Introduction
0000

Our Solution
00000000000000

Conclusions
000

Hardware-Assisted Application Misbehavior Detection

Marcus Botacin   Paulo de Geus   André Grégio

XVIII SBSEG

2018

Agenda

Roteiro

Bugs

Undesirable

- **Safety**: Crashes.
- **Security**: Exploitation.

Countermeasures

- **Good Software Engineering**: Really ?
- **Fuzzing**: Too slow to cover all paths.
- **CFI**: Too specific to extend to general cases.

Alternative

- **Runtime Monitoring**: COTS binaries monitoring.

Introduction
oo●o

Our Solution
oooooooooooooo

Conclusions
ooo

Background

Program as a Finite State Machine

Input   Output
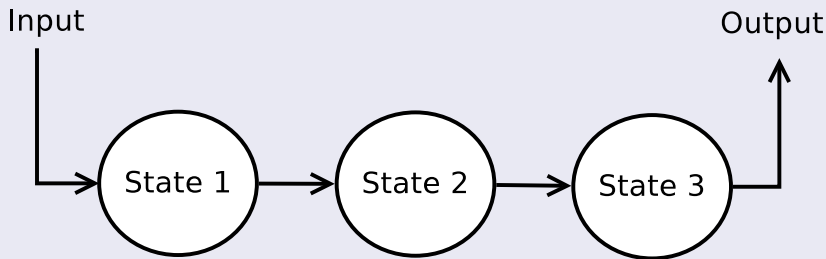

State 1 → State 2 → State 3

Figure: **Program as a Finite State Machine**. Data is inputted to an initial state and transitions lead to the final state, outputting the computation result.

Introduction
○○○●

Our Solution
○○○○○○○○○○○○○○

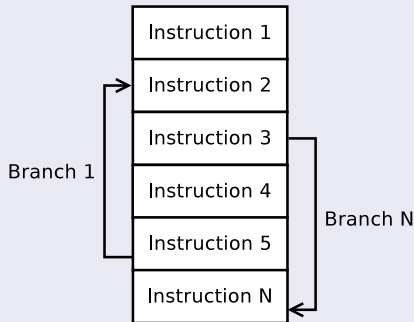Conclusions
○○○

Program in Memory

Branch as Transitions

Figure: **Program representation in memory**. Branch instructions are responsible for state transitions.

Roteiro

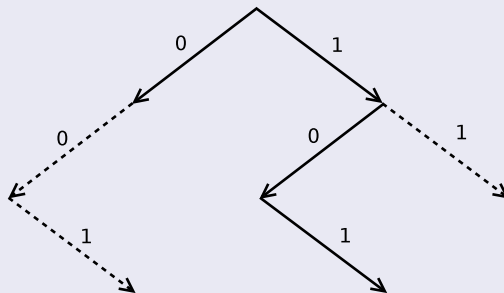# Roteiro

Our Solution

Tracking Expected Branches

Figure: **Expected Branches Policy**. The solid arrows correspond to paths previously seem, thus representing expected branches. The dotted arrows represent so-far unknown branches, which might indicate a misbehavior.

Key Idea

Our Solution

Learning Expected Branches

Figure: Automated learning. Flags 1 and 0 indicate, respectively, whether a given branch was expected (allowed) to occur or not.

Roteiro

1 **Introduction**

2 **Our Solution**
   - Key Idea
   - **Implementation**
   - Evaluation
   - Discussion

3 **Conclusions**

Introduction
0000

Our Solution
0000O●000OOOOOO

Conclusions
OOO

Implementation

Our Solution

Implementation

Table: ASLR-aware data collection. Offset normalization. Despite the distinct image base addresses, branch offsets are unique.

Branch	Execution 1	Execution 2	Execution N	Offset
I	0x7FF1D30	0x7FF3D30	0x7FF5D80	0x1D30
II	0x7FF1E30	0x7FF3E30	0x7FF5E80	0x1E30
II	0x7FF1EF0	0x7FF3EF0	0x7FF5F40	0x1EF0

