A22652 *No calculator permitted in this examination*

UNIVERSITY OF BIRMINGHAM

School of Computer Science

MSc Advanced Computer Science
MSc Computer Science
MSc Computer Security
MSc Human-Computer Interaction
Fourth Year – MEng Computer Science/Software Engineering
First Year - UG Affiliated Science without Borders Computer Science
Fifth Year – MEng Computer Science/Software Engineering with Industrial Year

06 20010

Secure Programming

Summer Examinations 2015

Time allowed: 1 hour 30 minutes

[Answer ALL Questions]

No calculator

[Answer ALL Questions]

1. (a) A journalist wrote about a recent computer security failure: “This was not really a hack, as no data was stolen”. Explain what is wrong with this statement. In particular, is it helpful to think about secure programming as being only about “data loss”? [10%]

   (b) Discuss which of the Saltzer and Schroeder security principles are relevant to the defence mechanism of a non-executable stack in C (sometimes called W^X). [10%]

2. (a) Explain SQL command injection attacks using an example. [10%]

   (b) What is the standard technique for preventing SQL attacks in JDBC? Explain why it works in terms of compiling and parsing. [10%]

3. (a) Suppose there is a buffer overflow, but the attacker cannot inject any machine code into the system. Explain whether it is still possible for an attacker to use a buffer overflow in order to cause arbitrary malicious behaviour. [10%]

(b) Consider the following code:

```
char secret[]; // some secret data

char *q1;
char *q2;

// an untrusted user has read and write access to this
char *user_data;

void work_on_secret()
{
    strcpy(q1, secret);
    // work on secret data accessed via q1
    free(q1);
    q1 = NULL;
}

void work_for_user(int user_request)
{
    switch(user_request)
    {
        case 0:
            strcpy(q2, user_data);
            break;
        case 1:
            strcpy(user_data, q2);
            break;
        default:
    }
}

int main(int argc, char *argv[])
{
    *q1 = malloc(100);
    work_on_secret();
    // some more code here
    *q2 = malloc(100);
    work_for_user(1);
    // some more code
    work_for_user(0);
}
```

Explain what the two main security defects are in the code above. Does setting q1 to NULL in the first function above help to make the code more secure? [10%]

(c) Explain which defences covered in the module would work in this case, and which would not. You should consider both automatic defences by the compiler or operating system, and manual defences, such as rewriting the code to be more secure. [10%]

(d) Suppose an attacker tries to overwrite the return address with a null pointer. Discuss whether that is feasible or useful to the attacker. [10%]

4. (a) Discuss the relevance of regular expressions to secure programming. [10%]

(b) Consider the following regular expression:

```
([a-z0-9] | [0-9]+)* $
```

Suppose this expression is matched on a backtracking matcher, as in Java or Perl, against a user-malleable input. Explain what attack is possible in this case and give a suitable attack string. [10%]