UNIVERSITY OF EDINBURGH

COLLEGE OF SCIENCE AND ENGINEERING

SCHOOL OF INFORMATICS

INFR11098 SECURE PROGRAMMING

Thursday 15th May 2014

14:30 to 16:30

INSTRUCTIONS TO CANDIDATES

Answer any TWO questions.

All questions carry equal weight.

CALCULATORS MAY NOT BE USED IN THIS EXAMINATION

Year 4 Courses

Convener: I. Stark
External Examiners: A. Cohn, T. Field

THIS EXAMINATION WILL BE MARKED ANONYMOUSLY

1. Buffer overflow exploits often rely on a piece of *shellcode* to achieve an intermediate aim of the attacker.

   (a) Briefly describe the purpose of shellcode and, from the point of view of the attacker, three desirable properties it should have. [*4 marks*]

   (b) An attack attempt has been discovered on a Linux system which targets an overflow vulnerability in a system utility. After some work in GDB, the investigator produces the disassembly of some injected code shown below:

```
0x0804a040 <+0>:      xor    %eax,%eax
0x0804a042 <+2>:      xor    %ebx,%ebx
0x0804a044 <+4>:      xor    %ecx,%ecx
0x0804a046 <+6>:      push   %ebx
0x0804a047 <+7>:      push   $0x68732f6e
0x0804a04c <+12>:     push   $0x69622f2f
0x0804a051 <+17>:     mov    %esp,%ebx
0x0804a053 <+19>:     mov    $0x9fd,%cx
0x0804a057 <+23>:     mov    $0xf,%al
0x0804a059 <+25>:     int    $0x80
0x0804a05b <+27>:     mov    $0x1,%al
0x0804a05d <+29>:     int    $0x80
```

   With the help of Figures 1 and 2 on the next page, explain in detail what this shellcode does, how it must be executed, and how an attacker might be able to use it to his or her advantage. [*8 marks*]

   (c) What is the general term for this kind of attack? [*1 mark*]

   (d) In a classic 1983 paper, *A Distributed Secure System*, Rushby and Randell describe four different mechanisms for *separation* to isolate users from one another and from access control mechanisms. The mechanisms are:

   - **physical**
   - **temporal**
   - **logical**
   - **cryptographic**

   Consider exploiting buffer overflow vulnerabilities in a SMTP mail server. For each separation mechanism, explain how it is (or might be) used to help prevent attacks or reduce the damage caused. [*12 marks*]

QUESTION CONTINUES ON NEXT PAGE

0 NUL	10 DLE	20	30 0	40 @	50 P	60 ‘	70 p
1 SOH	11 DC1	21 !	31 1	41 A	51 Q	61 a	71 q
2 STX	12 DC2	22 "	32 2	42 B	52 R	62 b	72 r
3 ETX	13 DC3	23 #	33 3	43 C	53 S	63 c	73 s
4 EOT	14 DC4	24 $	34 4	44 D	54 T	64 d	74 t
5 ENQ	15 NAK	25 %	35 5	45 E	55 U	65 e	75 u
6 ACK	16 SYN	26 &	36 6	46 F	56 V	66 f	76 v
7 BEL	17 ETB	27 ’	37 7	47 G	57 W	67 g	77 w
8 BS	18 CAN	28 (	38 8	48 H	58 X	68 h	78 x
9 HT	19 EM	29)	39 9	49 I	59 Y	69 i	79 y
A LF	1A SUB	2A *	3A :	4A J	5A Z	6A j	7A z
B VT	1B ESC	2B +	3B ;	4B K	5B [	6B k	7B {
C FF	1C FS	2C ,	3C <	4C L	5C \	6C l	7C |
D CR	1D GS	2D -	3D =	4D M	5D]	6D m	7D }
E SO	1E RS	2E .	3E >	4E N	5E ^	6E n	7E ~
F SI	1F US	2F /	3F ?	4F O	5F _	6F o	7F DEL

Figure 1: ASCII Character Table

%eax	**Name**	**%ebx**	**%ecx**	**%edx**
1	sys_exit	int		
2	sys_fork	struct pt_regs		
3	sys_read	unsigned int	char *	size_t
4	sys_write	unsigned int	const char *	size_t
5	sys_open	const char *	int	int
6	sys_close	unsigned int		
7	sys_waitpid	pid_t	unsigned int *	int
8	sys_creat	const char *	int	
9	sys_link	const char *	const char *	
10	sys_unlink	const char *		
11	sys_execve	struct pt_regs		
12	sys_chdir	const char *		
13	sys_time	int *		
14	sys_mknod	const char *	int	dev_t
15	sys_chmod	const char *	mode_t	
16	sys_lchown	const char *	uid_t	gid_t

