# Fuzzing strengths and weaknesses

# Who am I?

Florian Ledoux @Myst3rie

Was student here  (Master TIIR)

Passionate about vulnerability research for years

# Summary

- What is fuzzing?
- Type of fuzzing
- Fuzzing tips and tricks
- Monitoring for crashes
- Typical examples

# What fuzzing is?

Providing invalid data to the inputs of a computer program to find bugs

Will not find all bugs, but provide a good picture of the robustness of the target software

# How fuzzing works?

- Identify source of input to a program
- Permute or generate pseudorandom input
- Monitor for exceptions, crash, memory leak
- Record the input and state that generate faults

# Fuzzer types

- Mutation (Change input)
    - fuzzer is dependent from the input
    - require lots of differents clean input

- Generation (Create input)
    - require some level of intelligence (RFC, Spec)

# Fuzzer types

- Dumb / Blind fuzzer (Random)
- Smart fuzzer
  - Knowledge of the input format
- Evolutionary fuzzer (Feedback)
  - Binary Instrumentation
  - Compile-time Instrumentation

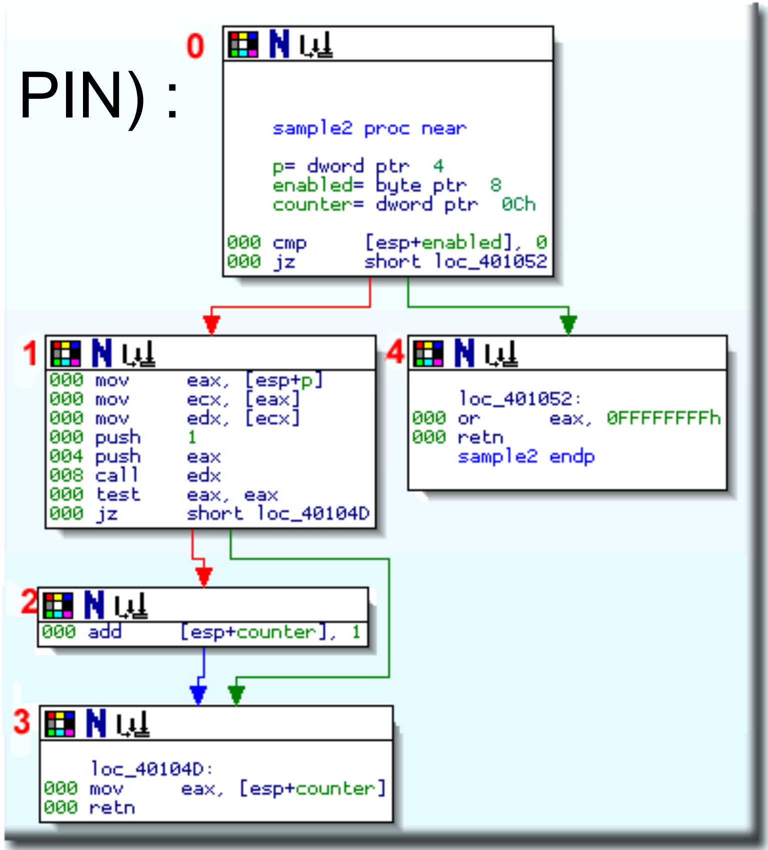
# Evolutionary fuzzer

Dynamic Binary instrumentation (like PIN) :

- Good documentation
- Works on both Linux / Windows

Give feedback:

- Code coverage
- Execution time



```
sample2 proc near

p= dword ptr  4
enabled= byte ptr  8
counter= dword ptr  0Ch

000 cmp     [esp+enabled], 0
000 jz      short loc_401052
```

```
000 mov     eax, [esp+p]
000 mov     ecx, [eax]
000 mov     edx, [ecx]
000 push    1
004 push    eax
008 call    edx
000 test    eax, eax
000 jz      short loc_40104D
```

```
000 add     [esp+counter], 1
```

```
    loc_40104D:
000 mov     eax, [esp+counter]
000 retn
```

```
    loc_401052:
000 or      eax, 0FFFFFFFFh
000 retn
    sample2 endp
```

