

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 Engineering at the Chinese Academy of Sciences
89 Minzhuang Road, Haidian, Beijing 100093, P. R. China
{songwei, yingjiameng, liboya}@iie.ac.cn

September 23, 2018

Contents

1	Introduction	5
2	Overview of the Security Benchmark Suite	7
3	Description of Test Cases	9
3.1	Control Flow Integrity (CFI)	9
3.1.1	wrong-num-arg-func	10
3.1.2	wrong-num-arg-vtable	11
3.1.3	wrong-type-arg-func	12
3.1.4	jump-mid-func	13
3.1.5	return-non-call-site	14
3.1.6	return-to-func	15
4	Remaining Issues	17

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
 - Jump
 - * 3.1.4: jump-mid-func
- Backward-edge CFI
 - Return
 - * 3.1.5: return-non-call-site
 - * 3.1.6: return-to-func

3.1.1 wrong-num-arg-func

Description

Illegally call a function with mismatched number of arguments.

Vulnerability

Break the function calling convention.

Test result

<i>return</i>	<i>description</i>
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 arguments by modifying the VTable pointer.

Vulnerability

Break the data integrity of the Vtable pointer.

Test result

<i>return</i>	<i>description</i>
0	vulnerable
other	might be safe

Known issues

x86_64: Only work with `-g` because the modifying of the VTable pointer is optimized away with `-O2`.

3.1.3 wrong-type-arg-func

Description

Illegally call a function with wrong types of arguments.

Vulnerability

Break the function calling convention.

Test result

<i>return</i>	<i>description</i>
0	vulnerable
other	might be safe

Known issues

None.

3.1.4 `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

<i>return</i>	<i>description</i>
0	vulnerable
other	might be safe

Known issues

None.

3.1.5 return-non-call-site

Description

Illegally modify the return address stored on the stack and then return to a non-call-site position.

Vulnerability

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

Test result

<i>return</i>	<i>description</i>
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.

3.1.6 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

<i>return</i>	<i>description</i>
0	vulnerable
other	might be safe

Known issues

Chapter 4

Remaining Issues

- `wrong-num-arg-func` 3.1.1: test for arguments 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.
- call a unvisible function (call a local function from outside).
- differentiate between global data, heap and stack.