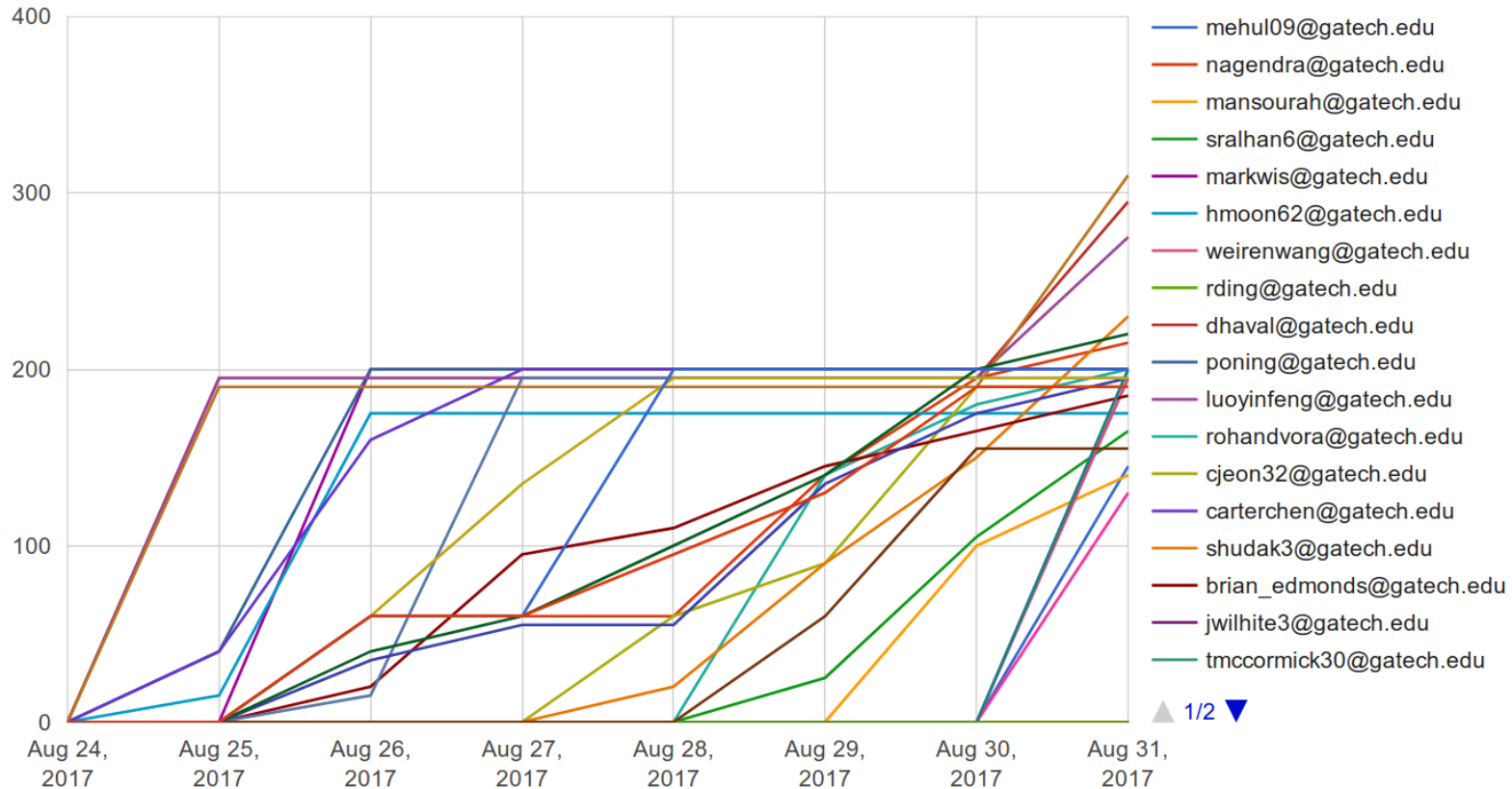


Lec02: x86_64 / Shellcode / Tools

Taesoo Kim

Scoreboard



Administrivia

- Survey: how many hours did you spend? (<3h, 6h, 10h, >20h)
- Join [Piazza](#)
- An optional recitation at 5-6pm on every Wed (in Klaus 1447)
- Lab02 is out!
- **Due** : Sept 8th at midnight