# Fuzzing tips and tricks

- There is no rules, no bad ideas

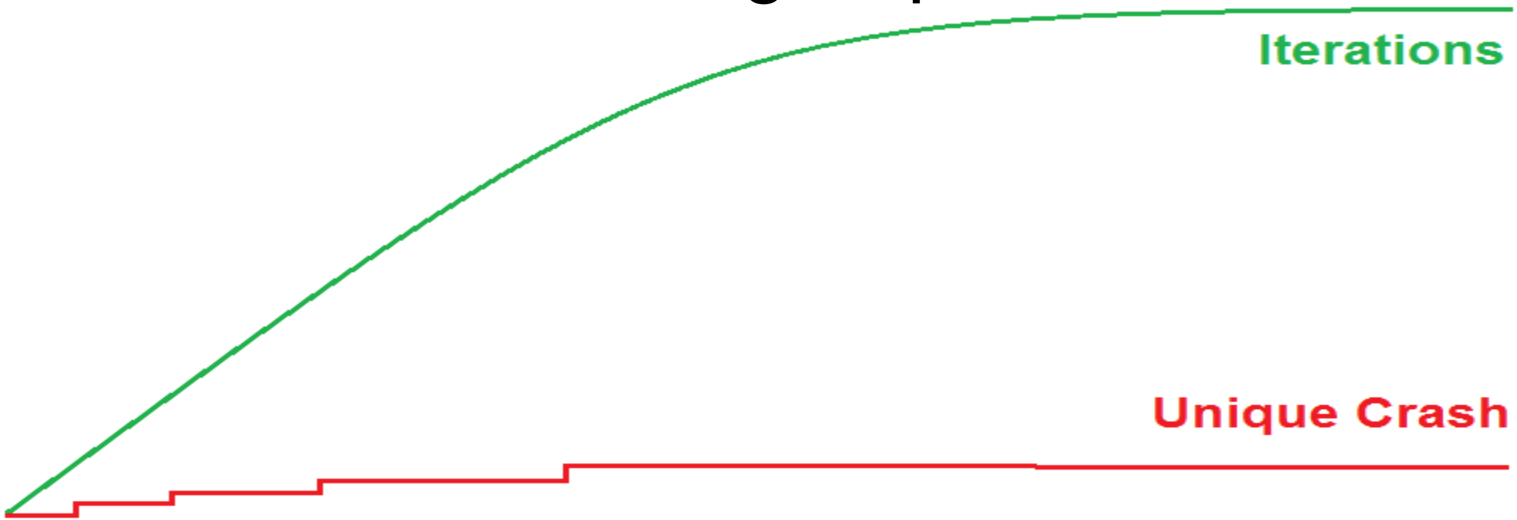
- Think out of the box

- Do what others won't (or find duplicates)

# Fuzzing tips and tricks

Quality is more important than Quantity :

“Microsoft SDL Fuzzing require 100K iterations”



# Fuzzing tips and tricks

Always check for (Mutation fuzz) :

- CRC
- Compression

# Fuzzing tips and tricks

Always check for (Generation fuzz) :

- Speed (Generation time)
- Overfitting (Too complex, conflict)

# Fuzzing tips and tricks

Fuzzing requires static analysis

Other approaches (Independent) are immature

- Satisfiability Modulo Theories (SMT like z3)

# Fuzzing tips and tricks

It's genuinely hard to compete with brute force when your "smart" approach is resource-intensive.

If your instrumentation makes it 10x more likely to find a bug, but runs 100x slower, your users getting a bad deal.

Michał Zalewski @lcamtuf

# Monitoring for crash

If you have source code :

AddressSanitizer (Work on Linux / Windows)

# Monitoring for crash

If you don’t have source code :

- Minidump
- PyDBG

# Monitoring for crash

Woot ! I get a crash :)

- Crashes
- Unique EIPs
- Really Unique EIPs
- Exploitable?

# Typical examples

Case of two formats :

- HTML
- PNG

# Fuzzing PNG

Compact structure, made of chunks

A chunk :

| | |
|---|---|
| ● Length | 4 bytes |
| ● Type | 4 bytes |
| ● Chunk data | “Length” bytes |
| ● crc | 4 bytes |

# Fuzzing PNG

3 critical chunks

15 optional chunks

Strategy?

# Fuzzing PNG

Mutation (Bit flip) + Feedback

AFL, American Fuzzy Loop

Fast, good on compact structure

# Fuzzing HTML

Mutation ? (Forget AFL) -> Radamsa

Radamsa :

- Based on input
- Aware of balise / grammar

# Fuzzing Exemples

Both AFL and Radamsa are easy to use

Good start for fuzzing

# Question ?