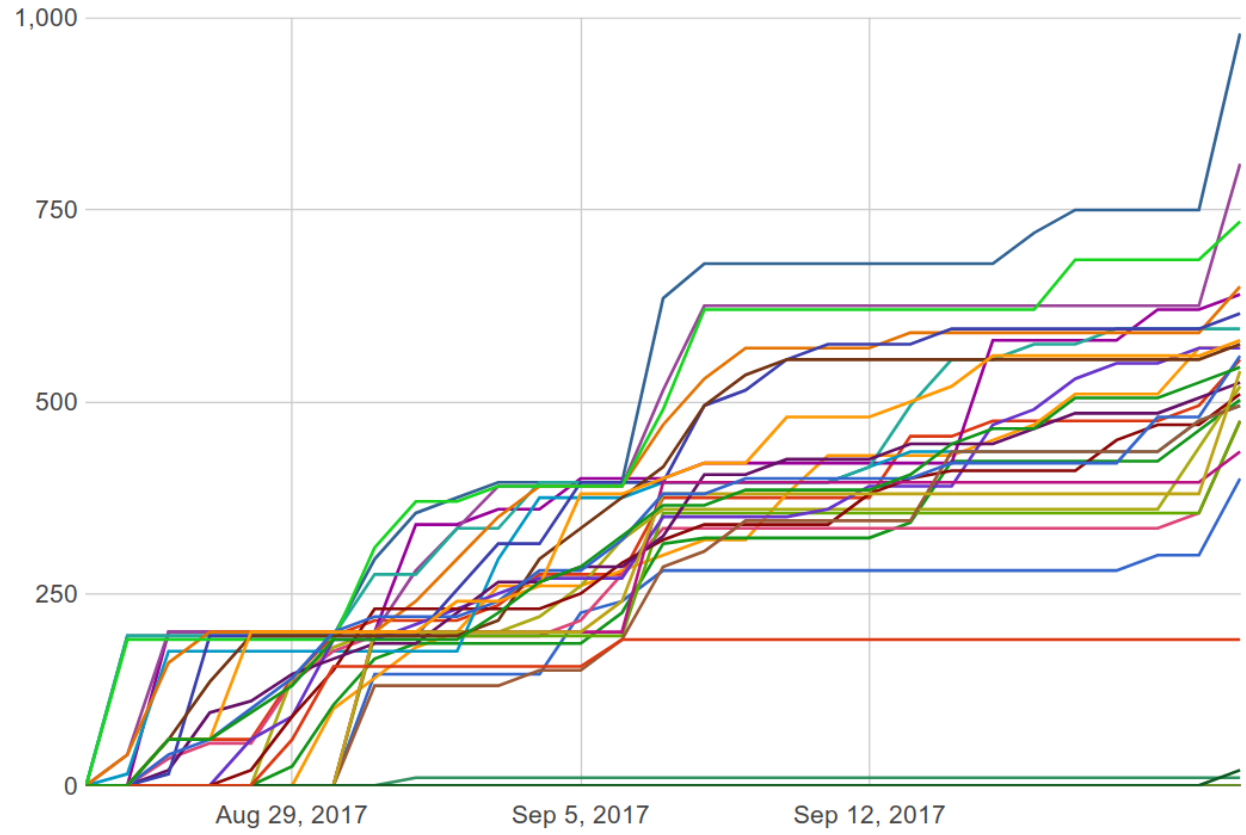
Lec05: Stack Protections

Taesoo Kim

Scoreboard



Administrivia

- In total, 10+ labs + NSA Challenge (we already completed four!)
- Due: Lab05 is out and its due on Sept 28 at midnight
- NSA Codebreaker Challenge started! (Due: Nov 30)

Best Write-ups for Lab03

- simple-bof : carterchen, markwis
- jmp-to-stack : shudak3, nagendra
- jmp-to-env : carterchen, brian_edmonds
- frobnicated : rpgiri, vseshadri30
- argc0 : rohandvora, shudak3
- lack-of-four : carterchen, shudak3
- jmp-to-where : dhaval, brian_edmonds
- unusual-main : shudak3, spark720
- man-strncpy : dhaval, poning
- upside-down : carterchen, dhaval

Course Grading (Expectation for A/B)

1. Absolute scoring/grading
2. Expectation:
   - 7.5 on average → A
   - 5.5 on average → B
   - If it's still too high for your goal, please talk to me in person!
3. NSA Challenge:
   - Task0/1/2/3: 20, Task4/5/6: 60
   - Bonus If you solve Task6, we will bump up your grade!

