UNIVERSITY OF MARYLAND
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

ENEE 459C
Computer Security
Instructor: Charalampos Papamanthou

Homework 3

Out: 11/01/13 Due: 11/15/13

Instructions

1. Strictly adhere to the University of Maryland Code of Academic Integrity.
2. **Type** and submit your solutions as a pdf document at Canvas. Include your full name in the solutions document. Name the solutions document as x-hw3.pdf, where x is your last name.
3. No extensions will be given. Everything will be due at 11.59pm on 11/15/13.

Problem 1 (20 points)

In class, we talked about the Tor protocol which is based on *onion routing*. Onion routing works as follows: When user A wants to send message x to B that has public key pk_B, A downloads the public keys $pk_1, pk_2, \ldots, pk_m$ of m intermediate stations $1, 2, 3, \ldots, m$ and uses these keys to encrypt message x. How does A use onion routing to encrypt message x? How does B decrypt this message? State one security property that onion routing aims to provide to user A.

Problem 2 (20 points)

Suppose you know that the passwords used by the students to authenticate to the university server consist of n uppercase letters of the English alphabet. Given the hash $h(p)$ of a password p describe three solutions to compute password p.

1. The first solution uses constant space, $O(26^n)$ query time and no preprocessing.
2. The second solution uses $O(26^n)$ space, constant query time and $O(26^n)$ preprocessing time.
3. The third solution uses $O(26^{\frac{2n}{3}})$ space, $O(26^{\frac{2n}{3}})$ query time and $O(26^n)$ preprocessing time.

Because of the above attacks, the administrator changes the way passwords are stored: Instead of storing $h(p)$ for password p, the administrator stores $(h(p||r), r)$, where r is some randomness that is chosen for each password. Which one of the above solutions will not work now? Why?

Problem 3 (20 points)

As discussed in class, a rainbow table is characterized by the length of the chains, t, and the number of chains, N. To advance along the chain, you are using some specialized hardware which performs the following operation: First it hashes a password and then it uses a reduction function to derive the next password in the chain.

1. What is the maximum number of such hardware operations required for recovering the password? Your answer should be a function of t.

2. Suppose now you have assigned 10^8 bytes of your computer memory for storing the rainbow table. Each row of the rainbow table takes 10 bytes of memory, and the rainbow table covers 10^{14} passwords. If you want your rainbow table to crack a password in *at most* 10 minutes, show that your hardware should perform *at least* 83.33 operations per nanosecond. For convenience, assume that the cost of searching for a hash value after you have reached the end of the rainbow table chain is negligible.

Problem 4 (20 points)

Given the confidentiality categories TOPSECRET, SECRET, CONFIDENTIAL, UNCLASSIFIED and the integrity categories HIGH, MEDIUM, LOW, indicate and explain what type of access (read, write, both, or neither) is allowed in the following situations.

1. Paul, cleared for (TOPSECRET, MEDIUM), wants to access a document classified (SECRET, HIGH).
2. Anna, cleared for (CONFIDENTIAL, HIGH), wants to access a document classified (CONFIDENTIAL, LOW).
3. Jesse, cleared for (SECRET, MEDIUM), wants to access a document classified (CONFIDENTIAL, LOW).
4. Sammi, cleared for (TOPSECRET, LOW), wants to access a document classified (CONFIDENTIAL, LOW).
5. John, cleared for (SECRET, LOW), wants to access a document classified (SECRET, LOW).
6. Robin, who has no clearances (and so works at the (UNCLASSIFIED, LOW) level), wants to access a document classified (CONFIDENTIAL, HIGH).

Problem 5 (20 points)

In this programming assignment, you are going to exploit the account of user raceme at a server running at IP 107.20.43.191, by writing into raceme's files without having explicit write permission. You can log into the server by using the account user@107.20.43.191 with password user. In order for multiple users not to interfere with each other despite all being logged in as the same user, the home directory of user is read-only. If you want to write programs or save files or anything, you can create a directory in /tmp and save your files there by using

mkdir `/tmp/some_unique_directory_name`

You will exploit a TOCTOU (time of check-time of use) race condition in a setuid program owned by user raceme (as we discussed in class). Once you log in as user@107.20.43.191 you'll see a program you can run, i.e., the program /home/user/append. Running

/home/user/append `string filename`

will add a new line containing `string` to the end of the file named `filename`. The append program is setuid raceme, so it runs with the permissions of the raceme user. The source code to the append program is in /home/user/append.c. Exploit a vulnerability in the append program to append your name to the file /home/raceme/list. For 15 bonus points, read the contents of the file /home/raceme/file.txt.

You can find information on race conditions and techniques to exploit them here. For a walkthrough of how to exploit the specific kind of access/open race seen in the append program, see here and here.