**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.

Answer: One-time pads are encryption key’s created by selecting a number between 1-26, then doing this for each letter in the message. You then have a number to shift by for each letter in the message. Each letter is then shifted by it’s respected number to it’s new letter. To decrypt the message, the letters are shifted backwards the number so that you get the original letters in the message.  
  
(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.

Answer:

C = M XOR K

M XOR C = M XOR (M XOR K)

M XOR C = (M XOR M) XOR K

M XOR C = K

C1 XOR C2 XOR M1 = M2

C1 XOR C2 = M1 XOR M2

(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.

Answer: To find out if M1 and M2 have the same ending bit, one can XOR the ending bit of C1 and C2. With the XOR operator, if the result is 0, the two bits are the same.

**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.

Answer: To verify that the two files are identical, one can use a hash function such as SHA3-256. Doing this will give a hash key that is unique only to the file. Even with the littlest change to the file, the hash will be completely different. This small key is under the limit to send between Alice and Bob, and will allow them to verify the files integrity.

**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.

Answer: I good system to use to verify origin of files is a signing function. The company can sign the update installation file with their own key. Once the user downloads the file, the user can check the signature to make sure it makes the original host. If the file is altered in any way, there is no way to sign the file again as the original host unless you have access to their private key.

**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.

Answer: One way to do this would to be encrypt the file with one password and share it. Then encrypt the password with each members public key and send them the cipher’ed password. Then each member can get the plaintext pass and decrypt the original file while no one without a sk to the encrypted pass can access it.

**Question 5 (20 points)**

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

Answer: Passwords should be saved as the hash, and not the plaintext password. Saving the password as a plaintext will allow any intruder to know the password. If saved as a hash, there’s no way to find the original password if a good hashing function is used. To make it more secure, the password should be hashed on the client computer. Then sent to the server as to prevent interception of the password while sending/receiving.

**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.

Answer: No, I don’t think it’s possible to design and implement a perfect system. Technology is constantly evolving, so techniques in creating and cracking system are always getting better. One “perfect” system today may be obsolete in a few years due to newer and better systems with the performance and power to crack said systems. This is shown in secure algorithms in the past. Older ones are broken and replaced with newer ones, sometimes replaced before the older can be broken. With that advancement, humans are also constantly learning new things and ways to do stuff. It’s always been a back and forth between the creators and the crackers. As a new secure algorithm is created, there is a cracker trying to break it. With enough time and advancements it will be broken.