CS 88: Week 5: Midterm Practice Questions

Identify the Security Principles that each of the following cases rely on. What is the implicit assumption each case relies on? State in one sentence if their approach provides a good line of defense against attacks:

A. Many home owners leave a house key under the floor mat in front of their door.

B. When a traffic light detects that it may be giving conflicting signals, it enters a state of error and displays a flashing red light in all directions.

C. Tesla vehicles come equipped with "Sentry Mode" which records footage of any break ins to the vehicle and alerts the vehicle owner of the incident.

Mark the following statements as T/F

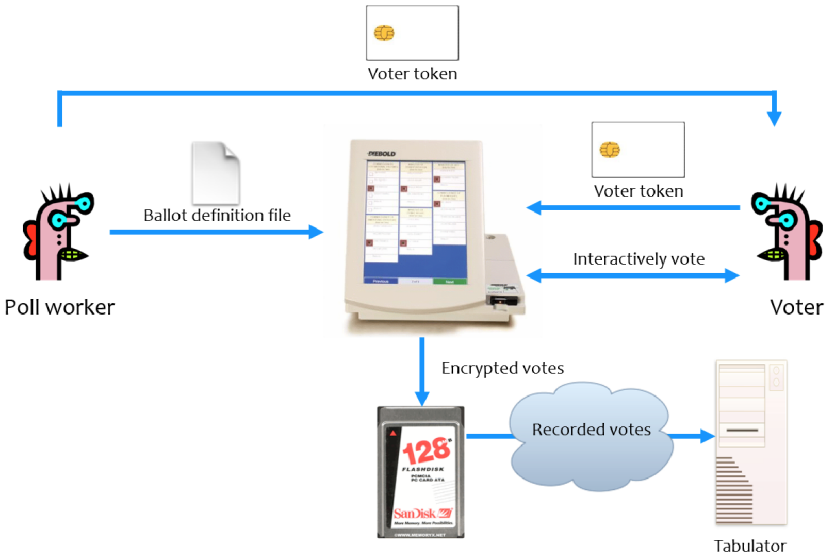
_____ Use fail-safe defaults: If security mechanisms fail or crash, they should default to the last executing behavior
_____ Defense in depth is about layering multiple defenses together
_____ ASLR, stack canaries, and NX bits all combined are sufficient to prevent all buffer overflow attacks.
_____ A format-string vulnerability can allow an attacker to overwrite values below the stack pointer.

Short Answer Questions

Q1. What vulnerability would arise if the stack canary was below the return address?

Q2. The GoogleFuzzer is a new fuzz testing suite that guarantees it will find at least one vulnerability in any C code that you give it (assuming there are vulnerabilities). Comment on whether such a claim is believable.

Security Mindset



Q1: What do you think are the security goals of the electronic voting system described in class and shown above? What assets must be protected?

Q2:

A. Who are the **adversaries** who might try to attack this electronic voting system?

B. What might be the **attacker's goals**? What potential **threats or vulnerabilities** do you see?

C. What are three defense approaches to security threats?

Memory Access

In which memory segments are the variables in the following code located?

```
int i = 0;
void func(char *str) {
      char *ptr = malloc(sizeof(int));
      char buf[1024];
       int j;
       static int y;
}
```

In which memory segments are the variables in the following code located? Specify whether they are located in the code, stack, heap, or data sections.

	Region of memory
void func(char * str)	
char * ptr	
data pointed to by char* ptr	
int i	
int j	
static int y	
char buf[1024]	

Buffer Overflow

The following function is called in a remote server program on a 32bit x86 architecture. The argument `str` points to a string that is entirely provided by users (the size of the string is up to 300 bytes).

The size of the buffer is X, which is unknown to us (we cannot debug the remote server program). However, somehow we know that the address of the buffer array is 0xAABBCC10, and the distance between the end of the buffer and the memory holding the function's return address is 8.

Although we do not know the exact value of X, we do know that its range is between 20 and 100. Please write down the string that you would feed into the program, so when this string is copied to the buffer and when the bof() function returns, the server program will run your code. You only have one chance, so you need to construct the string in a way such that you can succeed without knowing the exact value of X. In your answer, you don't need to write down the injected code, but the offsets of the key elements in your string need to be correct.

```
int bof(char *str) {
      char buffer[X];
      strcpy(buffer,str);
      return 1;
}
```

Buffer Overflow Defenses

Present two separate mechanisms (either at the compiler, hardware or implemented by the user) that can prevent the buffer overflow attacks in the above code. Are they fool-proof?

Pre-and Post-Conditions: Provide Pre- (conditions required upon entry) and Post (conditions ensured upon completion) conditions for the following code to ensure that it is memory safe

```
char buf[80];
void vulnerable(){
      int len = read_int_from_network();
      char *p = read_string_from_network();
      if(len > sizeof(buf)){
            perror("Length is too large!\n");
            return -1;
      }
      memcpy(buf, p, len);
      }

/* Note the following definitions */

void *memcpy(void *dst, const void *src, size_t n);
typedef unsigned int size_t;
```

SQL Injection

SQL Injection allows remote users to execute code on databases. In a typical setup, the database is only accessible to the web application server, not to remote users, so there is no direct path for users to interact with the database. How can users inject code to the database?

HTTP, the Web and Cookies

NYTimes decides to disable all third-party cookies on its website to prevent users from being tracked. This includes cookies from Google Analytics. Comment on whether you think this is a sufficient policy to prevent users from being tracked.