Department of Electrical Engineering and Computer Science

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

6.858 Spring 2018

Quiz I

You have 80 minutes to answer the questions in this quiz. In order to receive credit you must answer the question as precisely as possible.

Some questions are harder than others, and some questions earn more points than others. You may want to skim them all through first, and attack them in the order that allows you to make the most progress.

If you find a question ambiguous, be sure to write down any assumptions you make. Be neat and legible. If we can't understand your answer, we can't give you credit!

Write your name and submission website email address on this cover sheet.

This is an open book, open notes, open laptop exam.
NO INTERNET ACCESS OR OTHER COMMUNICATION.

This quiz is printed double-sided.

Please do not write in the boxes below.

I (xx/13)	II (xx/20)	III (xx/12)	IV (xx/12)	V (xx/14)	VI (xx/12)	VII (xx/13)	VIII (xx/4)	Total (xx/100)

Name:

Submission website email address:

You can answer the feedback questions on the back of the quiz before the official start time.

This page intentionally left blank.

I Paper reading questions

1. [5 points]:

Which of the following attack scenarios are addressed by some aspect of Google's security architecture as described in their whitepaper?

A. True / False Google engineers inserting backdoors into software.

B. True / False A CPU manufacturer such as Intel inserting a backdoor into their CPU.

C. True / False A user of GMail inadvertently running malware on their computer that steals their email (and sends it over TCP to an attacker's server).

D. True / False A user of GMail has a sneaky roommate who watches the user type in his password.

E. True / False An adversary gets a job as a data center operator and steals a server's hard drives in order to obtain user data.

2. [4 points]: Which of the following statements are true about U2F (as described in the assigned reading "Universal 2nd Factor (U2F) Overview")?

(Circle True or False for each choice.)

A. True / False U2F is a stronger second factor than sending an SMS code to a user's smartphone.

B. True / False An attacker that knows a user's password can easily guess the U2F key to access the user's account.

C. True / False A U2F USB dongle prevents malware on the user's computer from stealing the user's second factor to authenticate as that user even when the user's computer is turned off.

D. True / False A server using U2F can reliably determine that the user who is attempting to login is indeed behind the computer that sent the login request.

This page intentionally left blank.

3. [4 points]: Which of the following recommendations should one follow according to Paul Youn? **(Circle True or False for each choice.)**

A. True / False Sprinkle two-factor authentication everywhere.

B. True / False When designing a defense for a system like AirBnB, one should focus on particular attacks.

C. True / False One should think of computer security as an engineering discipline.

D. True / False One should learn how to penetrate computer systems to learn how to think like an adversary.

This page intentionally left blank.

II Buffer overflows

Consider the following code snippet from lab 1's `http.c`:

```
1  void http_serve(int fd, const char *name)
2  {
3      void (*handler)(int, const char *) = http_serve_none;
4      char pn[1024];
5      struct stat st;
6
7      getcwd(pn, sizeof(pn));
8      setenv("DOCUMENT_ROOT", pn, 1);
9      strcat(pn, name);
10     split_path(pn);
11
12     if (!stat(pn, &st))
13     {
14         /* executable bits -- run as CGI script */
15         if (valid_cgi_script(&st))
16             handler = http_serve_executable;
17         else if (S_ISDIR(st.st_mode))
18             handler = http_serve_directory;
19         else
20             handler = http_serve_file;
21     }
22
23     handler(fd, pn);
24 }
```

4. [10 points]: Point out two ways to exploit this code fragment. One of them should require running code on the stack, and the other should work even if the operating system marked the stack as non-executable (NX). For each exploit, describe it briefly, referring to specific lines of code above as needed.

This page intentionally left blank.

The stack layout of `http_serve` is as follows:

```
0xbffff59c: return address
0xbffff598: saved ebp
0xbffff58c: variable handler
0xbffff18c: variable pn
0xbffff134: variable st
```

5. [10 points]: For `zookd-nxstack`, what input should the adversary provide to `http_serve` to launch a return-to-libc attack to delete `/home/httpd/grades.txt`? (Assume the address of the `unlink` libc function is 0xBCBCBCBC.) Write down the sequence of bytes that the adversary should provide in the `name` argument.

This page intentionally left blank.

III Baggy bounds checking

Assume you compile the zookws sources with Baggy Bounds as described in the paper "Baggy Bounds Checking: An Efficient and Backwards-Compatible Defense against Out-of-Bounds Errors" by Akritidis et al. Refer to the code from the previous question about Buffer Overflows. Assume that both zookws and all system libraries are compiled with Baggy Bounds checking.

6. [6 points]: Will baggy bounds prevent `strcat` from writing past the end of `pn`? If so, briefly explain why? If not, briefly explain why not?

7. [6 points]: If the bound on `pn` is 1028 instead of 1024 will baggy bounds prevent `strcat` from writing past the end of `pn`? If so, briefly explain why? If not, briefly explain why not?

This page intentionally left blank.

IV Capsicum

Ben Bitdiddle observes that FreeBSD already uses integers to represent processes (i.e., FreeBSD process IDs are 32-bit integer values), and decides to simplify Capsicum's design by keeping the original FreeBSD process IDs instead of Capsicum's process descriptors.

8. [12 points]: What problem might arise in Ben's design that uses PIDs instead of Capsicum's design that uses process descriptors?

This page intentionally left blank.

V Native Client

Consider the validator for Native Client as shown in Figure 3 of the NaCL paper ("Native Client: A Sandbox for Portable, Untrusted x86 Native Code").

9. [14 points]: Note that if `inst_is_indirect_jump_or_call(IP)` is true, then IP is not added to `JumpTargets`. Ben's wonders why not adding is important. He modifies the code to add IP to `JumpTargets` in both branches of that `if` statement. Show a fragment of code for a module that an adversary could write, that would pass Ben's modified validator, and that would allow to the code to jump to an arbitrary (possibly not-32-byte-aligned) address `a`. (Briefly explain your answer.)

This page intentionally left blank.

VI iOS

10. [12 points]: Alyssa P. Hacker develops iOS applications, and wants to be able to upgrade and downgrade the iOS software on her iPhone, to test her applications with different iOS versions. Assume that she can read and write her iPhone's OS image from her computer via USB. How can she defeat Apple's downgrade protection?

This page intentionally left blank.

VII Android

Consider the Android system as described in the paper by Enck et al.

Android applications are distributed as `.apk` files, which are just ZIP files. When an application is installed, the `.apk` file is saved in the phone's file system. The installer temporarily extracts the contents of the ZIP file (the Java code, the manifest, and the signature), checks the signature on the Java code and manifest, and asks the user to approve the permissions from the manifest. At boot time, the Android system loads each `.apk` file by extracting the manifest and Java code.

Ben Bitdiddle discovers that the Android installer and the boot-time Android system differ in how they interpret ZIP files. If the ZIP file has two files with the same file name, the Android installer takes the first one, and the boot-time code takes the last one.

11. [13 points]: What aspects of Android's security model can be broken using this bug? Describe a specific attack that Ben can perform.

VIII 6.858

We’d like to hear your opinions about 6.858. Any answer, except no answer, will receive full credit.

12. [2 points]: Are there things you’d like to see improved in the second half of the semester?

13. [2 points]: Is there one paper out of the ones we have covered so far in 6.858 that you think we should definitely remove next year? If not, feel free to say that.

End of Quiz