0-Knowledge Fuzzing

Vincenzo Iozzo
vincenzo.iozzo@zynamics.com
Disclaimer

In this talk you won’t see all those formulas, definitions, code snippets and bullets.
From past experiences the speaker learned that all the aforementioned elements are no useful in making people understand your idea. You instead will see a lot of funny pictures which the speaker hopes will convey better understanding of the ideas explained in the talk.

\[(S_N f)(x) = \frac{a_0}{2} + \sum_{n=1}^{N} [a_n \cos(nx) + b_n \sin(nx)], \quad N \geq 0.\]

You don’t want slides like this, do you?
Motivations

That chart explained the quantum Hall effect. Now, if you'll bear with me for a moment, this next graph shows rainfall over the Amazon Basin...

If you keep saying "bear with me for a moment", people take a while to figure out that you're just showing them random slides.
Questions!
Fuzzing
How it used to be
How it is today
(aka the reason of this talk)
Dumb fuzzing
Smart Fuzzing
Evolutionary Based Fuzzing
The idea

- Static analysis metrics
- Data tainting
- In-memory fuzzing
- Code coverage/fault monitor
The surface
We need a filter
Cyclomatic complexity
This one
Not this one
Original formula

M = E − N + 2P

- Number of edges
- Number of nodes
- Connected components
Why? Cyclomatic number

\[ M = E - N + P \]
Simplify
Formula

\[ M = E - N + 2 \]
Problem
Loop detection
Dominator tree
WHERE DO YOU THINK YOU'RE GOING?

I NEED TO ASK OUR VP OF SALES A QUESTION.

WHOA! WHOA! WHOA!

YOU CAN'T SPEAK DIRECTLY WITH A VICE PRESIDENT.

YOU NEED TO TALK TO YOUR BOSS, WHO TALKS TO HIS BOSS, WHO TALKS TO SOMEONE WHO IS FRIENDS WITH THE VP OF SALES, WHO THEN TALKS TO HIM.

WOULDN'T THAT VIRTUALLY GUARANTEE THAT THE WRONG QUESTION GETS ASKED?

IT'S BETTER TO HAVE THE RIGHT PERSON ASK THE WRONG QUESTION THAN THE WRONG PERSON ASK THE RIGHT QUESTION.

DO YOU HAVE A MINUTE?

TALK TO MY SECRETARY.
Function
Dominator tree
Dominators
Implicit loops
<table>
<thead>
<tr>
<th>Address</th>
<th>Instruction</th>
<th>Operand 1</th>
<th>Operand 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>00002275</td>
<td>mov</td>
<td>eax</td>
<td>ds:[off_1C018]</td>
</tr>
<tr>
<td>000022EB</td>
<td>mov</td>
<td>edi</td>
<td>0x1281C</td>
</tr>
<tr>
<td>000022F0</td>
<td>mov</td>
<td>ecx</td>
<td>0x5</td>
</tr>
<tr>
<td>000022F5</td>
<td>cld</td>
<td></td>
<td></td>
</tr>
<tr>
<td>000022F6</td>
<td>mov</td>
<td>ebx</td>
<td>ds:[eax]</td>
</tr>
<tr>
<td>000022F8</td>
<td>mov</td>
<td>esi</td>
<td>ebx</td>
</tr>
<tr>
<td>000022FA</td>
<td>repe cmpsb</td>
<td></td>
<td></td>
</tr>
<tr>
<td>000022FC</td>
<td>mov</td>
<td>ebx</td>
<td>0x0</td>
</tr>
<tr>
<td>00002301</td>
<td>jz</td>
<td></td>
<td>byte cs:loc_230D</td>
</tr>
</tbody>
</table>
...to this one
Is that enough?

I don't think I have your full attention.

It's Asok's turn to listen. If you say anything useful, he'll send us an instant message.

He's asleep.

He's employing heuristics.
Not enough

Of course not, more heuristics needed

```c
void *safe_strncpy(void *old_dest, void *src, int size){
  void *dst = realloc(old_dest, size +1);
  strncpy(dst, src, size);
  return dst;
}
```
Add your own

For static analysis we use

zynamics
www.zynamics.com

BinNavi
graph visualization
Questions!
Data Tainting
Example

Taint Source

movl 0x4[eax], ebx
PIN
Taint sources
Markings granularity
Propagation

\texttt{add eax, ebx, edx}
Output

Registers

Memory locations
Questions!
In-memory fuzzing
Example

esi = 0x30f064 → Original loc

Fuzzed loc → esi = 0x30f0A4

rep movs
Why?

DANGER
WRONG WAY
TURN BACK
Problems
Expertise and patience
Memory instability
False positives
False negatives

DON'T TALK TO ME I'M UNDERCOVER
Mutation loop insertion

Function 1 → Function 2 → Tested function → Function 3

mutate → Tested function → Function 4
Snapshot mutation restoration

- Function 1
- Function 2
- snapshot
- Tested function
- restore
- Function 3
- Function 4
What do we do?

• Hook image
• Hook functions
• Hook instructions
First approach
For instance...

30f064-30f068

ABCD

0x8a Y 0x00 K
Second approach
Example

30f064-30f068

ABCD

30f084-30f098

0x89 K D F 0x96
0x00 J K U Y W 0xA7
0xB8 0x00 0x10 A T N
0x00 0xD3
Code coverage
Score

\[
\frac{\text{BB}_{\text{executed}}}{\text{BB}_{\text{total}}}
\]

- Basic Blocks executed
- Total Basic Blocks
Halting

\[ C_{\text{evil}} = C_{\text{good}} + t \]

- Code coverage evil sample
- Code coverage good sample
- User-supplied threshold
How?

Good sample

Score

Evil sample

Score

Compare
What do we use?

- Code coverage
- Faults monitor
DEMO
Future – A reasoner
Thanks
Questions!
More Info

viozzo.wordpress.com

@_snagg

vincenzo.iozzo@zynamics.com