OLLSCOIL NA hÉIREANN
THE NATIONAL UNIVERSITY OF IRELAND, CORK

COLÁISTE NA hOLLSCOILE, CORCAIGH
UNIVERSITY COLLEGE, CORK

AUTUMN EXAMINATION 2008

Fourth Year Computer Science

CS4253: Computer Security

Professor S. Craw,
Professor J. Bowen,
Dr. S.N. Foley

Answer *Four* questions
Questions carry equal marks

Three Hours

1. a) Explain the desirable properties of a *one way hash* function. Describe how such a function is used to protect passwords in the Unix system, and discuss the benefits of taking this approach. *(15 marks)*

   b) A vendor uses Message Authentication Codes (MACs) as signatures for orders from customers. A customer generating purchase order PO emails message $(PO, h_k(PO))$ to the vendor, where h_k is a keyed one-way-hash function and secret key k is known only to the vendor and the customer. To verify the signature the vendor simply recomputes the hash of the purchase order and compares it with the MAC provided. What is wrong with this signature scheme? *(15 marks)*

   c) A bank provides one-time password key-fobs to customers who wish to do their banking over the Internet. Each customer is given a unique key-fob which generates fresh time-based pass-codes at 30 second intervals. Each key-fob is tamper-resistant and stores a master secret key K (known only to the bank) and its owner's *userid*. A key-fob calculates the pass-code as $(\{time\}_K, \{userid\}_K)$. When a customer attempts to login, she provides $(userid, passcode)$; the remote bank system decrypts the pass-code fields, matches the use-rid and checks that the time is current.

   Outline an attack on this scheme that would allow an eavesdropper gain access to a another customer's account (without having to steal the victim's key-fob). *(15 marks)*

2. a) Describe the access-control mechanism that is used in Unix, paying particular attention to the permissions that are provided and their properties. *(15 marks)*

   b) Briefly describe the Biba model of Integrity and explain how it differs from the Clark Wilson model. *(15 marks)*

   c) A particular application system has users A and B, Transform Procedures (TPs) $T1$ and $T2$, and Constrained Data Items (CDIs) X, Y and Z. It has authorisation triples $(A, T1, (X))$ and $(B, T2, (Y, Z))$ which must be preserved according to the E2 rule of the Clark Wilson model.

      i. What application certification should be done given the above triples? *(3 marks)*

      ii. Describe how access control in standard Unix should be configured to support this policy. Note any potential security weaknesses in this implementation. *(7 marks)*

      iii. Suppose that a third user C may choose to always use either $T1$ *or* $T2$, but not both; once made, the choice cannot be reversed. Outline how this additional requirement could be supported *(5 marks)*

3. Alice (A) wishes to communicate securely with Bob (B) and proposes a symmetric session key K_{AB}, a copy of which she intends to give to Bob. Trent is a trusted third party who provides a message translation service. Trent shares symmetric K_{AT} with Alice, and symmetric key K_{BT} with Bob. The following protocol is used to pass the key K_{AB} to Bob.

$$\begin{aligned} \text{Msg1}: &\quad A \rightarrow T: \quad B, \{A, K_{AB}\}_{K_{AT}} \\ \text{Msg2}: &\quad T \rightarrow A: \quad \{A, K_{AB}\}_{K_{BT}} \\ \text{Msg3}: &\quad A \rightarrow B: \quad \{A, K_{AB}\}_{K_{BT}} \end{aligned}$$

a) What is the difference between long term and session keys? Describe how pass-phrase encryption might be used to provide long-term keys. What defences can be used to make it harder to carry out a dictionary attack on pass-phrases? *(15 marks)*

b) Describe how the above protocol might be used to secure services provided over a distributed system. Your answer should consider the issues of authentication, authorisation and revocation. *(15 marks)*

c) Illustrate how a third principle Eve (who shares a valid secret key K_{ET} with Trent) can subvert the protocol to get a copy of the key K_{AB} that Alice gives to Bob using this protocol. In addition, illustrate how Eve can subvert the protocol and masquerade as Alice to Bob, even when Alice does not initiate a key exchange with Bob. *(15 marks)*

