Forced ARM CFI Through DBM Utilization.pdf

Gili Yankovitch

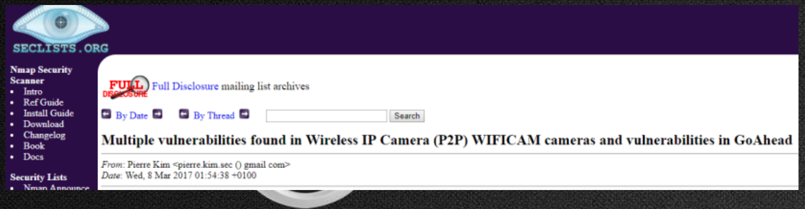
- Security Researcher
- Maker
- Linux Kernel Disciple
  - Co-Founder KernelTLV
- Embedded OS Ninja

@Cytingale

Knightingales

kg@cyberknights.io

IoT - Internet of Trickery



SECLISTS.ORG

Nmap Security Scanner
- Intro
- Ref Guide
- Install Guide
- Download
- Changelog
- Book
- Docs

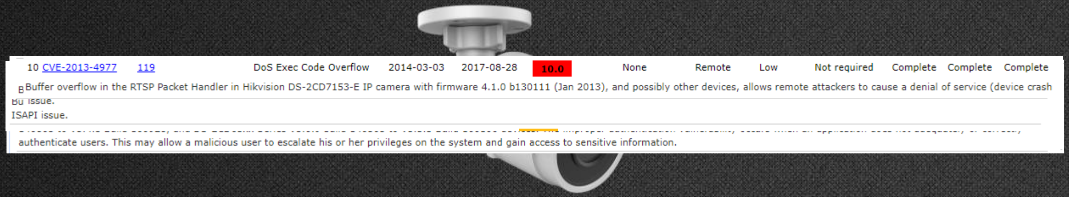
Security Lists
- Nmap Announce

Full Disclosure mailing list archives

By Date By Thread Search

Multiple vulnerabilities found in Wireless IP Camera (P2P) WIFICAM cameras and vulnerabilities in GoAhead

From: Pierre Kim <pierre.kim.sec () gmail com>
Date: Wed, 8 Mar 2017 01:54:38 +0100

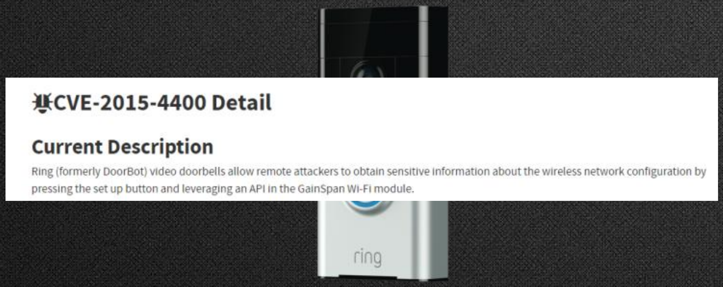


10	CVE-2013-4977	119	DoS Exec Code Overflow	2014-03-03	2017-08-28	10.0	None	Remote	Low	Not required	Complete	Complete	Complete

Buffer overflow in the RTSP Packet Handler in Hikvision DS-2CD7153-E IP camera with firmware 4.1.0 b130111 (Jan 2013), and possibly other devices, allows remote attackers to cause a denial of service (device crash

ISAPI issue.

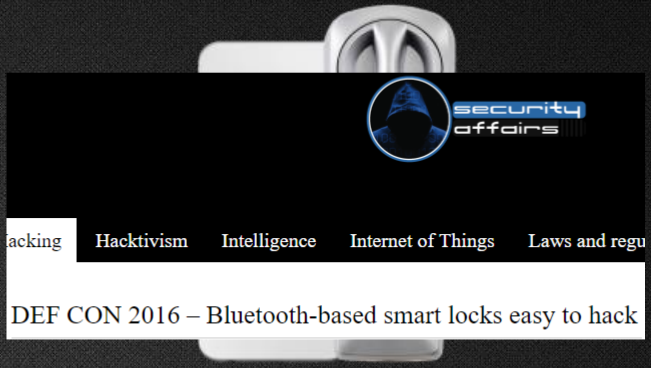
authenticate users. This may allow a malicious user to escalate his or her privileges on the system and gain access to sensitive information.



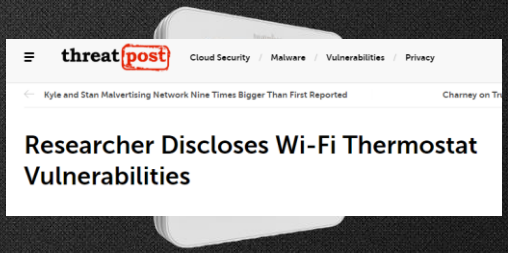
CVE-2015-4400 Detail

Current Description

Ring (formerly DoorBot) video doorbells allow remote attackers to obtain sensitive information about the wireless network configuration by pressing the set up button and leveraging an API in the GainSpan Wi-Fi module.



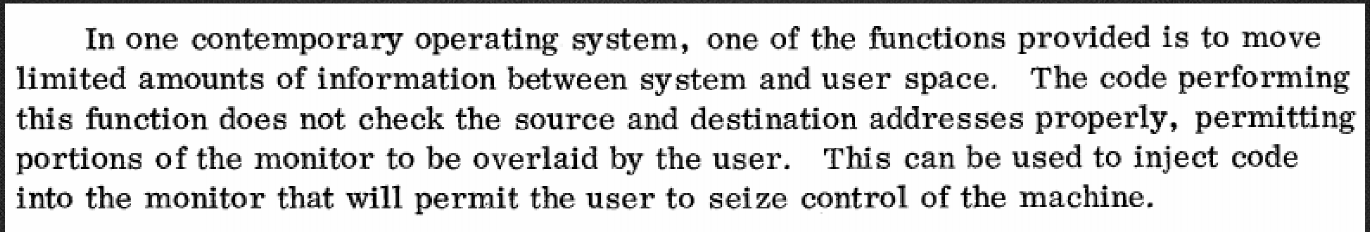
DEF CON 2016 – Bluetooth-based smart locks easy to hack



threatpost
Cloud Security / Malware / Vulnerabilities / Privacy
Kyle and Stan Malvertising Network Nine Times Bigger Than First Reported
Charney on Tr

Researcher Discloses Wi-Fi Thermostat Vulnerabilities

Aren’t you tired of Overflows?



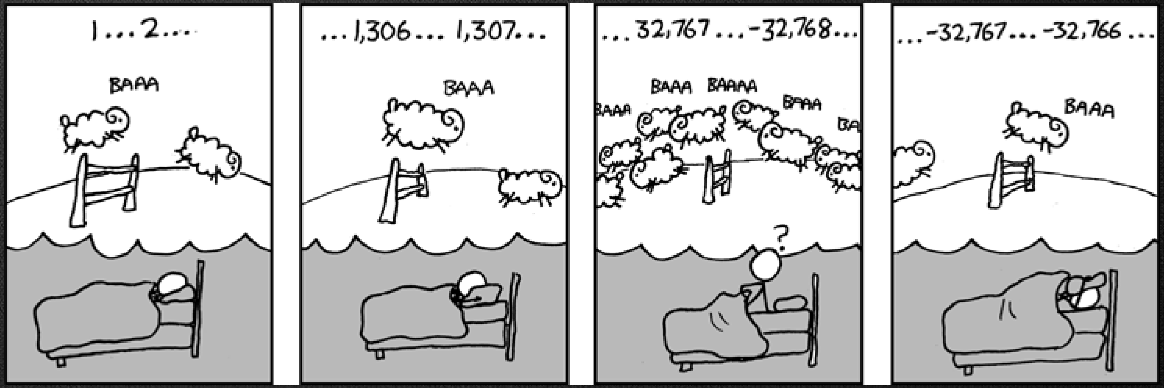
In one contemporary operating system, one of the functions provided is to move limited amounts of information between system and user space. The code performing this function does not check the source and destination addresses properly, permitting portions of the monitor to be overlaid by the user. This can be used to inject code into the monitor that will permit the user to seize control of the machine.

Can you guess when was this written?

The answer is **1972**!

Haven’t we suffered enough?

- Stack (based) Overflow
- Double Free
- Use After Free
- Heap Overflow
- …



Problem

- Awareness (Lack of)
- Development Schedule
- Don’t Care
- Awareness
- Awareness
- Awareness

Solution?