Introduction
0000

Our Solution
0000000●00000000

Conclusions
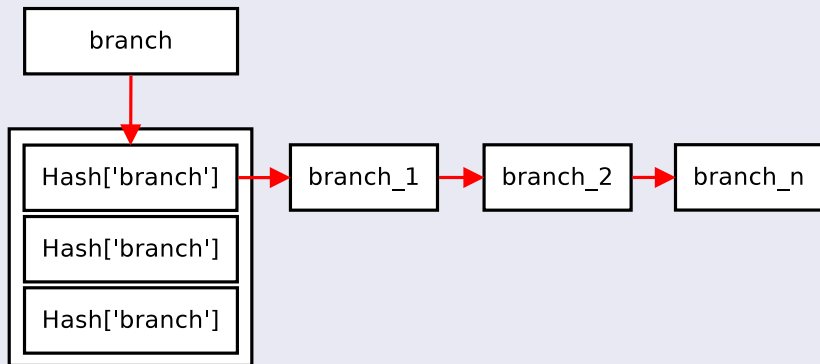000

Implementation

Our Solution

Implementation

Figure: Branch Database. Source addresses are used to index allowed
target addresses. Unidentified entries are considered as unexpected
branches.

Introduction
0000

Our Solution
0000000●0000000

Conclusions
000

Implementation

Detection Policies

Violation Detection


```
[BranchCluster] Starting looping
<11fb023,11fb037> (1)
<11fb037,11fb072> (1)
<11fb072,11fb072> (0)
<11fb072,11fb014> (1)
<11fb014,11fb023> (1)
<11fb023,11fb03c> (0)
<11fb03c,11fb04d> (0)
<11fb04d,11fb04d> (0)
Violation on 3 of last 4 branches
```

Figure: Misbehavior Detection. Solution detects violations using a threshold value over data from a moving window.
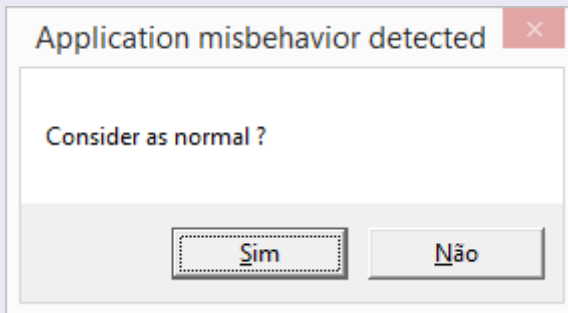
Introduction
0000

Our Solution
00000000●000000

Conclusions
000

Implementation

Our Solution

Implementation

Figure: Semi-supervised learning. Solution asks for user confirmation.

Introduction
0000

Our Solution
000000000●00000

Conclusions
000

Evaluation

Roteiro

Introduction
0000

Our Solution
0000000000●00000

Conclusions
000

Evaluation

Evaluation

Synthetic Example

Code 1: Validation code.

```c
main(){
 char str[MAX_STRING];
 int loop=0, opt=0;
 do{
  scanf("%d",&opt);
  if(opt>0){printf("Greater than zero\n");}
  elseif(opt<0){printf("Smaller than zero\n");}
  else{printf("Bad choice\n");scanf("%s",str);}
 }while(!loop);
 printf("Should never be executed\n");
```

Introduction
0000

Our Solution
000000000000000

Conclusions
000

Evaluation

Evaluation

Easy File Share

Code 2: Real application under a ROP-based attack. Differences between the expected and the observed branches.

```
Unexpected Branches: [0x150C, 0x1C80C, 0x13020]
Unexpected Branches: []
Unexpected Branches: [0x1731A, 0xD31A, 0x7C81A,
0x33B1A, 0x2AC1A, 0xFC21A, 0x12941A, 0x29A1A]
```

Introduction
0000

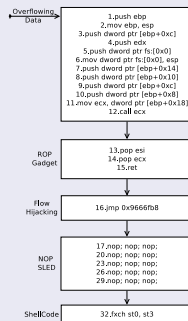Evaluation

Our Solution
0000000000000●00

Conclusions
000

Evaluation

Easy File Share

Figure: **Exploit Execution**

Roteiro

Introduction
0000

Our Solution
0000000000000●

Conclusions
000

Discussion

Discussion

Immediate Follow-up

- Enriching Crash Reports.

Future Developments

- Distributed Allowed Paths Identification.
- OS Self-Repair.
- Automatic Backup recovery.

Challenges

- Distinguish Exploits from Crashes.

Roteiro

Introduction
0000

Our Solution
00000000000000

Conclusions
0●0

Concluding Remarks

- **Advances**: Low-Overhead, Ruleless Misbehavior Detection.
- **Challenges**: Distinguish Exploitation from Crashes.
- **Future**: OS Self-repair.

Introduction
0000

Our Solution
0000000000000

Conclusions
00●

Questions ?

Contact Information

mfbotacin@inf.ufpr.br