NSA Codebreaker Challenges

University	Task 0	Task 1	Task 2	Task 3	Task 4	Task 5	Task 6
Georgia Institute of Technology	20	15	12	5	2	1	0
Pennsylvania State University	14	8	8	1	1	1	0
Carnegie Mellon University	7	4	4	1	1	1	0
Lafayette College	1	1	1	1	1	1	0
University of Hawaii	14	8	6	2	2	0	0
Johns Hopkins University	7	4	4	1	1	0	0
Virginia Community College System	3	1	1	1	1	0	0
Lesley University	1	1	1	1	1	0	0
Ohio State University, Columbus	1	1	1	1	1	0	0
University of Tulsa	9	5	5	5	0	0	0

Showing 1 to 10 of 315 entries

Previous 1 2 3 4 5 ... 32 Next

Lab03: Stack Overflow

Name	Points	Release	Deadline	Solved
simple-bof	20	09-08-2017 00:00:00	09-22-2017 00:00:00	25
jmp-to-stack	20	09-08-2017 00:00:00	09-22-2017 00:00:00	25
jmp-to-env	20	09-08-2017 00:00:00	09-22-2017 00:00:00	24
frobnicated	20	09-08-2017 00:00:00	09-22-2017 00:00:00	24
argco	20	09-08-2017 00:00:00	09-22-2017 00:00:00	24
lack-of-four	20	09-08-2017 00:00:00	09-22-2017 00:00:00	25
jmp-to-where	20	09-08-2017 00:00:00	09-22-2017 00:00:00	17
unusal-main	20	09-08-2017 00:00:00	09-22-2017 00:00:00	21
man-strncpy	20	09-08-2017 00:00:00	09-22-2017 00:00:00	18
upside-down	20	09-08-2017 00:00:00	09-22-2017 00:00:00	11

Discussion: Lab03

- What's the most "annoying" bug or challenge?
- What's the most "interesting" bug or challenge?
- What did you learn in general?

Discussion: Not Yet Motivated?

Discussion: Not Yet Motivated?

Discussion: jmp-to-where

Discussion: jmp-to-where

Discussion: jmp-to-where

- What's your lessons?
- How to prevent this?

Discussion: unusal-main

Discussion: man-strncpy

```
char *
strncpy(char *dest, const char *src, size_t n) {
  size_t i;
  for (i = 0; i < n && src[i] != '\0'; i++)
    dest[i] = src[i];
  for ( ; i < n; i++)
    dest[i] = '\0';
  return dest;
}
```

Discussion: man-strncpy

- What's your lessons?
- How to prevent this?

Discussion: man-strncpy (safe usage)

```
char buf[BUFSIZ];
strncpy(buf, input, sizeof(buf) - 1);
buf[sizeof(buf) - 1] = '\0';
```

Discussion: man-strncpy (alternative: strlcpy())

```
strlcpy(buf, s, sizeof(buf));
```

Discussion: upside-down

Discussion: upside-down

- More secure? less? in terms of security?
- What if we are not using stack at all? (e.g., stackless python)

Discussion: How to Prevent Stack Overflow?

- Two approaches:
  - Bug prevention
  - Exploitation mitigation
- Protect "integrity" of ra, funcptr, etc (code pointers)
  - (e.g., exploitation mitigation → NX, canary)
- Prevent the buffer overflow at the first place
  - (e.g., code analysis, better APIs)

Discussion: How to Prevent Stack Overflow?

- Exploitation mitigation (today's topic)
- Bug prevention

Today's Tutorial

- In-class tutorial
  - Let's understand the implementation of the stack protector.
  - Let's exploit the (insecurely) protected crackme0x00 to get a flag!

