Computer and Network Security – Exercise no. 3

Submit in Pairs/Single to mailbox 19 by 11/1/12, 2:00 p.m.

1. In UNIX the password file contains for each user: <username:salt:password>.
   Note that password is not saved as a cleartext, but rather as a hashed value, e.g., as $h(salt, password)$ for a secure hash function $h(\cdot)$.
   For the reminder of the question, we shall assume that the adversary has access to the password file.

   (a) To improve the security of UNIX systems, it was suggested to change the system such that for each user there will be 16 passwords. All 16 passwords are stored with the same salt. In the ith trial to login to the system, the server will ask the i mod 16 password. If the password is correct, the login succeeds, otherwise, login with that username is blocked for 2 minutes.
   Under the assumption that all passwords are of the same strength (i.e., all passwords are chosen out of N possible strings randomly with equal probability), answer the following questions:
      i. Compared with the original system, does this system offer higher/lower/same level of security?
      ii. Does your answer change if for each password a different salt value is used?
      iii. Does your answer change if the blocking was removed?

   (b) The use of salt in password files addresses two security problems compared with storing just $h(password)$:
      i. It is easy to identify whether two users of the system have the same password.
      ii. When the number of users in the system grows, the expected number of password trials to find the password of one of the users drops (compared with the expected number of trials to find the password of a given user).

   For each of the ways to combine the password and the salt, explain for each of these two problems whether they are solved or not:
      i. $h(password) \oplus h(salt)$.
      ii. $h(password \oplus salt)$.
      iii. $h(password \oplus h(password \oplus salt))$

   Assume that salt is padded with zeros to obtain a string of the same length as the password (and recall that $\oplus$ stands for XOR).

2. Recall that in Kerberos version 5 the messages are:

1. Client → KDC: $ID_c, ID_{TGS}, nonce_1$
2. KDC → Client: $E_{k_c}(k_{c,TGS}, nonce_1), Ticket_{c,TGS}$
3. Client → TGS: $Auth_{c,TGS}, Ticket_{c,TGS}, ID_S, nonce_2$
4. TGS → Client: $E_{k_{c,TGS}}(k_{c,S}, nonce_2), Ticket_{c,S}$
5. Client → Server: $Auth_{c,S}, Ticket_{c,S}$
6. Server → Client: $Auth_{S,c}$

where $Ticket_{c,S} = ID_S, E_{k_S}(k_{c,S}, ID_c, \text{validity period})$ and $Auth_{c,S} = E_{k_{c,S}}(ID_c, \text{timestamp})$.

In "Insecurity For You", the access to all servers is protected using Kerberos version 5. All communication to servers (such as FTP servers) is protected by the relevant $k_{c,S}$ generated by the TGS (using a secure encryption algorithm).

(a) The Antalyzer accessed his FTP account in the company's server while visiting some client. After a few hours, he tried to access the FTP server again, and found out that his access was blocked. Checking with the system administrator, he found out that someone has found his password and changed it. Explain how the Antalyzer's account was hacked.

To solve this problem, the Antalyzer suggested to change Kerberos, such that the authentication of the users will be done using public key cryptography. Each user will receive two RSA keys pub_c and prv_c and a certificate $Cert(pub_c)$ which is signed by a known and trusted CA.

(b) The Antalyzer suggested to replace the first two messages in the original Kerberos (AS exchange) by:

1. Client → KDC: $ID_c, ID_{TGS}, Cert(pub_c), g^x \bmod p, Sig_{prv_c}(g^x \bmod p), nonce_1$
2. KDC → Client: $E_{K_{c,KDC}}(k_{c,TGS}, nonce_1), Ticket_{c,TGS}, g^y \bmod p$

where g^x, g^y are Diffie-Hellman keys generated each time randomly by the server and the user (assume that g and p are publicly known values), $K_{c,KDC} = g^{xy} \bmod p$, and Sig is a secure signature algorithm.

Compared with the original Kerberos, is the new protocol more secure or less secure (analyze all 7 attacks we've seen in class). In each case, either show an attack or provide an explanation.

(c) Kevin Picnic, who also works at the company claimed that in the new protocol the use of $nonce_1$ is redundant. Is he right? Under which conditions? Explain or present an attack.

(d) Tav Zero claims that the original change is not secure enough, and that certificate authentication should be added to messages 3 & 4 as well. Is he right? Explain or present an attack.

(e) A graduate of our course claimed that the new solution wastes resources, and that the original problem can be solved using a single pair of RSA keys and a certificate for KDC. Let $Cert_{KDC}$ be the KDC's certificate and pub_{KDC}, prv_{KDC} are the KDC's public key and private keys, respectively. The new proposed messages are

1. Client → KDC: $ID_c, ID_{TGS}, g^x \bmod p, nonce_1$
2. KDC → Client: $E_{K_{c,KDC}}(E_{k_c}(k_{c,TGS}, nonce_1)), Ticket_{c,TGS}, g^y \bmod p, Cert_{KDC}, Sig_{prv_{KDC}}(g^y \bmod p)$

Is the new protocol as secure as the one suggested by the Antalyzer? Explain or present attacks.

3. Consider the EKE protocol we've seen in the class.

   (a) Explain why it is impossible to use a dictionary attack to find Alice's password.

   (b) Assume that Eve has recorded successfully an interaction between Alice and the server, and additionally obtained the value of b used by the server. Can Eve now determine Alice's password using a dictionary attack? If so, show the attack, otherwise, explain why it is impossible.

   (c) To improve the efficiency of EKE, it was suggested to change the third message from $E_k(challenge_u, challenge_s)$ into $challenge_u, challenge_s$ (i.e., the same challenges are sent, but unencrypted). Is it now possible to perform a dictionary attack on Alice's password? If so, show the attack, otherwise, explain why it is impossible.

   (d) In addition to the previous change, it was suggested to send in the second message $E_w(b)$ instead of $E_w(g^b \bmod p)$. The reminder of the protocol does not change. Is it now possible to perform a dictionary attack on Alice's password? If so, show the attack, otherwise, explain why it is impossible.

   (e) Assume that the encryption E is AES-128 in ECB mode of operation, and that $challenge_u$ and $challenge_s$ are both of 128 bits. Show an attack on the protocol using such an E.

4. Consider the SRP protocol.
   Throughout this question, we shall assume that Bob is using a weak password.
   Additionally, unless explicitly mentioned, all questions are independent of each other.

   (a) Assume that Eve obtained a list of all the usernames in the system, and a list of all v values, but not the correspondence between them. Additionally, assume that Eve have recorded in the past communications between Bob and the Server.

      i. Can Eve impersonate the server when speaking with Bob? If so, show an attack (or explain why she cannot).

      ii. Can Eve impersonate Bob when speaking with the server? If so, show an attack (or explain why she cannot).

   (b) To save computational resources, it was suggested that the system will use the same b for all of its communications. Can Eve impersonate Bob in these settings? If so, show an attack (or explain why she cannot).

   (c) To reduce the number of rounds in SRP, it was suggested that the fourth message will include M_1, i.e., will be $B = v + g^b; u; M_1$, whereas the fifth message will be M_2. Is the new protocol as secure as the original one? For each of the attacks shown in class, show an attack, or explain why it is impossible to attack the new protocol.