CS161–Spring 2015 — Solutions to Homework 1

Quoc Thai Nguyen Truong, SID 24547327, `cs161-di`

February 26, 2015

Collaborators: God

Problem 1

(a) False
In order to keep u secure, you have to keep your code and system secret.

(b) False
format string bugs can be xploited for control-hijacking attacks by hackers.

(c) False
I'm not sure about this. So I prayed, and "False" is my final answer :)

(d) False
we can't prevents every attack, but not all of them.

(e) False
calling system() is a very bad way to run because it runs the default shell and passes your program as arguments.

(f) False
you can't guarantee a perfect program without a bug.

Problem 2

(a) $\boxed{\textit{Buggy}}$ at line 5.
At line 5, each memmove copies string of size(p) to p+2, so it can overflow 1 byte. Therefore, the number of bytes overflow is the number of "\n" (newline character). Hence, attacker can inject shell-code into the buffer overflow and overwrite the return address, which can jump and execute the shell-code.

(b) $\boxed{\textit{Not Buggy}}$

(c) $\boxed{\textit{Buggy}}$ at line 12.

This is buffer overrun. If track = curcd→numtracks, this can write past curcd→tracklen array. If track = curcd→numtracks, then I can make newtracklen be the address of same shell-code that lives somewhere in the memory.Therefore, we can see that this will overwrite curcd→notify to some pointer of a shell-code.

Problem 3

(a) $s1\ != NULL\ \&\&\ s2\ != NULL\ \&\&\ n1 <= size(s1)\ \&\&\ n_2\ <= size(s2)$

(b) $s1\ != NULL\ \&\&\ s\ != NULL\ \&\&\ i < size(s1)\ \&\&\ i\ < size(s)\ \&\&\ i >= 0$

(c) $s2\ != NULL\ \&\&\ s\ != NULL\ \&\&\ j < size(s2)\ \&\&\ i + j\ < size(s)\ \&\&\ i + j >= 0$

(d) $s\ != NULL\ \&\&\ i + j\ < size(s)\ \&\&\ i + j >= 0$

Problem 4

(a) Since it' $unit64\_t$, we know that each $memcpy()$ will write 8 Bytes.
So if $i = 18$, it will past end of a.

$+\ i = 0 \rightarrow a[0 \cdots 7]$
$+\ i = 1 \rightarrow a[1 \cdots 8]$
$+\ i = 2 \rightarrow a[2 \cdots 9]$
$\cdots$
$+\ i = 18 \rightarrow a[18 \cdots 25]$

(b) $\boxed{Any\ Value\ that\ is\ greater\ or\ equal\ to\ 19}$

Problem 5

(a)

$$\boxed{(2^{64} - 6) \times \left(\frac{1}{7}\right) + 1}$$

(b) P = $(x_0 >= 0) \wedge (y_0 >= 0) \wedge (x_0 == 8) \wedge (y_0 <= 19)$

(c) Q = $(y_0 >= 0) \wedge (y_0 <= 13)$

(d) $(P \wedge \neg Q) = [(x_0 >= 0) \wedge (y_0 >= 0) \wedge (x_0 == 8) \wedge (y_0 <= 19)] \wedge [(y_0 < 0) \vee (y_0 > 13)]$
$\boxed{(x_0 == 8) \wedge (14 <= y_0 <= 19)}$

Problem 6

(a) I would allow a certain (variable) number of miss-types, so that allow the users to access the system if their number of mis-types is less than or equals to the variable.

(b) If we allow a certain number of mis-types, the authorized users will have more more chances to log in. Hence, "False negative" will be decrease because because the authorized knows their passwords.
Also, "False positive" will be increase because we allow mis-types in the passwords, so the unauthorized users will have more chances of hacking and logging in.

(c) Let x be the # of errors that we will allow

$$P(x \leq k) \geq 0.998$$

$$\sum_{i=0}^{k} P(x = i)$$

$$\sum_{i=0}^{k} \binom{10}{i} \cdot (98)^{10-i} \cdot (0.02)^2$$

If the users make 2 or less typos, let them log-in to the system
"False negative" rate < 0.002
"False positive" rate $= \frac{1}{2^{30}}$

Problem 7

(a) 1: *xorl %ecx %ecx*
2: *addl* $42 *%ecx*
3: *movl %ecx %eax*

(b) D: 2ef4f00d (xchg %eax %ecx)
C: 2ef4cafe (addl $42 %ecx)
B: 2ef4face (xorl %ecx %ecx)
A: Doesn't matter

(c) E: 0xbfdecaf0 (./a.out)
D: 0xbfdecae0 (gcc /tmp/foo.c)
C: 0x1ffa4b28 (System)
B: 0x1ffa4b28 (System)
A: doesn't matter

(d) H: 0xbfdecaf8 (rm a.out/tmp/foo.c)
G: 0xbfdecaf0 (./a.out)
F: 0x1ffa4b28 (System)
E: 0x1ffa4b28 (System)
D: 0xbfdecae0 (gcc /tmp/foo.c)
C: 0x2ef4dead (pop)
B: 0x1ffa4b28 (System)
A: doesn't matter