**CSE 484 In-class Worksheet #5**

Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ UWNetID: \_\_\_\_\_\_\_\_\_\_\_\_\_ Date: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Email address: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Student #: \_\_\_\_\_\_\_\_\_\_\_\_\_

Partner names for this activity: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Q1:** (Alone) Jot down 3 defenses you could deploy against pointer overflow.

**Q2**: (With your neighbor) Oftentimes defenses come at significant costs, or their design must be carefully considered to ensure they don’t break anything that users of the defended system need to do. What are some possible challenges with implementing PointGuard?

**Q3:** The goal of this code is to allow a program to open regular files, but not symlinks.

int openfile(char \*path) {

struct stat s;

if (stat(path, &s) < 0)

return -1;

if (!S\_ISRREG(s.st\_mode)) {

error("only allowed to regular files!");

return -1;

}

return open(path, O\_RDONLY);

}

Can you spot any potential problems?

**Q4:** Consider this code:

char buf[80];

void vulnerable() {

int len = read\_int\_from\_network();

char \*p = read\_string\_from\_network();

if (len > sizeof buf) {

error("length too large, nice try!");

return;

}

memcpy(buf, p, len);

}

And note the following definitions:

void \*memcpy(void \*dst, const void \* src, size\_t n);

typedef unsigned int size\_t;

Can you spot any potential problems?

**Q5:**  Consider this code:

size\_t len = read\_int\_from\_network();

char \*buf;

buf = malloc(len+5);

read(fd, buf, len);

Can you spot any potential problems?

**Q6:** What issues, if any, do you see with the following code for password comparisons?

// The following is the functional description of the code -- what it should do

PwdCheck(RealPwd, CandidatePwd) should:

Return TRUE if RealPwd matches CandidatePwd

Return FALSE otherwise

RealPwd and CandidatePwd are both 8 characters long

// The following is the implementation, like on the TENEX system

PwdCheck(RealPwd, CandidatePwd) // both 8 chars

for i = 1 to 8 do

if (RealPwd[i] != CandidatePwd[i]) then

return FALSE

return TRUE