- Education
- Education
- Education
- Education
- Education
- Education
- … Or we can force them ☺

DBM – Dynamic Binary Modification

- Examples
  - DynamoRIO
  - Pin

- Challenges:
  - Statically Linked Binaries
  - Non-PIE

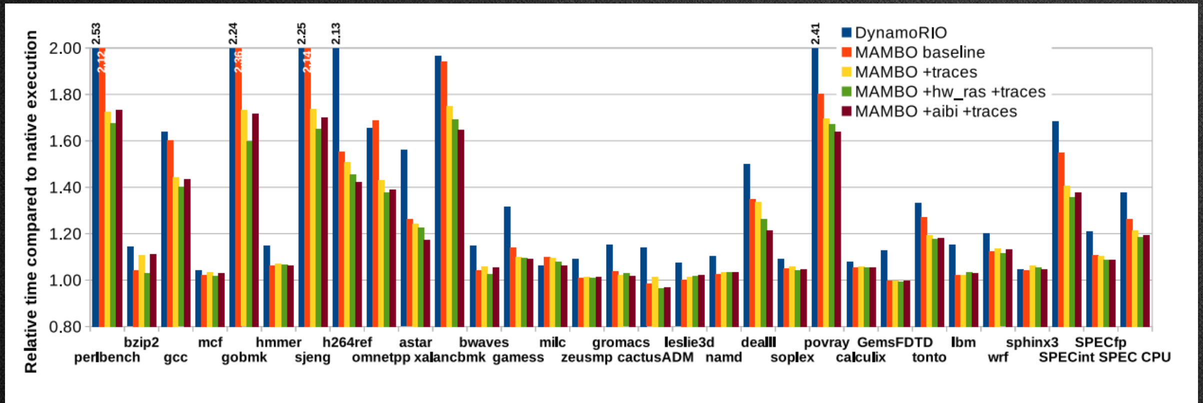
For fuzzing

High overhead

Almost **nothing** for embedded platforms

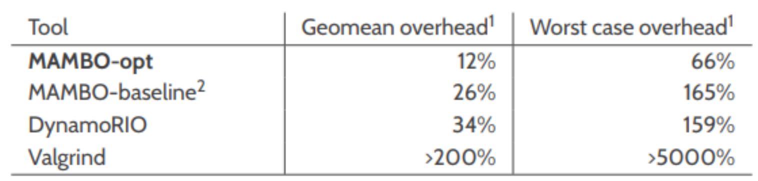
MAMBO

- *“A low-overhead dynamic binary instrumentation and modification tool for ARM (now with both AArch32 and AArch64 support)”*
- *Can be praised here: https://github.com/beehive-lab/mambo*
- *Much love and thanks to Cosmin **Gorgovan**, **Amanieu d'Antras**, **Mikel Luján** on this beautiful project.*
- *Performs very good on the average case*



Relative execution time for SPEC CPU2006 on ODROID-XU3 (Cortex A7 in-order)

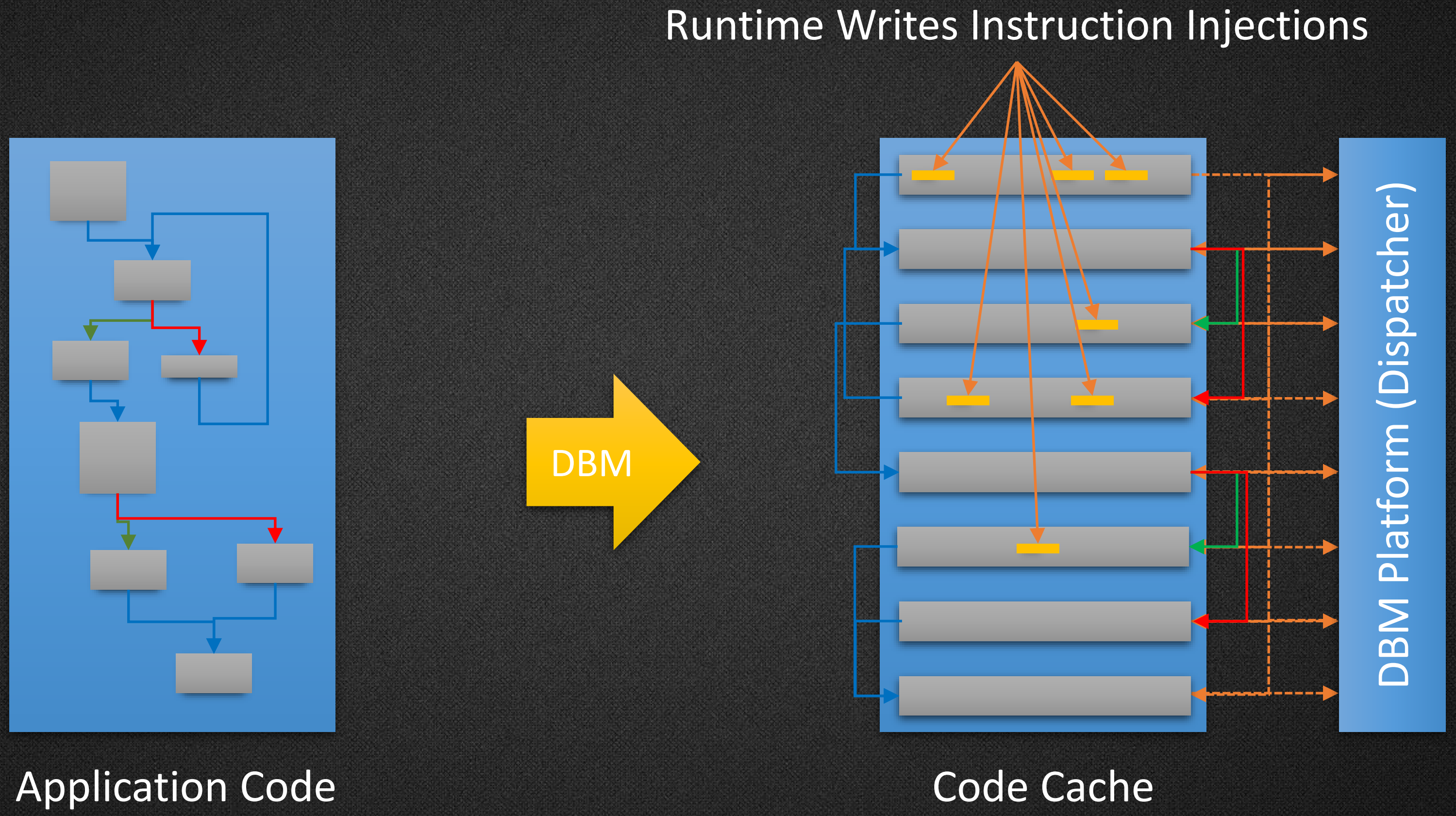
More performance graphs here
https://www.research.manchester.ac.uk/portal/files/65557332/cosmin_mambo_icpe2018.pdf



Tool	Geomean overhead[1]	Worst case overhead[1]
MAMBO-opt	12%	66%
MAMBO-baseline[2]	26%	165%
DynamoRIO	34%	159%
Valgrind	>200%	>5000%

