CRYPTOMATHiC

We enable the development of a secure digital future

Who are we?

Anna Laursen

- Java,
- C/C++

Jan Andersen

- C and C++ Java, C#
- Embedded programming

Agenda

- Why fuzzing?
- What is fuzzing?
- In practice
  - C examples
  - Java examples
- Questions

Why fuzzing?

- JSON parsing
- Untrusted input
- Confidence in code -> fuzzing
- Common Criteria Certification

What is fuzzing?

- Automated testing technique
- Bug hunting: presence of bugs, not absence!

What is fuzzing?

What can it detect?

Detect errors unrelated to functional requirements

- Memory leaks
- Buffer overflows
- Concurrency issues
- Infinite loops
- Uncaught exceptions
- …

What is fuzzing?

Valuable for

- Untrusted input - *security*
- Equivalence of algorithms - *correctness*
- Complex input to high-volume API - *stability*

What is fuzzing?

Watch out!

Fuzzing

- is **not** for explicit test of functional requirements
- finds errors **unrelated to program requirements**!

What is fuzzing?

Categories - input data

- Generation-based
- Mutation-based
  - extension: coverage guided fuzzing

What is fuzzing?

Categories – input structure awareness

- Structured
- Unstructured

What is fuzzing?

Categories – program structure awareness

- Black box

  Fast, “scratches surface”

- White box

  Slow, gets deep into code

- Grey box

  Best of the two worlds

In practice - examples

- C examples
- Java examples

C / C++

Fuzz testing tools

Many tools exists – the major ones being

- AFL / AFL++
- LibFuzzer
- Honggfuzz
- FuzzTest

C / C++

We chose

- LibFuzzer part of the LLVM suite
- Already using sanitizers
  - address
  - memory
  - undefined behavior

C / C++

How does it work?

- Instrumentation – grey box
- Input mutation
- Coverage feedback
- Seed inputs

C / C++

A little example

```
int doTest(const uint8_t *data, size_t size)
{
 if (size == 2)
  if (data[0] == 'H')
   if (data[1] == 'i')
    exit(1);
 return 0;
}
```

C / C++

Running the fuzzer - results

- Output is not easy to read

```
INFO: Running with entropic power schedule (0xFF, 100).
INFO: Seed: 881530181
INFO: Loaded 1 modules   (20 inline 8-bit counters): 20 [0x54f2c8, 0x54f2dc),
INFO: Loaded 1 PC tables (20 PCs): 20 [0x54f2e0,0x54f420),
INFO: -max_len is not provided; libFuzzer will not generate inputs larger than 4096 bytes
INFO: A corpus is not provided, starting from an empty corpus
#2      INITED cov: 3 ft: 3 corp: 1/1b exec/s: 0 rss: 30Mb
#3      NEW    cov: 4 ft: 4 corp: 2/3b lim: 4 exec/s: 0 rss: 30Mb L: 2/2 MS: 1 CopyPart-
#64     NEW    cov: 5 ft: 5 corp: 3/5b lim: 4 exec/s: 0 rss: 30Mb L: 2/2 MS: 1 CMP- DE: "H\x00"-
#12921  NEW    cov: 6 ft: 6 corp: 4/7b lim: 128 exec/s: 0 rss: 31Mb L: 2/2 MS: 2 CrossOver-ChangeByte-
#2179251       DONE   cov: 6 ft: 6 corp: 4/7b lim: 4096 exec/s: 1089625 rss: 187Mb
```

C / C++

‘Graphical’ result

```
doTest.cpp:
    1|       |#include "doTest.h"
    2|       |#include <stdlib.h>
    3|       |
    4|       |int doTest(const uint8_t *data, size_t size)
    5|  2.12M|{
    6|  2.12M|  if (size == 2)
    7|   521k|    if (data[0] == 'H')
    8|   122k|      if (data[1] == 'i')
    9|  22.7k|        return 1;
   10|  2.10M|  return 0;
   11|  2.12M|}
```

C / C++

‘Graphical’ result

```
File 'doTest.cpp':
Name                        Regions    Miss   Cover     Lines    Miss   Cover  Branches    Miss   Cover
-------------------------------------------------------------------------------------------------------
_Z6doTestPKhm                     8       0 100.00%         7       0 100.00%         6       0 100.00%
-------------------------------------------------------------------------------------------------------
TOTAL                             8       0 100.00%         7       0 100.00%         6       0 100.00%

File 'fuzzTest.cpp':
Name                        Regions    Miss   Cover     Lines    Miss   Cover  Branches    Miss   Cover
-------------------------------------------------------------------------------------------------------
LLVMFuzzerTestOneInput            1       0 100.00%         4       0 100.00%         0       0   0.00%
-------------------------------------------------------------------------------------------------------
TOTAL                             1       0 100.00%         4       0 100.00%         0       0   0.00%
```

C / C++

Dictionary

- Keywords / byte sequences
- Helping the fuzzer
- Improving speed

- Example dictionaries

C / C++

Corpus

- Accumulation of ‘interesting’ input
- Acts as starting point for next run

C / C++

Crash ‘triage’

- Call stack
- Crash file
- Rerunning with the input causing the crash

Java

Errors in memory safe languages:

- Uncaught exceptions
- Inconsistent implementations (correctness)
- Infinite loops
- Out Of Memory/Stack Overflow
- …

Java

Examples

- No advanced details, just arouse curiosity
- Using Jazzer
  - based on libFuzzer

Java

Uncaught Exceptions - example

```
public int divide(final int a, final int b)
{
  return a/b;
}
```

Java

Out Of Memory - example

```
private static final List<Double> list = new
ArrayList<>();

public List<Double> getListWithRandomData()
{
  for (int i = 0; i < 1000; i++)
    list.add(Math.random());
  return list;
}
```

Java

Stack Overflow - example

```
public int fibonacci(final int n)
{
  if (n < 2)
    return n;
  return fibonacci(n - 1) + fibonacci(n - 2);
}
```

Questions?

Time to take control of your cryptographic security

enquiry@cryptomathic.com www.cryptomathic.com +45 8676 2288