# CMPT 479/980

Milestone Presentation

Defense: Detecting and repairing control-flow hijacking attacks

## Content

- Motivation
- Problem
- 2 ideas
- Solution
- Challenges
- Results
- Learned Lessons

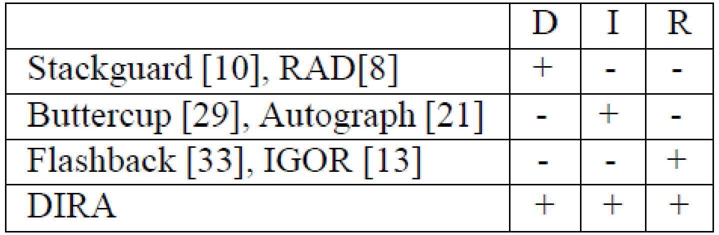
## Motivation

- Control-hijacking attack is harmful
- Based on the paper "DIRA: Automatic Detection, Identification, and Repair of Control-Hijacking Attacks"[1]. (Smirnov, A., & Chiueh, T. C.)
- A comprehensive protection strategy which consists of:

  **(D)etection**,

  **(I)dentification**

  and **(R)ecovery**.



| | D | I | R |
|---|---|---|---|
| Stackguard [10], RAD[8] | + | - | - |
| Buttercup [29], Autograph [21] | - | + | - |
| Flashback [33], IGOR [13] | - | - | + |
| DIRA | + | + | + |

**Table 1. Previous work addressing problems of attack (D)etection, (I)dentification, and (R)epair.**

## Problem

- For this project we will omit the identification part.
- what our program should do:
  - Detect control-hijacking attack exploiting control-sensitive data (function-pointer, return address)
  - Repair: use memory logging and tree traversal (function call tree).

## 2 ideas

- LLVM

  compiler infrastructure framework designed for compile-time, link-time, and run time optimizations

  LLVM IR (intermiate representation)

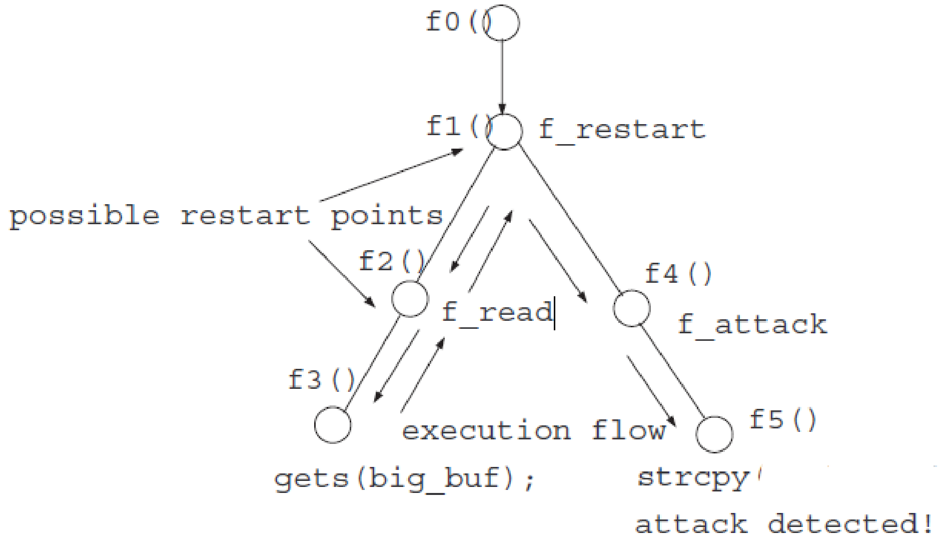
- GCC plugin

  loadable modules that provide extra features to the compiler
  since GCC 4.5

## Solution

- Detection:
    - Buffer the return address at the function prologue, and check them with the value at function epilogue.  Dectect a mismatch.
    - Buffer function pointers once it is declared or modified. Check it with the buffered value every time a function pointer is to be used in a function call.

- Repair:
    - Determine a restart point, taking both the point that read in malicious data and the point the attack is detected into account. (later)


f0()
f1()
f_restart
possible restart points
f2()
f_read
f4()
f_attack
f3()
execution flow
f5()
gets(big_buf);
strcpy(
attack detected!

## Challenges

- Lack of GCC-plugin documentation
  - have to refer to source code
- GCC-Plugin API changes at each release of GCC
  - `Error: macro "gen_rtx_SET" requires 3 arguments, but only 2 given.` on my system with gcc 7.5 as host compiler
  - Use gen_rtx_set instead of gen_rtx_SET. The former is a wrapper macro that handles the difference between GCC versions implementing the latter.
- Remote collaboration is difficult and time-consuming

## Results

- Detection Memory Log
- GCC-Plugin mechenism
- To be done: attack repair

## Results

Detection Log

| type | out-dated | var_name | var_addr |
|---|---|---|---|

Recovery Log

| timestamp | function_call | var_name | var_type | size | val_from | val_to |
|---|---|---|---|---|---|---|

## Learned Lessons

- Make timeline more realistic
  - Taking into account of unexpected problems (eg. stuck on an error, time spent searching for documentations)
- Improve the ability to deal with limited documentation

## Thanks!