Figure 2: First 16 Linux system calls and parameter types

2. This question concerns some high profile vulnerabilities found in real-world code.

   (a) In early 2014, CVE-2014-1266 was released by US-CERT and NIST, with the description below:

   > *The SSLVerifySignedServerKeyExchange function in libsecurity_ssl/lib/sslKeyExchange.c in the Secure Transport feature in the Data Security component in Apple iOS 6.x before 6.1.6 and 7.x before 7.0.6, Apple TV 6.x before 6.0.2, and Apple OS X 10.9.x before 10.9.2 does not check the signature in a TLS Server Key Exchange message[...]*

   Apple issued a security update for these multiple versions of iOS and OS X. The updates consisted of large binary downloads, and required devices to be restarted after installation.

   i. Based on your understanding of TLS and X.509 certificate management, explain what can go wrong if this check fails and how an attacker can take advantage. [*4 marks*]

   ii. In Q4 2013, Apple sold 33.8 million iPhones and 4.6 million Macs. Suppose that, on average across different connection types and countries, the cost of downloading 10Gb of data is £1. Give and justify a very approximate "back-of-the-envelope" calculation for the cost of fixing this vulnerability, based on these figures and some of your own, including an estimate of the value of a person's time. What lesson from the Software Security Programming Lifecycle does this demonstrate? [*5 marks*]

   iii. On the next page is an excerpt from `sslKeyExchange.c` which shows the complete function `SSLVerifySignedServerKeyExchange` which is responsible for checking a signature on a session key sent from the server. Find the error in the code and briefly explain it. [*3 marks*]

   iv. How do you think this error might have risen? Give two alternative explanations. [*2 marks*]

   v. What urgent steps would you recommend that Apple deploy to avoid problems like this in future? [*1 mark*]

```
static OSStatus
SSLVerifySignedServerKeyExchange(SSLContext *ctx,
    bool isRsa, SSLBuffer signedParams,
    uint8_t *signature, UInt16 signatureLen)
{
  OSStatus        err;
  SSLBuffer       hashOut, hashCtx, clientRandom, serverRandom;
  uint8_t         hashes[SSL_SHA1_DIGEST_LEN + SSL_MD5_DIGEST_LEN];
  SSLBuffer       signedHashes;
  uint8_t         *dataToSign;
  size_t          dataToSignLen;

  signedHashes.data = 0;
  hashCtx.data = 0;

  clientRandom.data = ctx->clientRandom;
  clientRandom.length = SSL_CLIENT_SRVR_RAND_SIZE;
  serverRandom.data = ctx->serverRandom;
  serverRandom.length = SSL_CLIENT_SRVR_RAND_SIZE;


  if(isRsa) {
     /* skip this if signing with DSA */
     dataToSign = hashes;
     dataToSignLen = SSL_SHA1_DIGEST_LEN + SSL_MD5_DIGEST_LEN;
     hashOut.data = hashes;
     hashOut.length = SSL_MD5_DIGEST_LEN;

     if ((err = ReadyHash(&SSLHashMD5, &hashCtx)) != 0)
        goto fail;
     if ((err = SSLHashMD5.update(&hashCtx, &clientRandom)) != 0)
        goto fail;
     if ((err = SSLHashMD5.update(&hashCtx, &serverRandom)) != 0)
        goto fail;
     if ((err = SSLHashMD5.update(&hashCtx, &signedParams)) != 0)
        goto fail;
     if ((err = SSLHashMD5.final(&hashCtx, &hashOut)) != 0)
        goto fail;
  }
  else {
     /* DSA, ECDSA - just use the SHA1 hash */
     dataToSign = &hashes[SSL_MD5_DIGEST_LEN];
     dataToSignLen = SSL_SHA1_DIGEST_LEN;
  }

  hashOut.data = hashes + SSL_MD5_DIGEST_LEN;
  hashOut.length = SSL_SHA1_DIGEST_LEN;
  if ((err = SSLFreeBuffer(&hashCtx)) != 0)
      goto fail;

  if ((err = ReadyHash(&SSLHashSHA1, &hashCtx)) != 0)
```

```
      goto fail;
  if ((err = SSLHashSHA1.update(&hashCtx, &clientRandom)) != 0)
      goto fail;
  if ((err = SSLHashSHA1.update(&hashCtx, &serverRandom)) != 0)
      goto fail;
  if ((err = SSLHashSHA1.update(&hashCtx, &signedParams)) != 0)
      goto fail;
      goto fail;
  if ((err = SSLHashSHA1.final(&hashCtx, &hashOut)) != 0)
      goto fail;

  err = sslRawVerify(ctx,
                     ctx->peerPubKey,
                     dataToSign,      /* plaintext */
                     dataToSignLen,   /* plaintext length */
                     signature,
                     signatureLen);
  if(err) {
    sslErrorLog("SSLDecodeSignedServerKeyExchange:␣sslRawVerify␣"
        "returned␣%d\n", (int)err);
    goto fail;
  }

fail:
    SSLFreeBuffer(&signedHashes);
    SSLFreeBuffer(&hashCtx);
    return err;

}
```