Reminder: crackme0x00

```
$ objdump -d crackme0x00
...
8048448:       8d 45 e8                lea    -0x18(%ebp),%eax
804844b:       89 44 24 04             mov    %eax,0x4(%esp)
804844f:       c7 04 24 8c 85 04 08    movl   $0x804858c,(%esp)
8048456:       e8 d5 fe ff ff          call   8048330 <scanf@plt>
...
```

```
           |<-- 0x18-->|+--- ebp
top                    v
[          [~~~~>  ]   ][fp][ra]
|<----   0x28  ------->|
```

Reminder: Exploiting crackme0x00

```
            |<-- 0x18-->|+--- ebp
top                       v
[           [~~~~>  ]   ][fp][ra]
|<----   0x28  ------->|
            AAAABBBB.....GGGGHHHH
```

crackme0x00 in C

```c
int main(int argc, char *argv[])
{
  char buf[16];
  printf("IOLI Crackme Level 0x00\n");
  printf("Password:");

  scanf("%s", buf);

  if (!strcmp(buf, "250382"))
    printf("Password OK :)\n");
  else
    printf("Invalid Password!\n");
  return 0;
}
```

By the way, how to fix crackme0x00's bug?

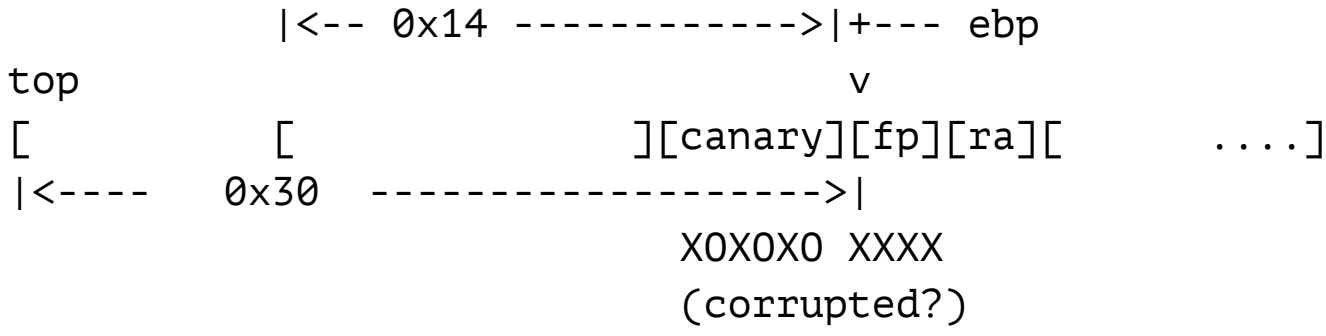
```
scanf("%15s", buf);  // NOTE. 15 not 16
or
scanf("%as", &buf);  // NOTE. char *buf, require a manual free()
```

DEMO: GCC's Stack Protector

- makefile
- compilation options
- diff.sh

Core Idea of Stack Protector

- Use a "canary" value as an indicator of the integrity of fp/ra



Why is it called "Canary"?

Why is it called "Canary"?

Subtle Design Choices for the Stack Canary

- Where to put? (e.g., right above ra? fp? local vars?)
- Which value should I use? (e.g., secrete? random? per exec? per func?)
- How to check its integrity? (e.g., xor? cmp?)
- What to do after you find corrupted? (e.g., crash? report?)

Lab05: Exploiting Weakness of Canary

Name	Points	Release	Deadline	Solved
xor	20	09-22-2017 00:00:00	09-29-2017 00:00:00	5
stackshield	20	09-22-2017 00:00:00	09-29-2017 00:00:00	3
weak-random	20	09-22-2017 00:00:00	09-29-2017 00:00:00	5
gs-random	20	09-22-2017 00:00:00	09-29-2017 00:00:00	3
terminator	20	09-22-2017 00:00:00	09-29-2017 00:00:00	2
assassination	20	09-22-2017 00:00:00	09-29-2017 00:00:00	2
mini-heartbleed	20	09-22-2017 00:00:00	09-29-2017 00:00:00	3
pltgot	20	09-22-2017 00:00:00	09-29-2017 00:00:00	2
ssp	20	09-22-2017 00:00:00	09-29-2017 00:00:00	2
fd	20	09-22-2017 00:00:00	09-29-2017 00:00:00	2

In-class Tutorial

- Step 1: Understanding GCC's Stack Protector
- Step 2: Let's exploit 0xdeadbeef canary!

```
$ ssh YOURID@cyclonus.gtisc.gatech.edu -p 2023
$ ssh YOURID@cyclonus.gtisc.gatech.edu -p 2022
$ ssh YOURID@computron.gtisc.gatech.edu -p 2023
$ ssh YOURID@computron.gtisc.gatech.edu -p 2022

$ cd tut/lab05
$ cat README
```

References

- Bypassing StackShield