**USC UPSTATE**

**CSCI 455: Computer Security**

**Spring 2019**

**Final Exam**

**Question 1 (20 points)**

(a) [4 Points] Describe the one-time pad encryption scheme. In particular, describe how a message is encrypted with a one-time pad, and how a ciphertext is decrypted with a one-time pad.

* + Requires a one-time key that is pre-shared and never revealed or reused. This key has to be the same length or longer than the plaintext. In this technique, the plaintext is paired with a random secret key (one-time pad). Each char of the plaintext(M) is encrypted with the corresponding one-time pad char(k). (C = EK(M)). The operation performed is an XOR (exclusive or) with the plaintext and one-time pad.
  + We reverse the encryption function to decrypt the message by using the one-time pad and ciphertext. (M = DK(C))

(b) [8 Points] Suppose that two equal-sized messages M1 and M2 are encrypted with the same one-time pad, and let C1 and C2 be the resulting ciphertexts. Show that C1 XOR C2 = M1 XOR M2.

m1: 011001000110111101100111

m2: 011000110110000101110100

key: 011100000110000101110011

c1: 011100110110111101111001

c2: 011100100110000101101100

m1 xor m2: 11011101111101011011100111111

c1 xor c2: 11011101111101011011100111111

If the two encrypted messages are using the same cipher and the same key, C1 xor C2 results in M1 xor M2 where C1 and C2 are the respective ciphertext and M1 and M2 are the corresponding plaintext.

(c) [8 Points] As in Part (b), suppose that two equal-sized messages M1 and M2 are encrypted with the same one-time pad, and let C1 and C2 be the resulting ciphertexts. Suppose further that an attacker captures both ciphertexts C1 and C2. Based on the result of Part (b), describe how the attacker can determine whether the two messages M1 and M2 end with the same bit.

Since the key can only have 2 possible values for each instance (Ki = (0 or 1), you can use the ciphertext to determine what key was generated for each value in the plaintext. If the last character xor = 0, the last bits are the same, if xor = 1, the last bits are different.

**Question 2 (20 points)**

Suppose that a huge file (e.g. 50 GB) is replicated and shared between Alice and Bob who are now far apart and are connected only by a communication channel with very low data bandwidth (e.g. 50 Kb/s). Alice and Bob need to check whether their copies are identical. This problem would have been easy if Alice and Bob were able to exchange and compare their copies. However, since Alice and Bob can only communicate over the given low bandwidth channel, they cannot afford sending lots of data over the channel. Devise an efficient protocol that requires Alice and Bob to communicate only a very small amount of data (e.g. 512 bits) in total, and yet allows them to determine whether their copies are identical with high confidence.

Alice and Bob can check the hash value of their file and compare for accuracy. If the file is even 1 char off, the hash value will be completely different. The hash value can be shortened to a certain section, say the 1st or last 10 chars, and compared to ensure they are exactly the same.

**Question 3 (20 points)**

A software company periodically disseminates updates to its clients. However, before accepting an update, a client must make sure that the update indeed comes from the true software company and is not a piece of malware that a malicious attacker attempts to install on the client's computer. Describe a method by which a client can verify that an update is authentic when it is released by the company, yet will not be fooled into accepting any update that is not released by the company.

The client can look for the specific digital signature of the company within the update. If the digital signature is intact, it is likely that the message is untampered and from the correct source. The client can also use a certificate from a trusted Certificate Authority to verify the messenger’s identity.

**Question 4 (20 points)**

Suppose that you need to share a large (e.g. 1 GB) file privately with a group of 100 users, and that a secure PKI is available so that each user has a public encryption key for RSA that can be looked up and verified through the PKI. Since the file is large and RSA encryption is slow, it is inefficient to encrypt the file 100 times, each time using the RSA key of one of the users. Describe an efficient hybrid method to encrypt the file so that you encrypt the file only once, and each of the 100 users but no one else is able to open the file.

Use a hybrid encryption model where you use public key encryption to share secret keys for private key encryption. You generate a pair (pk, sk) and share the pk with your group of 100 users. The users choose a random secret key, encrypt the message and send to you so that both you and each user share the secret key, which becomes the shared secret. Use public key encryption to encrypt keys, use private key encryption to encrypt data. This method is 100 to 1000 times faster than PKI using RSA and you only encrypt the message once. You encrypt the password once but also encrypt the messages sent with the encrypted password for each individual user. The password is encrypted once but the messages transferred add another encryption layer.

**Question 5 (20 points)**

Describe how passwords should be stored on a server, and give your reasons.

Passwords should be stored as a hash value using a one-way hash function on a server. If the server becomes compromised, the list of passwords is not easily read by the attacker. Adding a salt value, essentially a one-time pad, for each user can further secure the passwords on a server by adding an additional step to decryption. Salt values may not fully stop an attacker from breaking the encryption but can make it much more difficult to break. (Assuming the attacker has not already compromised the server.)

**Question 6 (20 points)**

Some say that in general, it is impossible to design and implement a perfectly secure system. Do you agree with this statement? Justify your answer, from both the technical and the human perspectives.

I agree and disagree. I disagree because I believe it is possible to create a perfectly secure system, at least from bounded attackers, given the correct system security implementation, infrastructure and personnel. The system has to be designed, in every aspect, with security built in. The best way to me is to take a hybrid approach and stack security. For example, your users create strong passwords that are stored on a secure server as a hash value with another salt value for additional decryption. Systems may not be perfectly impenetrable but can be layered with security to become wildly impractical to break.

However, I mainly agree because it is not practical in theory. Humans are the largest security flaw and designing a perfect system likely means it is limited to experienced professionals only. For example, in a security company full of professionals, it only takes one weak security user, like a secretary of the CEO, to be the weakest link and bring down the entire security design.