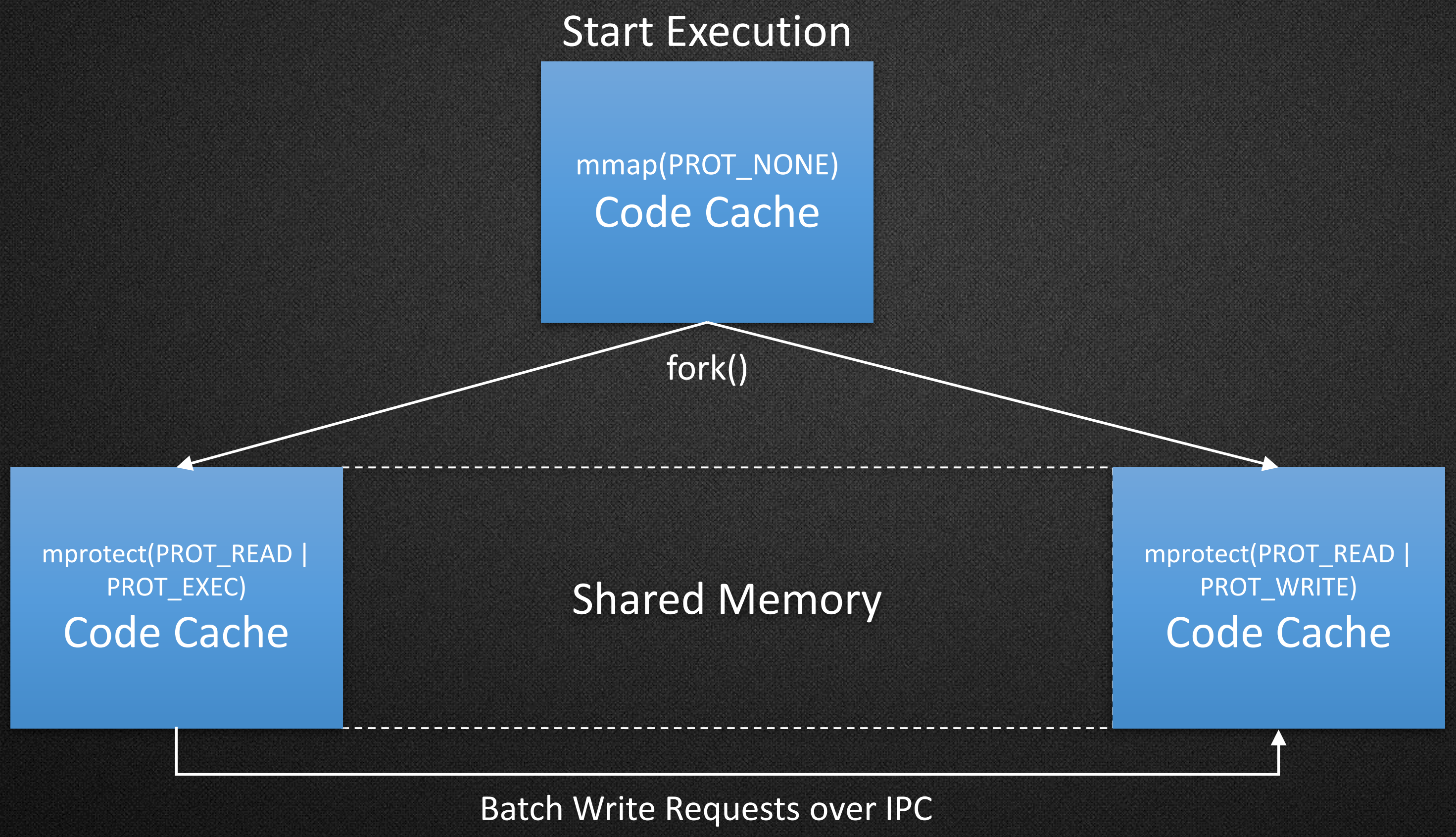
Performance of DBM tools for ARM

How a DBM works


Runtime Writes Instruction Injections
DBM
DBM Platform (Dispatcher)
Application Code
Code Cache

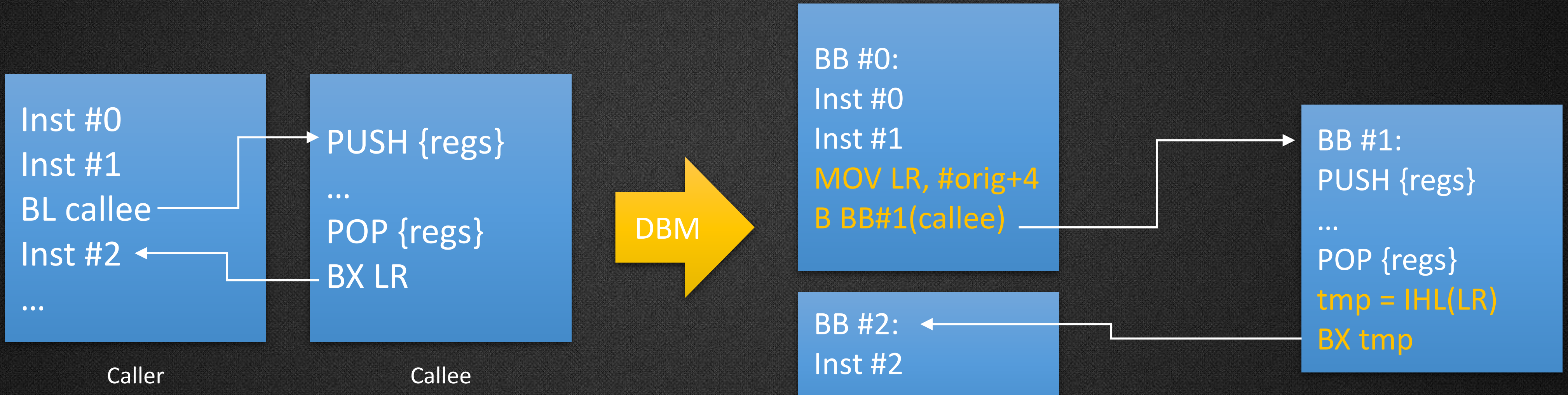
The JIT Problem

- JITs are a pain because of RWX


Start Execution
mmap(PROT_NONE)
Code Cache
fork()
mprotect(PROT_READ | PROT_EXEC)
Code Cache
Shared Memory
mprotect(PROT_READ | PROT_WRITE)
Code Cache
Batch Write Requests over IPC

…Or just interleave mprotect(W^X) for every BB

How a DBM works (cont)



Inst #0
Inst #1
BL callee
Inst #2
…

Caller

PUSH {regs}
…
POP {regs}
BX LR

Callee

BB #0:
Inst #0
Inst #1
MOV LR, #orig+4
B BB#1(callee)