Course Grading

- 100% Lab (if you didn't turn in a single lab, you will get F)
- No midterm and final exams
- 40%: A , 30-40%: B , 30-20%: C and below
- If you solve 7.5 on average (correction!) from all labs, you will get A !
- We will pick two best write-ups per challenge
- See [Game Rules](#)

Best Write-ups for Lab01

- bomblab1_01: carterchen, brian_edmonds
- bomblab1_02: markwis, shudak3
- bomblab1_03: mansourah, brian_edmonds
- bomblab1_04: poning, prengasamy6
- bomblab1_05: spark720, meenal
- bomblab1_06: shudak3, nsehatbaksh3
- bomblab1_07: spark720, luoyinfeng
- bomblab1_08: meenal, whuang328
- bomblab1_09: shudak3, markwis
- bomblab1_10: shudak3, hwang312

Bomb Stats

- Bombs exploded ?? times in total?
- in ?? phases?
- ?? people exploded at least once?

Bomb Stats

- Bombs exploded 76 times in total ($76 \times -5 = -380$ pts)
- in ALL phases!
- 19 people exploded at least once! (so how many alive?)
 - Each lab: 19/02/02/08/02/03/09/06/01/04 people
 - Each lab: 32/02/02/09/03/03/12/08/01/04 times

Discussion 0

1. How the bomb works?

Discussion 1

1. How did you prevent bombs from explosion?

Discussion 2

1. What made your bombs exploded?

Discussion 2

1. What was the most difficult/annoying phase?

Discussion 3

1. How did you find 'secret_phrase'?

Discussion 4

1. Any tricky assembly?

Discussion 5

1. Any useful trick to share with other students?

ASM showcases 1

```
1ac1: e8 ba ef ff ff      call    a80 <__x86.get_pc_thunk.bx>
1ac6: 81 c3 3a 35 00 00     add     ebx,0x353a
...
```

```
0000a80 <__x86.get_pc_thunk.bx>:
a80: 8b 1c 24              mov     ebx,DWORD PTR [esp]
a83: c3                   ret
```

ASM showcases 2

```
1525: 65 a1 14 00 00 00    mov     eax,gs:0x14
152b: 89 84 24 24 04 00 00  mov     DWORD PTR [esp+0x424],eax
1532: 31 c0                xor     eax,eax
...
15cb: 8b 84 24 1c 04 00 00  mov     eax,DWORD PTR [esp+0x41c]
15d2: 65 33 05 14 00 00 00  xor     eax,DWORD PTR gs:0x14
15d9: 75 1c                jne    15f7 <print_key+0xfa>
...
5f7: e8 14 0c 00 00      call   2210 <__stack_chk_fail_local>
```


ASM showcases 3

```
2144: 8d a4 24 d4 e7 ff ff    lea    esp,[esp-0x182c]
214b: 83 0c 24 00             or     DWORD PTR [esp],0x0
214f: 8d a4 24 0c 10 00 00    lea    esp,[esp+0x100c]
```

ASMs that you read in Lab1

- function calls
- switch: jump table
- for/while loops
- recursion
- data structure: array/list/tree
- etc

Lab02: Bomb Lab2 / Shellcode

- Another Bomblab (be extra careful this time)!
- Writing five different shellcodes
 - x86, x86_64, both!, ascii, minimal size (competition)
 - **Bonus** : the smallest shellcode gets extra **10 pts!**

Today's Tutorial

- x86 shellcode overview
- In-class tutorial
 - PEDA (a fancy gdb plugin)
 - Walk over x86 shellcode (+ exercise!) and various tools

DEMO: PEDA commands

- (python: gdb.execute())
- context
- telescope/xinfo
- checksec/aslr
- vmmap/find
- elfheader, elfsymbol, hexdump
- pdisass, nearpc
- deactive
- rop: asmsearch, dumprop, ropgadget ...

shellcode (in C)

```
#include <stdio.h>
#include <unistd.h>

int main() {
    char *sh = "/bin/sh";
    char *argv[] = {sh, NULL};
    char *envp[] = {NULL};
    execve(sh, argv, envp);
    return 0;
}
```

DEMO: shellcode.S

- explain: asm, structure
- `man syscall` (about convention)
- `execve()`
- tutorial: `/bin/sh` to `/bin/cat /proc/flag`

In-class Tutorial

- Step 1: Install PEDA
- Step 2: Play with shellcode!

```
$ 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/lab02
$ cat README
```


References

- [Assembly](#)
- [x86](#)
- [x86_64](#)
- [PEDA](#)