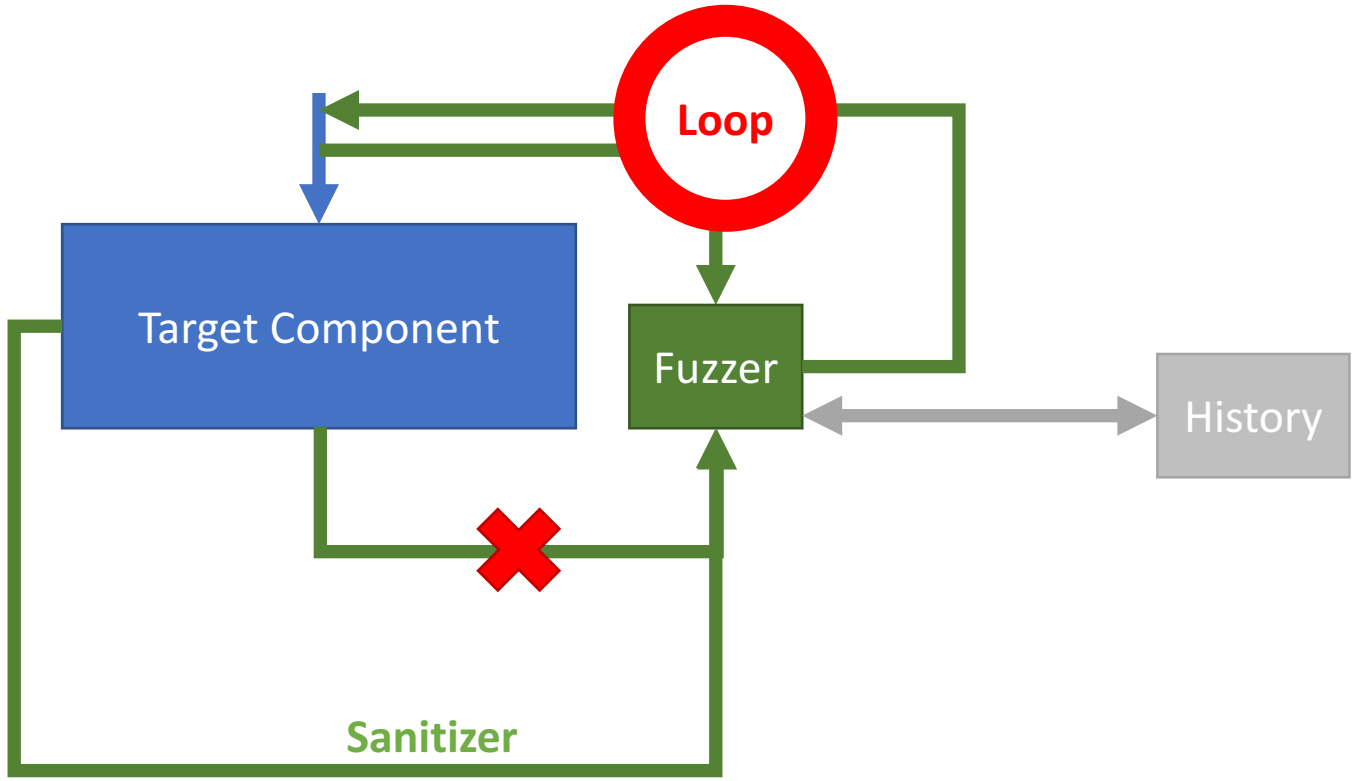


Fuzzing Programs with Radamsa

- Independent with fuzzed programs

```
bash-3.2$ echo "EuroBSDCon" | radamsa
EuroBSBSBSDCon
```

- Work with shell scripts
  - No modification!



Evaluation

- Modifications (LoC)

	expr(1)	*sed(1)*	*sh(1)*	*file(1)*	*ping(8)*
libFuzzer	128	96	60	48	582
AFL/honggfuzz	142	0	0	0	590/0
Radamsa	0	0	0	0	N/A

Evaluation

- Binary size

	Dependency	*Compilers*	*Fuzzer*	*Tools*	*Total*
libFuzzer	0	56MB	N/A	0	56MB
AFL	0	24KB	292KB	152KB	468KB
honggfuzz	36KB	840KB	124KB	0	1000KB
Radamsa	588KB	0	608KB	0	1196KB

Evaluation

- Performance (cases from libFuzzer)

	libFuzzer	AFL	honggfuzz	Radamsa
DivTest +S	< 1s	7s	1s	7s
DivTest	> 10min	> 10min	2s	> 10min
SimpleTest +S	< 1s	> 10min	1s	> 10min
SimpleTest	< 1s	> 10min	1s	> 10min
CxxStringEqTest +S	< 1s	> 10min	2s	> 10min
CxxStringEqTest	> 10min	> 10min	2s	> 10min
CounterTest +S	1s	5min	1s	7min
CounterTest	1s	4min	1s	7min
SimpleHashTest +S	< 1s	3s	1s	2s

Fuzzing Functions with libFuzzer

- libFuzzer is not designed for whole program fuzzing
  - Fuzzing separate functions from NetBSD
  - *regex(3)*
  - checksum functions
    - *mdX(3)*
    - *rmd160(3)*
    - *sha1(3)*
    - *crc*
      - Kernel
      - *cksum(1)*
  - *libutil(3)*
  - *bozohttpd(8)*

Potential Bugs

- Null-pointer errors (2)
- Infinite recursion (1)
- Undefined behavior (2)
- Buffer overflow (1)

Keywords

- NetBSD
  - Thanks to
    - Kamil Rytarowski
    - Christos Zoulas
    - Matthew Green
    - Joerg Sonnenberger
- libFuzzer
- Google Summer of Code (GSoC)
- https://github.com/plusun/src

Thanks for Listening!

Q&A