BB #2:
Inst #2
…

BB #1:
PUSH {regs}
…
POP {regs}
tmp = IHL(LR)
BX tmp

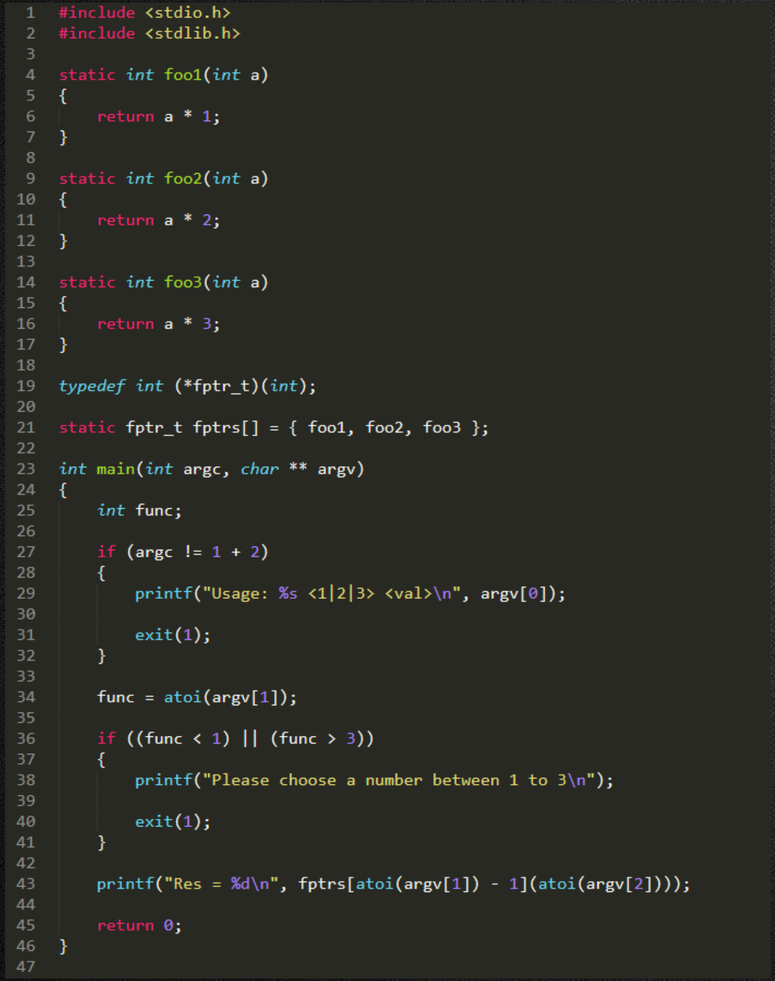
Original Instructions

Translated Instructions

Enforcing Mitigations - Requirements

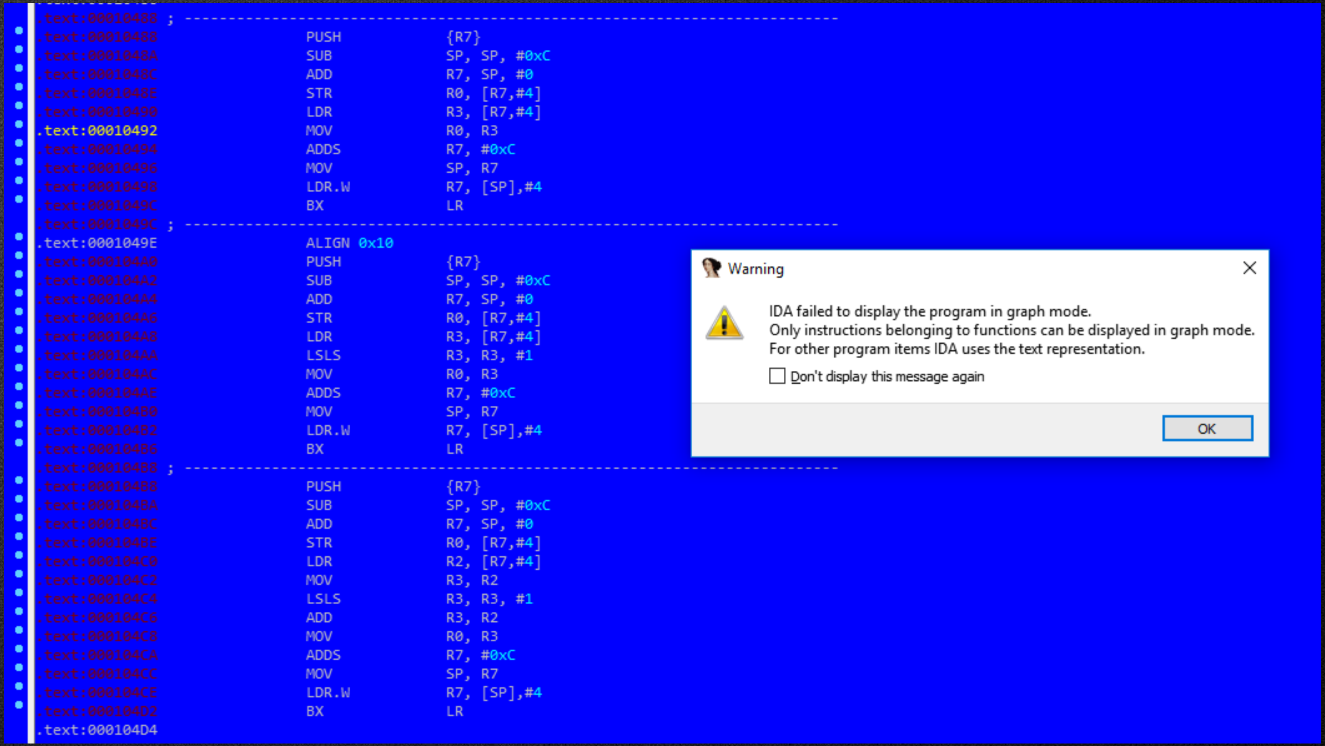
- Preprocessed ELF Functions Table
  - Hook thread start
  - Preprocessing ELF during startup
  - Use all information available to map ELF Functions
    - Use cheap tricks and heuristics to find functions

Trouble in Paradise



```c
#include <stdio.h>
#include <stdlib.h>

static int foo1(int a)
{
    return a * 1;
}

static int foo2(int a)
{
    return a * 2;
}

static int foo3(int a)
{
    return a * 3;
}

typedef int (*fptr_t)(int);

static fptr_t fptrs[] = { foo1, foo2, foo3 };

int main(int argc, char ** argv)
{
    int func;

    if (argc != 1 + 2)
    {
        printf("Usage: %s <1|2|3> <val>\n", argv[0]);

        exit(1);
    }

    func = atoi(argv[1]);

    if ((func < 1) || (func > 3))
    {
        printf("Please choose a number between 1 to 3\n");

        exit(1);
    }

    printf("Res = %d\n", fptrs[atoi(argv[1]) - 1](atoi(argv[2])));

    return 0;
}
```

- IDA **Failed** in finding these functions



```
.text:00010488 ; ---------------------------------------------------------------------------
.text:00010488                 PUSH            {R7}
.text:0001048A                 SUB             SP, SP, #0xC
.text:0001048C                 ADD             R7, SP, #0
.text:0001048E                 STR             R0, [R7,#4]
.text:00010490                 LDR             R3, [R7,#4]
.text:00010492                 MOV             R0, R3
.text:00010494                 ADDS            R7, #0xC
.text:00010496                 MOV             SP, R7
.text:00010498                 LDR.W           R7, [SP],#4
.text:0001049C                 BX              LR
.text:0001049C ; ---------------------------------------------------------------------------
.text:0001049E                 ALIGN 0x10
.text:000104A0                 PUSH            {R7}
.text:000104A2                 SUB             SP, SP, #0xC
.text:000104A4                 ADD             R7, SP, #0
.text:000104A6                 STR             R0, [R7,#4]
.text:000104A8                 LDR             R3, [R7,#4]
.text:000104AA                 LSLS            R3, R3, #1
.text:000104AC                 MOV             R0, R3
.text:000104AE                 ADDS            R7, #0xC
.text:000104B0                 MOV             SP, R7
.text:000104B2                 LDR.W           R7, [SP],#4
.text:000104B6                 BX              LR
.text:000104B8 ; ---------------------------------------------------------------------------
.text:000104B8                 PUSH            {R7}
.text:000104BA                 SUB             SP, SP, #0xC
.text:000104BC                 ADD             R7, SP, #0
.text:000104BE                 STR             R0, [R7,#4]
.text:000104C0                 LDR             R2, [R7,#4]
.text:000104C2                 MOV             R3, R2
.text:000104C4                 LSLS            R3, R3, #1
.text:000104C6                 ADD             R3, R2
.text:000104C8                 MOV             R0, R3
.text:000104CA                 ADDS            R7, #0xC
.text:000104CC                 MOV             SP, R7
.text:000104CE                 LDR.W           R7, [SP],#4
.text:000104D2                 BX              LR
.text:000104D4
```

IDA failed to display the program in graph mode.
Only instructions belonging to functions can be displayed in graph mode.
For other program items IDA uses the text representation.

□ Don't display this message again

... This is what the cheap tricks are for.