QUESTION CONTINUES ON NEXT PAGE

(b) In 2008, Microsoft's Zune Media player stopped working. The cause was isolated to a piece of code in the clock driver, shown below. The idea of the driver is to calculate the current year based on the count of days stored in an integer in NVRAM.

```
year = ORIGINYEAR; /* = 1980 */

while (days > 365) {
    if (IsLeapYear(year))
    {
        if (days > 366)
        {
            days -= 366;
            year += 1;
        }
    }
    else
    {
        days -= 365;
        year += 1;
    }
}
```

i. Explain the programming error in this piece of code and describe the observable consequences. What is the general term for this kind of mistake? *[4 marks]*

ii. Supposing this programming error exists on a device that allowed downloadable applications to run, describe how it could be turned into an attack and possible motives an attacker might have. *[3 marks]*

iii. Give a fix for the problem. *[1 mark]*

iv. Do you believe this mistake could have been found automatically with a static analysis tool? Briefly justify your answer. *[2 marks]*

3. (a) In 2012, a group who affiliated themselves with the "Anonymous" hacktivist collective attacked web servers operated in the Top 100 universities in the world. University of Edinburgh was among those targeted. Dumped text files containing information obtained from running servers was posted on the public site `pastebin.com`.

   The next page contains a section of excerpts from one of the text files.

   i. Give two reasons why secure programming vulnerabilities on the web provide a good vehicle for groups like Anonymous. [*4 marks*]

   ii. Looking at the text file up to line 37, explain what you see and what observations you would make about the likely attack vector and the suspected vulnerability being exploited. [*5 marks*]

   iii. Looking at the password table on line 40, what three observations would you make concerning the site's password management? [*3 marks*]

   iv. The data in the file would likely be considered highly sensitive by its owners. Explain why, and what *threat agents* and *threats* might be posed by its discovery, beyond the publicity motivations of Anonymous. [*4 marks*]

   (b) Imagine that you are asked to conduct an exercise to make recommendations to development teams working on websites that are hosted by a university. There are two general categories of website, those hosting resources for *teaching* and *research*.

   For each of the following "touchpoints" in the Secure Software Development Lifecycle, briefly make three points that might be in your recommendations.

   - **abuse cases**: suggest three *specific* attacks and desirable responses;
   - **code review**: propose three review mechansisms;
   - **penetration testing, operations**: propose three specific activities. [*9 marks*]

QUESTION CONTINUES ON NEXT PAGE

```
http://www.roslin.ed.ac.uk/grants/grant.php?id=1012

available databases [4]:
[*] dev_roslin_institute
[*] roslin_institute
...

Database: roslin_institute‘
[137 tables]
+---------------------------------+
| JRF                             |
| ‘airg2012-abstracts‘            |
| ‘tse-rc-mouse-lines‘            |
| animalScienceForum              |
| animalTerritories               |
| base64_data                     |
| chickStrains                    |
| chickStrainsReferences          |
...
Database: dev_roslin_institute
Table: person
[11 columns]
+---------------+------------------------------------------------------+
| Column        | Type                                                 |
+---------------+------------------------------------------------------+
| email         | varchar(255)                                         |
| first_name    | varchar(255)                                         |
| interests     | text                                                 |
| last_name     | varchar(255)                                         |
| ninja_status  | enum('mortal','career track fellow','group leader') |
| password      | varchar(255)                                         |
| personid      | int(11)                                              |
| qualification | varchar(255)                                         |
| super         | tinyint(1) unsigned                                  |
| title         | enum('Prof','Dr','Mr','Mrs','Miss','Ms')             |
| visible       | tinyint(4)                                           |
+---------------+------------------------------------------------------+

Database: roslin_institute
Table: person
[110 entries]
+----------------------------------+
| password                         |
+----------------------------------+
| fca4d30918ee4b4c98225ad9661c0c27 |
| fca4d30918ee4b4c98225ad9661c0c27 |
| 5f7f266a1c9e7a58a978455fae58f897 |
| 957ad685428f4733268415ed1cb4a23c |
| fca4d30918ee4b4c98225ad9661c0c27 |
| 228c2bff0d2b0f3149a516abcacd6c6a |
...
```