Department of Electrical Engineering and Computer Science

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

6.566 Spring 2024

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 Gradescope email address on this cover sheet.

This is an open book, open notes, open laptop exam.
Internet access limited to public static content. No communication.

This quiz is printed double-sided.

Please do not write in the boxes below.

I (xx/18)	II (xx/12)	III (xx/12)	IV (xx/10)	V (xx/8)	VI (xx/8)	VII (xx/6)	VIII (xx/6)	Total (xx/80)

Name:

Gradescope 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 OS/VM isolation

An adversary submits a patch to the Linux kernel that introduces a back door in the `ioctl()` system call, where if the system call is invoked with specific arguments, the kernel will change the calling process UID to 0 (root). Suppose this patch gets accepted by Linux and widely deployed. Assume there are no other implementation bugs that an adversary can exploit.

1. [6 points]: Can an adversary use this backdoor to escape from isolation on a system using Firecracker? Explain how (and under what assumptions), or why not.

2. [6 points]: Can an adversary use this backdoor to escape from isolation on a system using gVisor? Explain how (and under what assumptions), or why not.

3. [6 points]: Can an adversary use this backdoor to escape from isolation on a system using LXC? Explain how (and under what assumptions), or why not.

This page intentionally left blank.

II WebAssembly

4. [12 points]: Suppose that you run a WebAssembly function in a module from an adversary where the function pushes a large number of values onto the stack (more than the compiler/runtime stack has space for). What is the earliest point at which the compiler/runtime can catch this problem, and how would it do so? Assume that the compiler/runtime has no bugs.

This page intentionally left blank.

III Spectre/Meltdown

Ben Bitdiddle works for a CPU manufacturer and is charged with preventing the original Meltdown attack, "Meltdown-PF," where the CPU speculates that a memory access to kernel memory will succeed, even if it will eventually be rolled back when the CPU discovers the access should not have been allowed. He decides to change the CPU as follows. When a speculatively-executed instruction turns out to have been mis-speculated, and is rolled back, the CPU also evicts any cache lines brought into the cache as part of that speculative operation.

5. [12 points]: Does Ben's design prevent an adversary from using the Meltdown-PF vulnerability to extract data that the adversary should not have access to? Explain why or why not.

This page intentionally left blank.

IV iOS security

6. [10 points]: A user wants to get data from their broken iPhone, and they have extracted the flash storage chips from it; the rest of the phone doesn't work. The user also manages to get their K_{PIN} (the passcode key from the iOS security paper). Explain how the user can decrypt the data in their flash storage chips, or explain precisely why it's not possible.

This page intentionally left blank.

V Baggy

Ben Bitdiddle decides to drop the requirement that out-of-bounds (OOB) pointers must be at most `slot_size/2` past the end of an object from Baggy, and instead allows OOB pointers to be an entire `slot_size` past the end of an object. Consider an application that starts with the following code:

```
char *p = malloc(256);
  // Assume p gets the value 0x1000
char *q = p + 256 + 12;
  // Now q gets the value 0x8000110c
...
```

7. [8 points]: Explain how an application could perform an out-of-bounds write under Ben's design, starting with the above example code. Clearly state any assumptions under which the out-of-bounds write would occur. Assume Ben's variant of Baggy has no implementation bugs (i.e., correctly implements Ben's design), and all of the application's code is compiled with Baggy.

This page intentionally left blank.

VI Formal verification

Ben Bitdiddle wants to implement a new cryptographic scheme, and outsources the job of writing the code to another developer, who happens to be malicious. Ben asks the developer to write their code in the F* system, and the developer gives Ben an F* file that contains code written in Low*. The file passes the F* type checker, and running Kremlin/Karamel on the file generates a seemingly working C implementation.

8. [8 points]: What should Ben check to make sure there aren't any bugs in the resulting C code? Assume the tools (F*, Kremlin/Karamel, etc) are correct and bug-free.

This page intentionally left blank.

VII Lab 2

9. [6 points]: An adversary wants to directly modify the database storing their Zoobar balance. What container would they need to break into, and what file (pathname) would they need to modify?

VIII 6.566

We'd like to hear your opinions about 6.566. Any answer, except no answer, will receive full credit.

10. [3 points]: Out of the papers that we have covered so far, listed below, circle the one that you think we should definitely remove next year (or mark none if you think all papers should stay).

- OS/VM isolation: Firecracker, gVisor, comparison study.
- WebAssembly: wasm design paper, rWasm/vWasm paper.
- BitLocker.
- Transient execution attacks.
- OpenSSH privilege separation.
- Google security architecture, BeyondProd whitepaper.
- iOS security.
- Web security readings.
- Baggy bounds checking.
- EXE symbolic execution.
- HACL* verification.
- Do not remove any papers.

11. [3 points]: What would you like to see improved in this class in the second half of the semester?

End of Quiz