Cheap Tricks

ELF Functions Table
.text:0010464
.text:0010488
…

Disassemble from start of .text

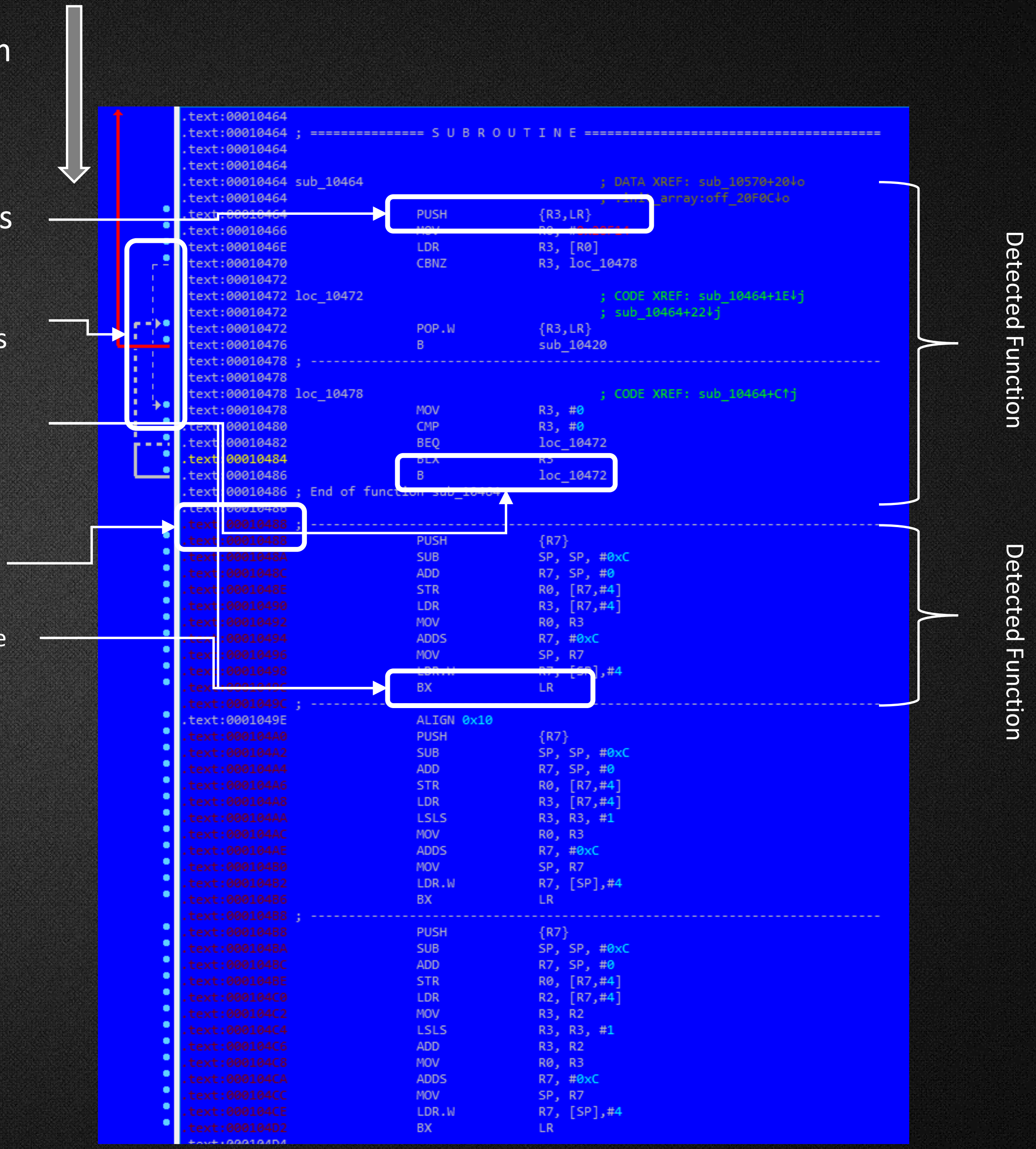
Look for function header signatures

Follow forward branches to expand function boundaries

Function boundaries when no other branch exceeds a branch backward instruction

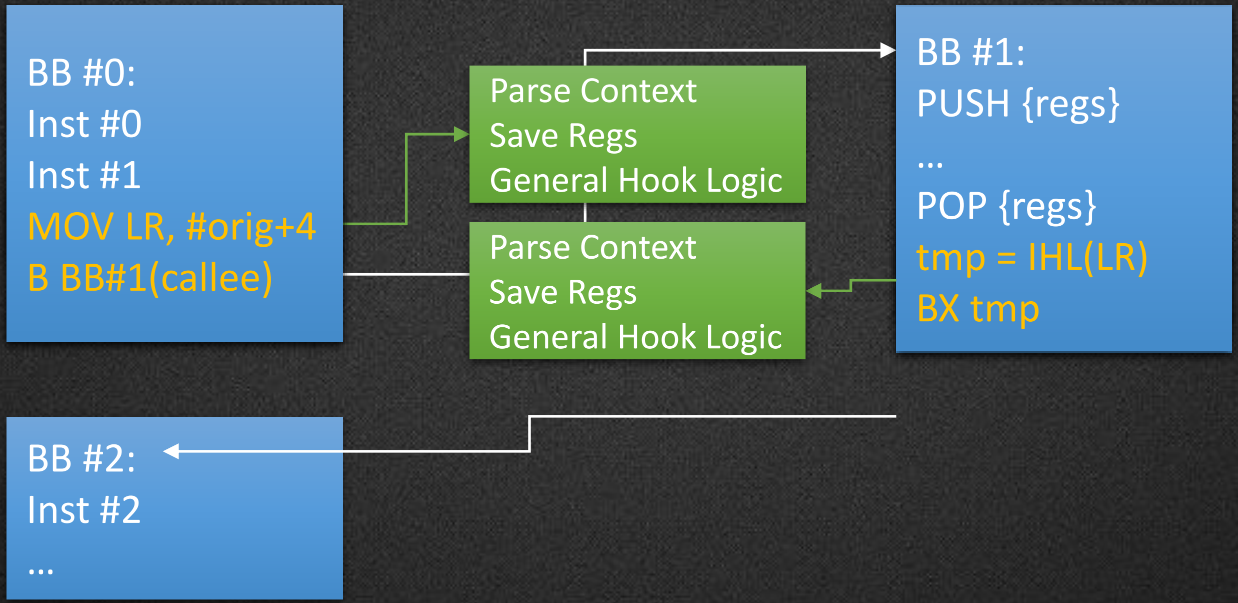
Start a new function

Function tail signature

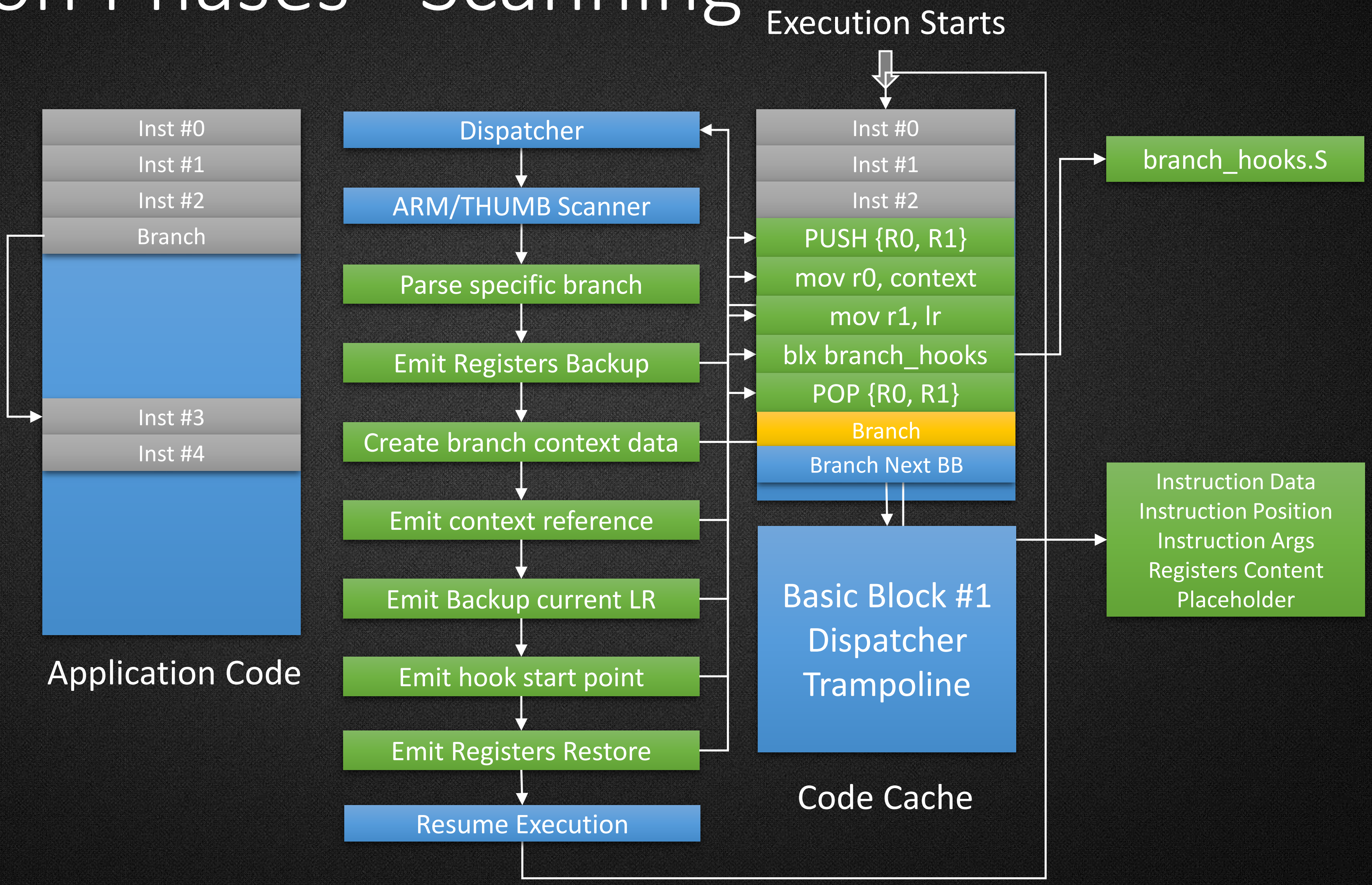


```
.text:00010464
.text:00010464 ; =============== S U B R O U T I N E =======================================
.text:00010464
.text:00010464
.text:00010464 sub_10464                               ; DATA XREF: sub_10570+20↓o
.text:00010464                                         ; .init_array:off_20F0C↓o
.text:00010464                 PUSH            {R3,LR}
.text:00010466                 MOV             R0, #0x20F14
.text:0001046E                 LDR             R3, [R0]
.text:00010470                 CBNZ            R3, loc_10478
.text:00010472
.text:00010472 loc_10472                               ; CODE XREF: sub_10464+1E↓j
.text:00010472                                         ; sub_10464+22↓j
.text:00010472                 POP.W           {R3,LR}
.text:00010476                 B               sub_10420
.text:00010478 ; ---------------------------------------------------------------------------
.text:00010478
.text:00010478 loc_10478                               ; CODE XREF: sub_10464+C↑j
.text:00010478                 MOV             R3, #0
.text:00010480                 CMP             R3, #0
.text:00010482                 BEQ             loc_10472
.text:00010484                 BLX             R3
.text:00010486                 B               loc_10472
.text:00010486 ; End of function sub_10464
.text:00010486
.text:00010488 ; ---------------------------------------------------------------------------
.text:00010488                 PUSH            {R7}
.text:0001048A                 SUB             SP, SP, #0xC
.text:0001048C                 ADD             R7, SP, #0
.text:0001048E                 STR             R0, [R7,#4]
.text:00010490                 LDR             R3, [R7,#4]
.text:00010492                 MOV             R0, R3
.text:00010494                 ADDS            R7, #0xC
.text:00010496                 MOV             SP, R7
.text:00010498                 LDR.W           R7, [SP],#4
.text:0001049C                 BX              LR
.text:0001049C ; ---------------------------------------------------------------------------
.text:0001049E                 ALIGN 0x10
.text:000104A0                 PUSH            {R7}
.text:000104A2                 SUB             SP, SP, #0xC
.text:000104A4                 ADD             R7, SP, #0
.text:000104A6                 STR             R0, [R7,#4]
.text:000104A8                 LDR             R3, [R7,#4]
.text:000104AA                 LSLS            R3, R3, #1
.text:000104AC                 MOV             R0, R3
.text:000104AE                 ADDS            R7, #0xC
.text:000104B0                 MOV             SP, R7
.text:000104B2                 LDR.W           R7, [SP],#4
.text:000104B6                 BX              LR
.text:000104B8 ; ---------------------------------------------------------------------------
.text:000104B8                 PUSH            {R7}
.text:000104BA                 SUB             SP, SP, #0xC
.text:000104BC                 ADD             R7, SP, #0
.text:000104BE                 STR             R0, [R7,#4]
.text:000104C0                 LDR             R2, [R7,#4]
.text:000104C2                 MOV             R3, R2
.text:000104C4                 LSLS            R3, R3, #1
.text:000104C6                 ADD             R3, R2
.text:000104C8                 MOV             R0, R3
.text:000104CA                 ADDS            R7, #0xC
.text:000104CC                 MOV             SP, R7
.text:000104CE                 LDR.W           R7, [SP],#4
.text:000104D2                 BX              LR
.text:000104D4
```