4. a) Explain how a potential buffer overflow can result a Unix security vulnerability. Which of the following C programs have this vulnerability. Explain your answer. *(8 marks)*

```
void main1(int argc, char* argv[]){
  char buff[6];
  strcpy(buffer,argv[0]);
}/*main1*/
```

```
void main2(int argc, char* argv[]){
  char buff[6];
  strcpy(buffer,"long text");
}/*main2*/
```

b) Given suitable public generator g and modulus n, principals A an B generate suitable secrets x and y, respectively, and engage in the Diffie-Hellman Key exchange.

$$\begin{aligned} \text{Msg1}: &\quad A \rightarrow B \quad g^x \bmod n \\ \text{Msg2}: &\quad B \rightarrow A \quad g^y \bmod n \end{aligned}$$

i. How do A and B determine their shared key? *(5 marks)*

ii. Explain why this shared key cannot be determined by some third party. *(5 marks)*

iii. Modify the protocol so that it can be used to established a single shared key between three principals. *(5 marks)*

c) A programmer reads the S/KEY one-time-password scheme and (incorrectly) implements it as:

- For a given i, $h^i(s)$ denotes $h(h(\ldots h(s)))$, representing i applications of one-way hash function h to value s, where s represents an initial password (seed) chosen by the user.
- When a user picks her (initial) password s, the system stores $(1, h(s))$ in the password file.
- If a user has logged-in $i-1$ times since choosing initial password s, then the system stores $(i, h^i(s))$ in the password file.
- When a user attempts to log-in for the i^{th} time the system presents i as a challenge. The user (knowing s) provides response $r = h^{i-1}(s)$. The system compares $h(r)$ with the hash value stored in the password file and, if equal, updates this password entry to $(i+1, h(h(r)))$.

Outline an attack on this scheme and describe how it should be fixed. *(15 marks)*

5. a) Develop suitable Java security policy *grant* entries for the following requirements.

   i. Anybody may read and write files in `/tmp/`. *(5 marks)*

   ii. Any code signed by the public key `simon` may have read and write access to files under `/usr/home/simon/`. *(5 marks)*

   iii. *Any* jar files or classes from source `http://cs.ucc.ie` may have read access to any file in the directory `/usr/home/simon/cs`. *(5 marks)*

   b) The following Java fragment establishes an SSL connection to a server.

```
char[] storepass= "spasswd".toCharArray();
KeyStore keystore= KeyStore.getInstance("JCEKS");
keystore.load(new FileInputStream("keystore"),storepass);

TrustManagerFactory tmfactory= TrustManagerFactory.getInstance("SunX509");
tmfactory.init(keystore);
TrustManager[] clienttm= tmfactory.getTrustManagers();

SSLContext sslcontext= SSLContext.getInstance("SSL");
sslcontext.init(null, clienttm, null);

SocketFactory  sfactory = sslcontext.getSocketFactory();
/* open an SSL socket on host port 5999 */
SSLSocket s= (SSLSocket) sfactory.createSocket("serverhost",5999);
DataOutputStream out = new DataOutputStream(s.getOutputStream());
```

   Explain the purpose of the keystore, trust manager and socket factory in this code. Outline how they contribute to the authentication of, and the secure connection to, the server. *(15 marks)*

   c) The following protocol is used to authenticate a client C to a server S. Both principles share secret *pass*, R is a random challenge, and $h()$ is a one-way hash function.

$$\begin{aligned} \text{Msg1}: &\quad S \rightarrow C: \quad R \\ \text{Msg2}: &\quad C \rightarrow S: \quad h(R, pass) \end{aligned}$$

   The following Java code fragment from the server-side of this protocol reflects a number of (poor) implementation decisions. You may assume that the client-side uses similar implementation decisions.

```
MessageDigest md= MessageDigest.getInstance("MD5");
DataOutputStream out = ...   // stream to connecting client
DataInputStream in = ...   // stream from connecting client
byte[] passwd = ...                          // shared password

Random rangen = new Random(0);//java.util.Random generator-
byte[] R = new byte[1];              //-random seed used is 0
rangen.nextBytes(R);         // generate 1 byte random value
out.write(R);                                // send to client

byte[] hashR = new byte[16]; in.readFully(hashR);
byte[] hashpass = new byte[16]; in.readFully(hashpass);
if (MessageDigest.isEqual(hashR,md.digest(R))
        && MessageDigest.isEqual(hashpass,md.digest(pass)))
    ...                                      // client authenticated
```

   Identify and explain the security vulnerabilities in this implementation. Outline how the code should be repaired. *(15 marks)*