ITIS 6200/8200 Principles of Information Security and Privacy

Weichao Wang

Fall Semester of 2018

Homework 2

Hand out: September 12th, 2018

Due time: September 21st, 2018 before 11:00 am

1. We have a symmetric encryption algorithm EK(M)=C. Here K is the secret key, M is the plaintext, and C is the ciphertext. We (and the attacker) know that the key length is 192 bits. The attacker eavesdrops on the communication line and gets a copy of the ciphertext C1. Now the attacker decides to conduct the brute force attack and try every possible key to get the plaintext M1 (that means, every possible key of the 2192 keys). Let us assume that there is only one possible M1 and if the attacker sees it, he will know that this is the correct one. The attacker has 1,000,000 machines, with each machine having the capabilities to try 6,000,000 decryption of C1 with different keys per second. If one machine finds the right key, it will automatically notify the attacker. Now please answer, how many years (roughly) does the attacker need to try 10% of the keys?

Hint: Note that Google has around 3 million servers. So we assume that the attacker is as powerful as 1/3 Google. Also, check the Internet and see what the expected life time of the Sun is. Can you crack the key before that? This question is to show that after a certain key length, brute force is not the best way of cracking it. Side channel, or some other mechanisms will serve the purpose better.

1. Bob has a public-private key pair (pub\_Bob, pri\_Bob). Alice needs to send some information to Bob. She wants to make sure that when Bob opens the message, he can verify that this is from Alice but not anyone else. So she sends out the message as: [ Alice, Epub\_Bob(message) ] to Bob. Basically, she first sends out her name in clear text, then encrypts the message with Bob’s public key. Please discuss, can an attacker M impersonate Alice and send out a packet in Alice’s name? How can he do it? Here we assume that M also has the public key of Bob. For the same question, if Alice sends out [ Epub\_Bob(Alice, message) ], can M still impersonate Alice? (Here Alice puts her name in the encryption.)
2. Alice’s computer stores the files in the following way: for every file F, the computer will calculate the hash value of the file hash(F) and store it after the file. Every time when Alice login, the machine will automatically hash all the files and compare the results to the stored hash values. In this way, if by accident the hard drive is mis-functioning and flips a few bits in a file, Alice can immediately detect it since the hash value will be different. Now an attacker hacks into Alice’s machine and he tries to change several files. The attacker also knows the hash function that the computer uses. Please describe what the attacker needs to do so that the next time Alice login, the machine will not detect the changes. **Also, please discuss how we should improve the mechanism to detect such changes.**

Hint: this is actually directly related to how to use hash functions to protect integrity of data. The defense capability of a hash function and that of a keyed hash function are actually different.

1. Please draw a binary Merkle’s hash tree with 10 leaf nodes. The leaf nodes are labeled as Leaf1 to Leaf10, which correspond to the hash values of the files F1 to F10, respectively. Now please answer:
   1. For each node in the tree, please label clearly how the hash value is calculated based on its children; Please note that the number of leaf nodes is not power of 2. Therefore, you may need to change the way in which intermediate nodes are calculated. There are different ways to handle this. Please label clearly how you calculate each node.
   2. Now the creator of the file F7 needs to verify that his file’s hash value is integrated in the root of the tree. Please show the minimum number of hash values in the tree that the creator needs to accomplish the task. **Please also show how the verification is accomplished.**