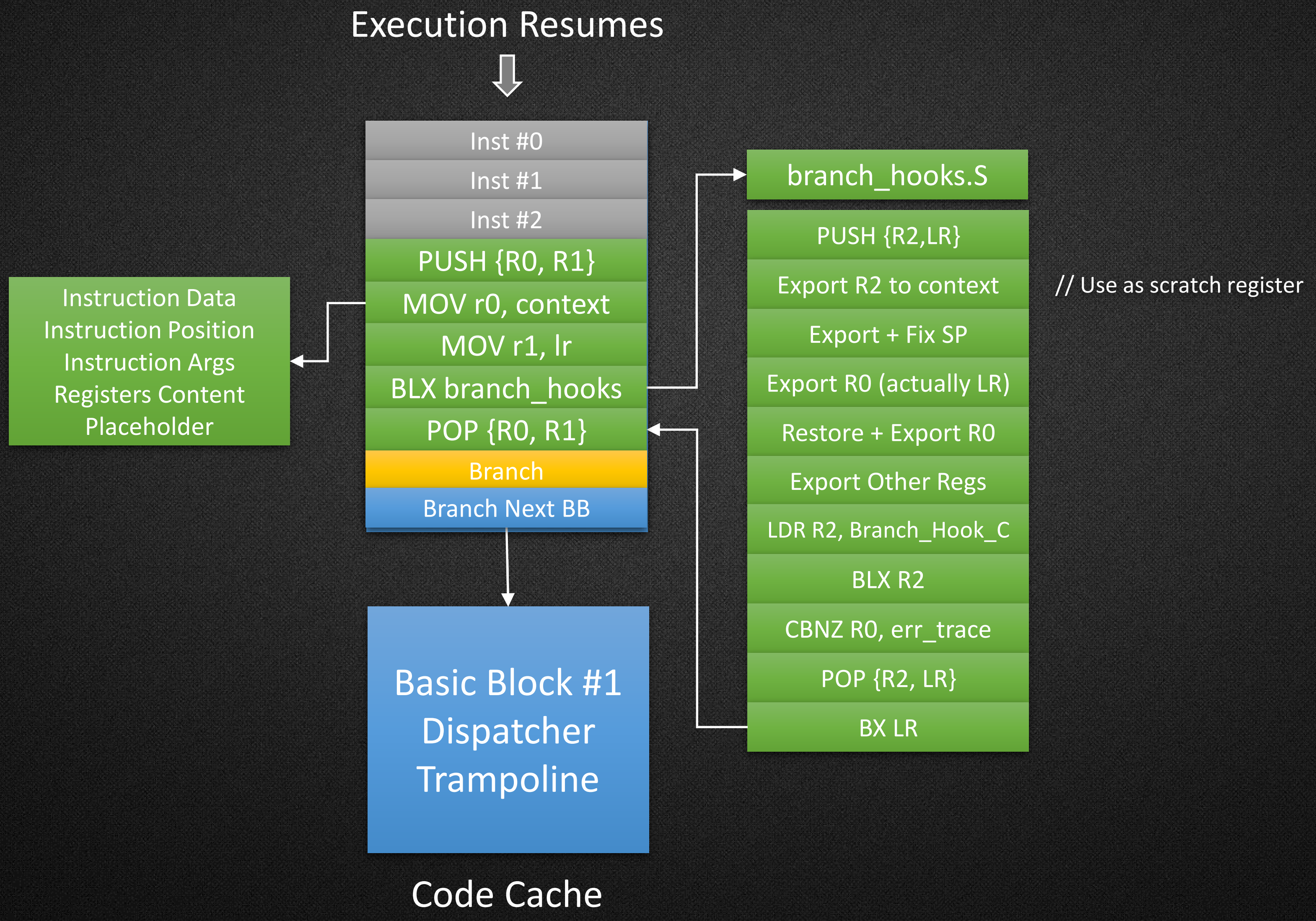
Enforcing Memory Mitigations



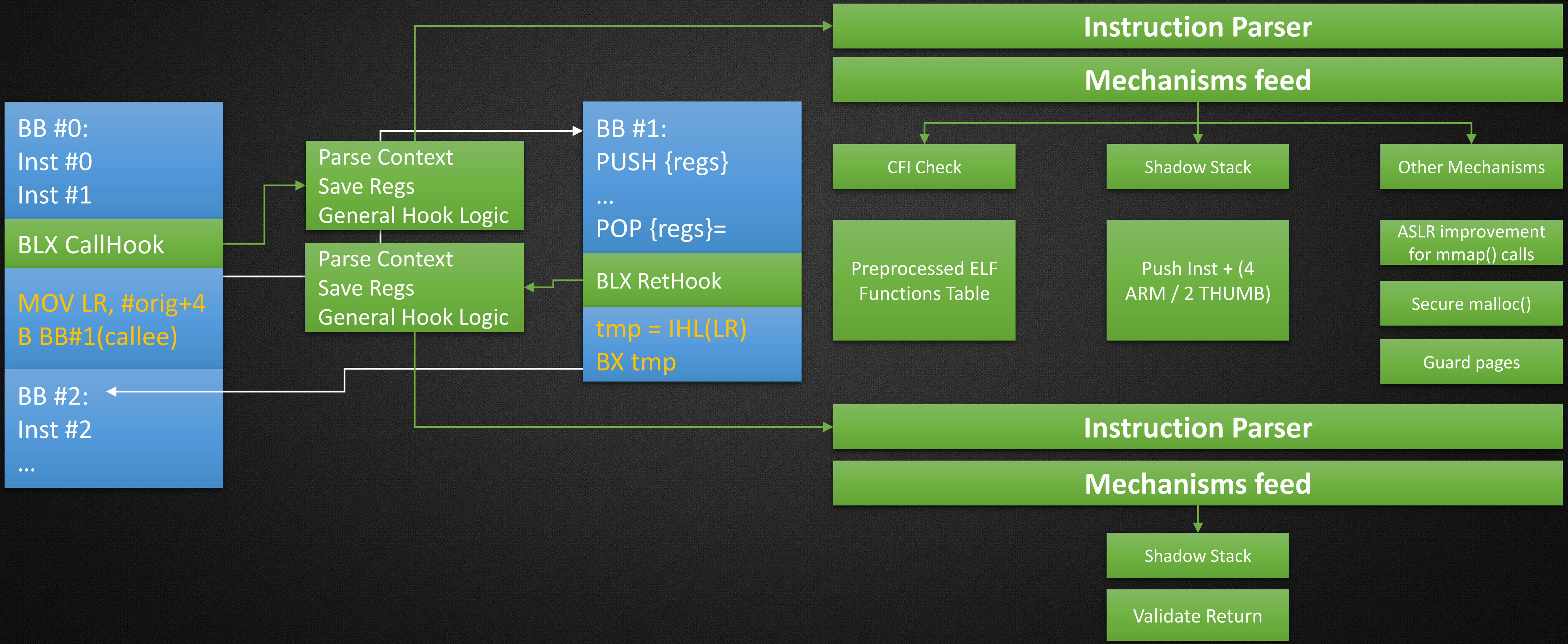
Protection Phases - Scanning



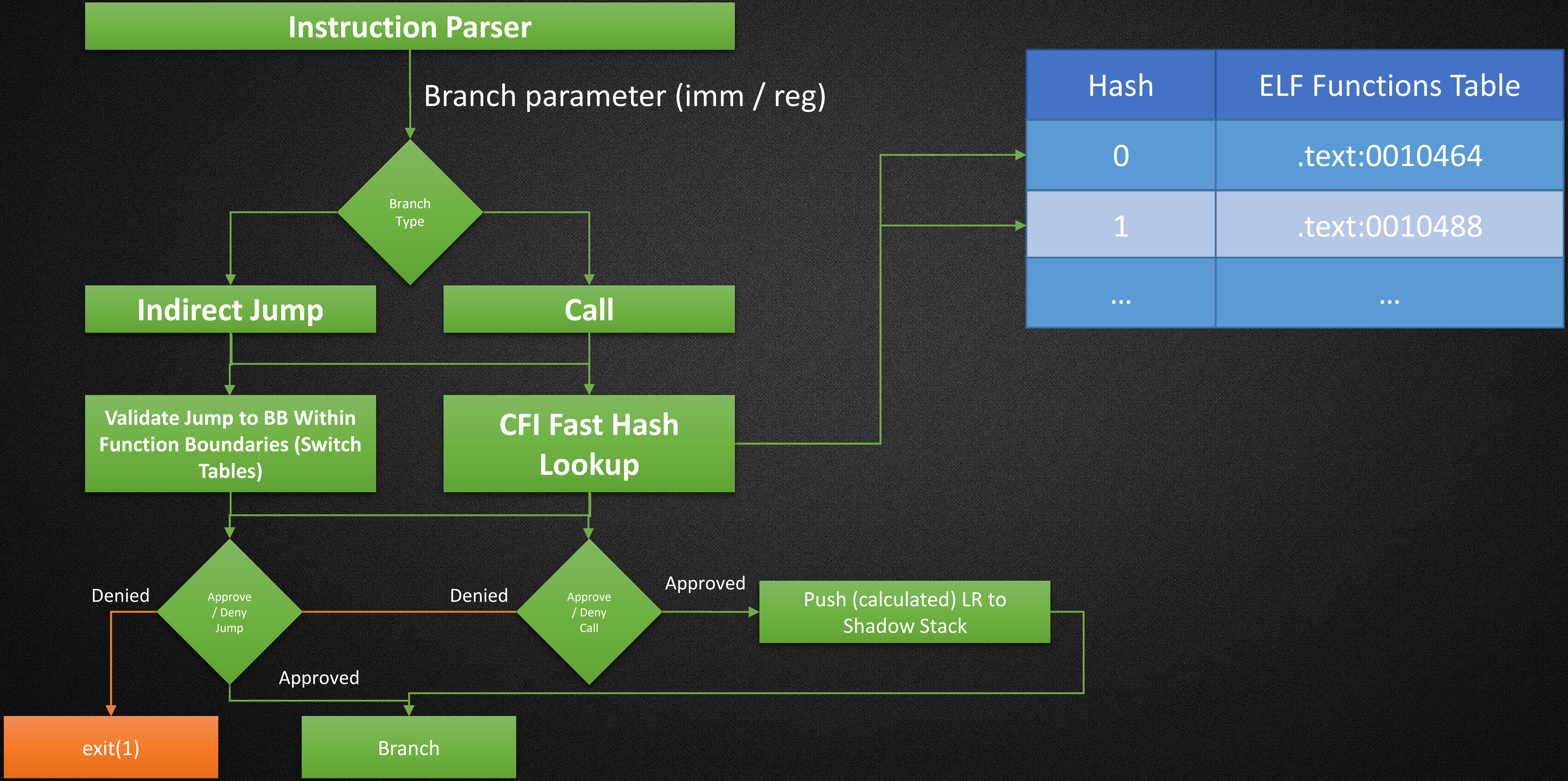
Protection Phases - Running



Enforcing Memory Mitigations

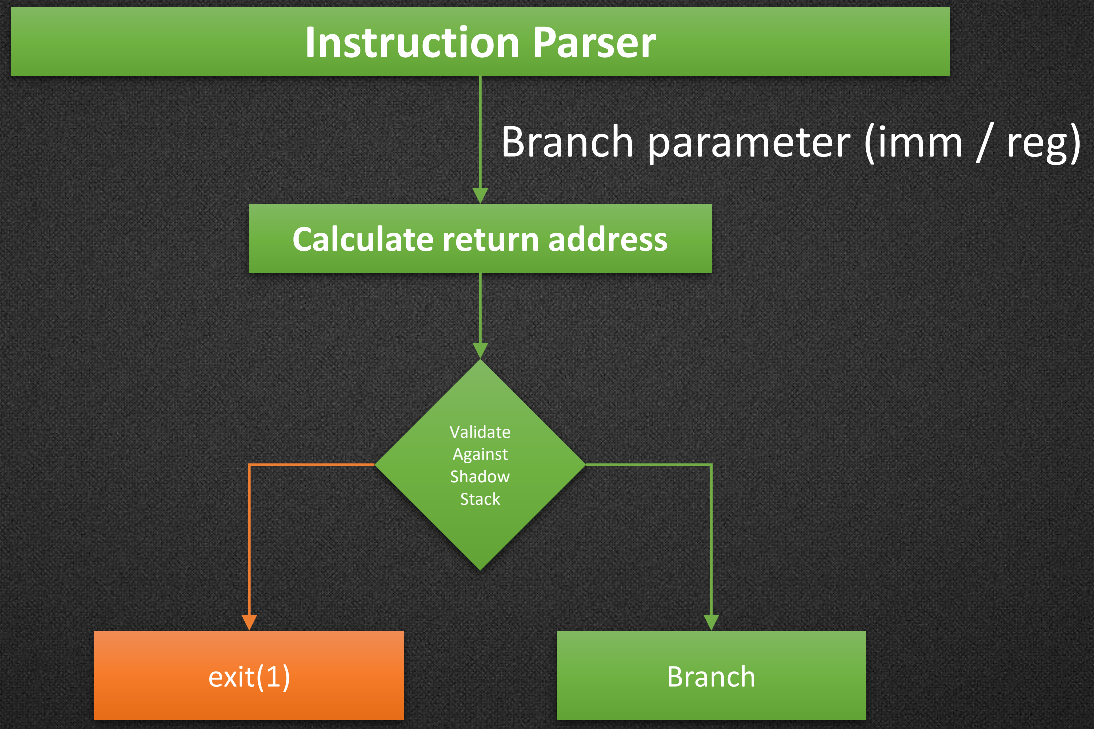


