Name:

Computer Security: Homework 4 - Comments

Due September 27, 2010 on compass. Shortened late hand in period to October 1, 2010 to ensure adequate time to post answers before exam.

25 points for each question.

1. Alice and Bob use RSA to exchange data. Alice's public key is (e_A, n_A). Bob's public key is (e_B, n_B). Their private keys are d_A and d_B respectively.
   a) Show the computations that Alice would perform to send Bob the message, m, such that only Bob could read the message.

Send to Bob the message encrypted with Bob's public key, i.e., $m^{eb} \bmod n_b$

   b) Bob want's to send Alice a message, m, that only she can read. He also wants her to be assured that Bob created the message. Show the computations that Bob would perform to create such an encrypted message.

Send to Alice the message first signed by Bob's private key and then encrypted with Alice's public key, $(m^{db} \bmod n_b)^{ea} \bmod n_a$

   c) What is the mathematical relationship between Alice's public and private key?

$e_a * d_a \bmod \Phi(n_a) = 1$

   d) The computationally efficient solution of what hard problem will eliminate the strength of the RSA algorithm? Why?

$\Phi(n_a) = (p-1)*(q-1)$. *If we had an efficient algorithm to factor composites of large primes, then we could take the publicly available* n_a *and factor it. Then we could compute* $\Phi(n_a)$ *and use* $\Phi(n_a)$ *and* e_a *to solve for* d_a *just as the system did when it originally computed the key pair for Alice.*

To get full points, the answer needs to discuss some specifics on how an efficient factoring algorithm could be used to recover the private key.

2. Alice and Bob need to communicate securely. They only have symmetric key algorithms like AES available to them. They each have an interchange key (K_a and K_b), and they need to develop a session key negotiation protocol.

a. They start with a straightforward algorithm that assumes each participant has the other's interchange key. The initiator picks a session key and encrypts it with

Name:

its peer's interchange key. E.g., Alice picks session key K_{ab} and sends $\{K_{ab}\}K_b$ to Bob. What kind of attack could Eve launch on this scenario?

Eve could save packets and perform a replay attack. She could send $\{K_{ab}\}K_b$ again to trick Bob into using an old session key again. Then she could resend other messages encrypted with that key. Or she perhaps Eve has broken the key K_{ab} and she can generate new messages encrypted with that key.

b. Needham-Schroeder is a key exchange algorithm that involves a trusted third party. What is a benefit of using a trusted third party as opposed to the direct approach described in part a?

The amount of initial secret sharing is much lower. Each participant only has to share a secret with the trusted third party. This would be N keys for N participants. Only one key would need to be registered with the trusted third party for each new group member.

In the direct case, $O(N^2)$ initial secrets need to be shared. One for each pair of communicating parties. N(N-1)/2 is the precise number of keys.

Another answer is that Needham-Schroeder includes authentication and protection from replay attacks, so Eve could not commit the replay attack identified in part a.

c. What is the purpose of the last exchange of messages between Alice and Bob in the Needham-Schroeder protocol?

The last exchange of messages has Bob sending a nonce encrypted with the new key to Alice. Then Alice decrypts the message, subtracts one from the value, and re-encrypts the result to send to Bob.

This exchange proves to Bob that Alice indeed has access to the newly negotiated shared key.

3. Work with Gnu Privacy Guard (GPG). You can access GPG from http://gnupg.org. I have used this on Linux and installed it via yum on my personal system. I am running it on my Windows system via cygwin. It is already be installed on the University Linux systems. Type "man gpg" to check if it is installed on your system. Once you get your GPG system operational perform the following tasks:
   a. Create a key pair and submit your exported public key. Export your key in the armored ASCII format.
   b. Sign the class public key posted at http://www.cs.illinois.edu/class/fa10/cs461/assignments/cs461-pub.asc with

Name:

your key. Submit the exported signed key. Again export in ASCII using the armored format.

c. Select an ascii file. Encrypt it with the class key and sign it with your key. Submit the signed and encrypted file

4. Alice is writing a secure network messaging library. Confidentiality of the data is less of a concern than integrity. The message will be sent unencrypted.

   a. She is considering using HMAC-SHA or MD5. Which algorithm should she use? Give a reason for the preference.

Since the hash result will be sent along with the message, it must be protected from the attacker. So the keyed hash HMAC-SHA would be most appropriate. The attacker could change the message in transit. But if the attacker does not have access to the hash key, he cannot recompute a new hash value to match the altered message.

The student should only get partial credit if they say HMAC-SHA because the HMAC-SHA has more bits or is newer than the MD5 hash.

   b. If Alice is worried about Eve launching a birthday attack, what order of number of hash tests would Eve have to perform to find a pair of messages that breaks the hash? (for the algorithm chosen in step a)

SHA1 is 160 bits. If the student selected MD5 in part a, they should not lose additional points here for using 128 bits.

By the birthday attack, the attacker would have to generate 2^(80) message pairs to have a better than 50% change of finding a match.

   c. How does the crypto-hash protect the sender and receiver from changes made by a malicious intermediary?

As explained in part b, the attacker doesn't have access to the hash key to generate a new has that matches the new version of the message.

Because the crypto hash is a one-way function, it is computationally infeasible for the attacker to make a version of his message that happens to match the original cryptohash.