CSc 4222/6222 – Assignment #3

Deadline: Thursday, February 1st / Late Deadline: Sunday, February 4th

Make sure to justify all your answers clearly.
No handwritten submissions

---

1. Joe often sends funny jokes to Lisa. He does not care about confidentiality of these messages but wants to get credit for the jokes and prevent Tom from claiming authorship of, or modifying them. He can achieve this by using public-key cryptography and digitally signing his jokes and send each one with its signature. However, as public-key cryptography is computationally intensive and drains the battery of Joe's device, he comes up with an alternative approach. First, he shares a secret key k with Lisa, but not with Tom. Next, together with a joke x, Joe sends over the value $d = h(k||x)$, where each h is a cryptographic hash function. Does value d provide assurance to Lisa that Joe is the author of joke x and that x was not modified by Tom?

SOLUTION: Value d is a message authentication code (MAC), which give Lisa assurance of the authorship and integrity of Joe's jokes. The reason is that a cryptographic hash function is one-way, Tom cannot recover the key k from value d. Thus, Lisa knows that only Joe could have computed value d from joke x. Also, if Tom replaces joke x with a joke of his own, x', it would be infeasible for Tom to compute the MAC value corresponding to x'.

2. Alice and Bob shared an n-bit secret key some time ago. Now they are no longer sure they still have the same key. Thus, they use the following method to communicate with each other over an insecure channel to verify that the key K_A held by Alice is the same as the key K_B held by Bob. Their goal is to prevent an attacker from learning the secret key.

i. Alice generates a random n-bit value R.

ii. Alice computes $X = K_A \oplus R$, where $\oplus$ denotes the exclusive-or Boolean function, and sends X to Bob.

iii. Bob computes $Y = K_B \oplus X$ and sends Y to Alice.

iv. Alice compares R and Y. If $R = Y$, she concludes that $K_A = K_B$, that is, she and Bob have indeed the same secret key.

Show how an attacker eavesdropping the channel can gain possession of the shared secret key.

SOLUTION: The attacker eavesdrops X and Y. The attacker recovers key K_B by computing $X \oplus Y = X \oplus (K_B \oplus X) = (X \oplus X) \oplus K_B = K_B$

3. Suppose you could use 100 characters in the ASCII character set in a password. What is the number of 8-character passwords that could be constructed from such a character set? How long, on average, would it take an attacker to guess such a password if he could test a password every nanosecond?

SOLUTION: There are 100^8 possible passwords with 8 ASCII characters. Guessing a password will take on average ½ (100^8 x 10^{-9}) seconds. This is 5,000,000 seconds or about 58 days.

4. If a password is salted with a 40-bit random number, how big is the dictionary attack search space for a 240,000 word dictionary?

SOLUTION: The search space is 2^{40} x 240,000 = 2.64 x 10^{17} options.

5. Eve has just discovered and decrypted the file that associates each userid with its 32-bit random salt value, and she has also discovered and decrypted the password file, which contains the salted-and-hashed passwords for the 100 people in her building. If she has a dictionary of 500,000 words and she is confident all 100 people have passwords from this dictionary, what is the size of her search space for performing a dictionary attack on their passwords? What if she did not have access to the file that associates userids with the salt values – what would the impact be then?

SOLUTION: The size of the search space is 100 x 500,000, which is 50 million. This is due to the fact that the attacker needs to do the dictionary attack against each person separately, because of the password salt.

6. In order to prevent against stack-based buffer overflow attacks, Kevin is using 10-bit canary with his system. Do you think this will be effective against an attack? Explain why or why not.

SOLUTION: A 10-bit canary only provides 1024 possible canary values. So an attacker can try all possibilities in a fairly short amount of time. As soon as one of them has succeeded, he has compromised the system, and each one has a pretty good, one out of 1024, chance of succeeding.

7. Viruses that perform no explicit malicious behaviors are called *bacteria* or *rabbits*. Explain how such seemingly benign viruses can still have negative impacts on computer systems.

SOLUTION: Bacteria and rabbits can have negative impacts on computer systems, since they use up valuable resources. For example, their replication itself uses up disk space and CPU time.

8. You are given the task of detecting the occurrences of a polymorphic virus that conceals itself as follows. The body, *C*, of the virus code is obfuscated by XORing it with a byte sequence, *T*, derived from a six-byte secret key, *K*, that changes from instance to instance of the virus in a random way. The sequence *T* is derived by merely repeating over and over the given key *K*. The length of the body of the virus code is a multiple of six – padding is added otherwise. Thus, the obfuscated body is $T \oplus C$, where $T = K \| K \| \ldots$ and $\|$ denotes string concatenation. The virus inserts itself to the infected program at an unpredictable location. And infected file contains a *loader* that reads the key *K*, unhides the body *C* of the virus code by XORing the obfuscated version with the sequence *T* (derived from *K*), and finally launches *C*. The loader code, key *K*, and the obfuscated body are inserted at random positions of infected programs. At some point of the execution of the infected program, the loader gets called, which unhides the virus and then executes it. Assume that you have obtained the body *C* of the virus code and a set of programs that are suspected to be infected. You want to detect the occurrences of this virus among the suspected programs without having to actually emulate the execution of the programs. Give an algorithm to do this in polynomial time in the length of the program. Assume that the loader of the virus is a short piece of code that can be commonly found in legitimate programs. Therefore, it cannot be used as a signature of our virus. Hence, looking for the loader is not an acceptable solution. Remember, the loader is in binary, and as such, extracting information from it is nontrivial, i.e., wrong.

SOLUTION: Take the virus code *C* and XOR repeated copies of it with the body of the program, offset by various amounts. One of these will line up with the virus code in the program, in which case XORing *C* with itself in this location will reveal the repeated pattern of the key *K*. So all that is needed is to find the offset that reveals a six-byte key that is repeated for the length of *C*.

9. Suppose you want to use an Internet café to login to your personal account on a bank web site, but you suspect that the computers in this café are infected with software keyloggers. Assuming that you can have both a web browser window and a text editing window open at the same time, describe a scheme that allows you to type in your userID and password so that a keylogger, used in isolation of any screen captures or mouse event captures, would not be able to discover your userID and password.

SOLUTION: Open both the web browser, pointing to your bank's login page, and a text editing window, open to a new un-named file. To enter your userID and password, use your mouse to toggle input between the text editor and the web browser. When you are in the browser window, type a single character of your userID or password and then click back to the text editor window. When you are in the text editor window, type a reasonably-long sequence of random characters. By toggling back and forth between these two windows, you will end up typing in your userID and password, but a keylogger will only see a sequence of random characters with the characters of your userID intermixed in such a way as to be hard to detect. In fact, you could cycle through all the characters on the keyboard for each character in your userID and password, clicking to the browser just for the appropriate character needed in each cycle and then immediately click back to the text editor. In this case, a keylogger would only see a repeated series of sequences of all the characters on the keyboard.

10. Assume a network of 100 people, such that each pair wise communication is possible at one time or another. What would be the number of keys required to encrypt all communication if using shared secret keys vs public key encryption?

SOLUTION: Public key encryption: 100 public keys and 100 private keys = 200 total.
Shared secret key encryption: 100(100-1)/2 = 4950 keys total.

Grad students only:
Read the following paper – you can find an online copy here. Provide a 1-2 page summary of the paper.

B. M. Padmanabhuni and H. B. K. Tan, "Light-Weight Rule-Based Test Case Generation for Detecting Buffer Overflow Vulnerabilities," *2015 IEEE/ACM 10th International Workshop on Automation of Software Test*, Florence, 2015, pp. 48-52.