Coarse Grained CFI



Hash	ELF Functions Table
0	.text:0010464
1	.text:0010488
…	…

Coarse Grained CFI


Instruction Parser

Branch parameter (imm / reg)

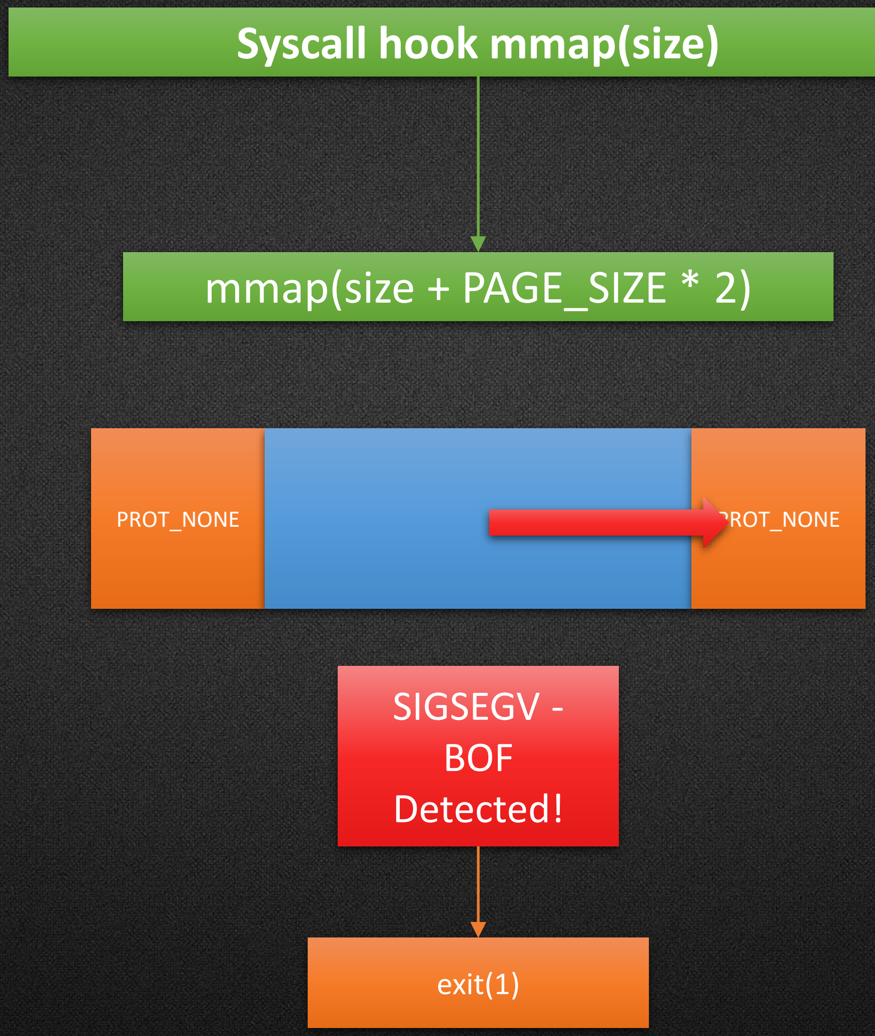
Calculate return address

Validate Against Shadow Stack

exit(1)

