Software Security (201500032) Exam

Wed, 27 January

1

Given the following variation of the buffer overflow assignment, does it still allow a shellcode to be invoked when protections are disabled?

The difference with the lab assignment is that the badfile is only 40 bytes instead of 517. With ASLR enabled, is there more or less chance of starting a shellcode compared to the original code with a buffer of 517 bytes? Explain your answers.

Do not use the alternative ASLR-breaking solution that was found in the lab.

```
int bof(char* str)
{
  char buffer[24];
  strcpy(buffer, str);
  return 1;
}
int main(int argc, char**argv)
{
  char str[40];
  FILE *badfile;
  badfile = fopen("badfile", "r");
  fread(str, sizeof(char), 40, badfile);
  printf("Calling bof\n");
  bof(str);
  printf("Returned Properly\n");
  return 1;
}
```

2

Consider the following program fragments in the C and Java programming languages:

```
// C
for(int i = 0; i < n; i++) {
        char *x; ... x = a[i]; ... x ...
}

// Java
for(int i = 0; i < n; i++) {
        String x; ... x = a[i]; ... x ...
}

// Java
for(String x : a) { ... x ... }
```

Discuss the differences in memory and type safety that these languages / constructs provide. What errors can programmers make? What is the effect of such errors? What countermeasures can be taken

by programmers? How does the language help? Motivate your answers. Use and explain the relevant terminology.

3

A web application with a search feature shows the entered search query on the page that presents the results. When entering the following query:

```
<script>alert(document.cookie);</script>
```

An alert popup is triggered on the results page and shows a value. What type of exploit is this page vulnerable to (be as specific as you can be to describe the exploit)? How could an attacker abuse this and what could he achieve? How could this exploit be prevented?

4

What vulnerability can be identified in the following HTML fragment, which is displayed after a user logs in. How can the vulnerability be expoited? Describe a countermeasure to prevent the vulnerability.

```
<form action="account.php" method="post">
  new amount:
  <input name="newvalue">
  <input type="submit" value="save">
</form>
```

Tips:

- The form does something that requires you to log in, so it's something that is not available to just any user. For example, it will update admin settings.
- The login creates a session cookie in the browser. The account.php code on the server is not given so assume that the problem doesn't lie there.
- None of the form's HTML is coming from user data stored or sent to the server. Think of the common web vulnerabilities analyzed in the I2 lab.

5

You use a web application that refers to a database with the following tables.

- TABLE: users, with columns: (username, password, ssn)
- TABLE: products, with columns: (productcode, productname, producer, color, price)

Your web application checks the username and password using the following query, where you control both $username and $password:

```
SELECT username FROM users WHERE username = '$username' AND password = '$password'
```

Suppose both username and password are vulnerable (non-sanitized); craft an sql injection that allows you to enter in the system as user "sandro" without knowing the password of "sandro",

Followup question:

Besides logging in without a password, what else could an attacker achieve? Explain your answer.

6

What is the cause of SQL injection vulnerabilities in programs? Illustrate with an example. How should programmers defend against such vulnerabilities? Can these counter measures be enforced? That is, is it possible to guarantee the absence of SQL injection attacks? Sketch the design and implementation of a language that provides such guarantees. Can this design be generalized to other types of injection attacks?