A Security Benchmark Suite Exploring the Existing Vulnerabilities of a Computer System

Version: 0.1.0

Wei Song, Jiameng Ying and Boya Li
Institute of Information Engineeringat at the Chinese Academy of Sciences
89 Minzhuang Road, Haidian, Beijing 100093, P. R. China
{songwei,yingjiameng,liboya}@iie.ac.cn

October 15, 2018

Contents

Chapter 1

Introduction

Chapter 2

Overview of the Security Benchmark Suite

Chapter 3

Description of Test Cases

3.1 Control Flow Integrity (CFI)

- Forward-edge CFI
  - Call
    - 3.1.1: `wrong-num-arg-func`
    - 3.1.2: `wrong-num-arg-vtable`
    - 3.1.3: `wrong-type-arg-func`
    - 3.1.4: `wrong-num-func-vtable`
    - 3.1.5: `call-mid-func`
  - Jump
    - 3.1.6: `jump-mid-func`
- Backward-edge CFI
  - Return
    - 3.1.7: `return-non-call-site`
    - 3.1.8: `return-to-func`

3.1.1 `wrong-num-arg-func`

Description

Illegally call a function with mismatched number of arguements.

Vulnerability

Break the function calling convention.

Test result

return	*description*
0	**vulnerable**
other	might be safe

Known issues

None.

3.1.2 `wrong-num-arg-vtable`

Description

Illegally call a virtual function with mismatched number of arguements by modifying the VTable pointer.

Vulnerability

Break the data integrity of the Vtable pointer.

Test result

return	*description*
0	**vulnerable**
other	might be safe

Known issues

x86_64: Currently only works with object allocated on heap.

3.1.3 `wrong-type-arg-func`

Description

Illegally call a function with wrong types of arguements.

Vulnerability

Break the function calling convention.

Test result

return	*description*
0	**vulnerable**
other	might be safe

Known issues

None.

3.1.4 `wrong-num-func-vtable`

Description

Illegally call a fake virtual function with the VTable being replaced with another one of different number of virtual functions.

Vulnerability

Break the data integrity of the Vtable pointer.

Test result

return	*description*
0	**vulnerable**
other	might be safe

Known issues

x86_64: Currently only works with object allocated on heap.

3.1.5 `call-mid-func`

Description

Illegally call a fake function located at the middle of a function from `main()`.

Vulnerability

Illegal callee site.

Test result

return	*description*
0	**vulnerable**
other	might be safe

Known issues

None.

3.1.6 `jump-mid-func`

Description

Illegally jump from the `main()` function to the middle of another function.

Vulnerability

Break the execution compartment complied by most C/C++ programs.

Test result

return	*description*
0	**vulnerable**
other	might be safe

Known issues

None.

3.1.7 `return-non-call-site`

Description

Illegally modify the return address stored on the stack and then return to an none call-site position.

Vulnerability

Break the backward CFI and the integrity of the return address.

Test result

return	*description*
0	**vulnerable**
other	might be safe

Known issues

x86_64: The `rbp` register might be (with `-g`) or not be (with `-O2`) pushed to the stack. The return address is modified by embedded assembly using `rsp` as the base register. See `STACK_STRUCT` in the make file.

3.1.8 `return-to-func`

Description

Illegally modify the return address stored on the stack and directly return to another function.

Vulnerability

Break the backward CFI and the integrity of the return address.

Test result

return	*description*
0	**vulnerable**
other	might be safe

Known issues

Chapter 4

Remaining Issues

- `wrong-num-arg-func` 3.1.1: test for arguements passed on stack.
- `wrong-type-arg-func` 3.1.3: more importantly, test (data/code) pointer to integer.
- `wrong-num-arg-vtable` 3.1.2: known issues.
- `wrong-num-func-vtable` 3.1.4: known issues.
- call a unvisible function (call a local function from outside).
- differentiate between global data, heap and stack.