Branch

Other Mechanisms (Guard Pages)



Syscall hook mmap(size)

mmap(size + PAGE_SIZE * 2)

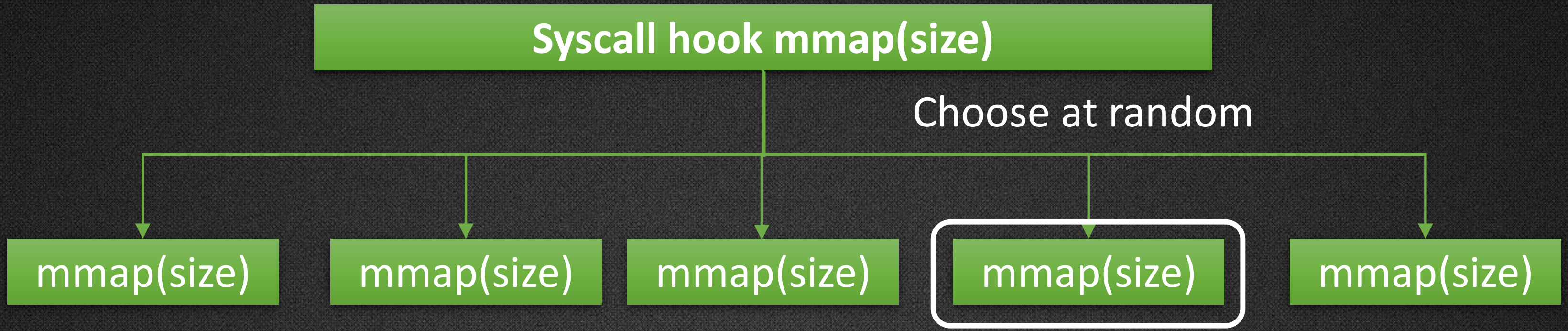
PROT_NONE

PROT_NONE

SIGSEGV - BOF Detected!

exit(1)

Other Mechanisms (ASLR)



Syscall hook mmap(size)

Choose at random

mmap(size)

mmap(size)

mmap(size)

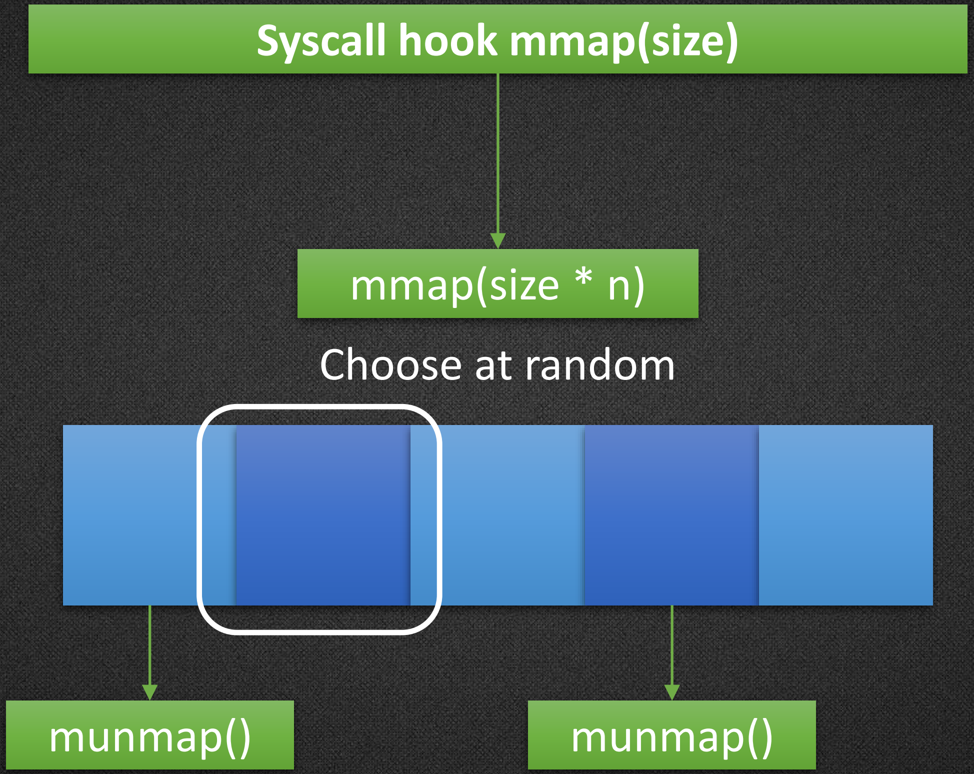
mmap(size)

mmap(size)

Multiple syscall overhead dependent on **n** calls

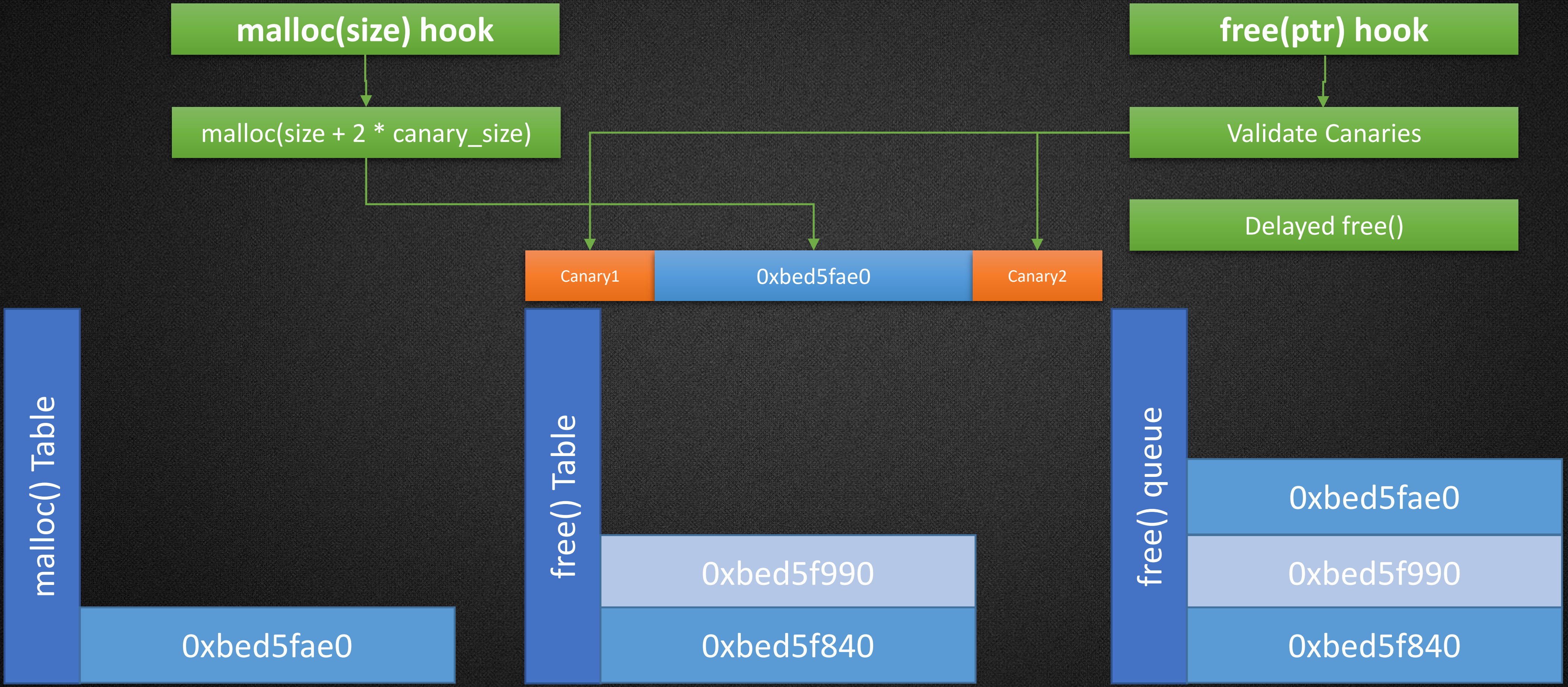
Memory Fragmentation ☹

Other Mechanisms (ASLR)



Better performance independent of **n** ☺
Memory still fragmented ☹

Lets Play with Malloc()



Questions?