

CS 378 - Network Security and Privacy

Fall 2007

Homework #1

Due: 3:30pm CST (in class), September 27, 2007

YOUR NAME: _____

Collaboration policy

No collaboration is permitted on this assignment. Any cheating (*e.g.*, submitting another person's work as your own, or permitting your work to be copied) will automatically result in a failing grade. The Computer Sciences department code of conduct can be found at <http://www.cs.utexas.edu/users/ear/CodeOfConduct.html>

Late submission policy

This homework is due at the **beginning of class** on **September 27**. All late submissions will be subject to the following policy.

You start the semester with a credit of 4 late days. For the purpose of counting late days, a "day" is 24 hours starting at 2pm on the assignment's due date. Partial days are rounded up to the next full day. You are free to divide your late days among the take-home assignments (3 homeworks and 2 projects) any way you want: submit four assignments 1 day late, submit one assignment 4 days late, *etc.* After your 4 days are used up, no late submissions will be accepted and you will automatically receive 0 points for each late assignment.

You may submit late assignments to Vitaly Shmatikov (TAY 4.115C—slide under the door if the office is locked). **If you are submitting late, please indicate how many late days you are using.**

Write the number of late days you are using: _____

Homework #1 (50 points)

Problem 1 (5 points)

Your task is to enhance security of password-based access control to a database system. The database has approximately 50 users, each of whom has an 8-character password. Characters may include lower- and upper-case letters, digits, or special symbols.

You decided to store database passwords as salted hash values. What's the smallest size (in bits) for the random salt value that would, in your opinion, provide reasonable security for this system? Explain your reasoning. (Hint: at the very least, no two users of the system should end up with the same salt value.)

Problem 2

A big software company has just released an advanced video game player called Zbox 180. For a monthly subscription fee, gamers can join Zbox Half-Alive, a new online service with multi-player games, chatrooms, and bulletin boards.

User authentication for Zbox Half-Alive works as follows. When the user first subscribes for the service, she must establish a password. This password is stored on the Half-Alive server together with the hard-coded serial number of the user's Zbox 180. Afterwards, whenever the user's Zbox connects to the server over the Internet, she is asked for her password, which is transmitted in the clear together with the serial number of the Zbox. The server verifies whether the received password matches the password in its database and whether subscription fees have been paid for this serial number. If so, it allows the user to connect.

Problem 2a (4 points)

Suppose your neighbor has a paid-up Half-Alive subscription. He is using a wireless Internet connection for his 3am gaming marathons, and the signal leaks into your room (*i.e.*, you can passively eavesdrop on all messages transmitted to and from his Zbox, but you cannot modify them or introduce new messages). Describe how you can exploit this to connect your own Zbox to the Half-Alive server for free. (Assume that you can modify your Zbox.)

Problem 2b (8 points)

Design a user authentication scheme for Zbox Half-Alive based on a cryptographically secure hash function that prevents passive attackers from exploiting eavesdropped messages between the Zbox and the Half-Alive server.

Problem 3

`BakeORama.com` is a big online seller of baked goods. To set up an account on the website, a user must create a username and a password. Because the Web server might be vulnerable to a hacking attack, the IT department of `BakeORama.com` decided that it is too dangerous to keep passwords stored on the server, so they came up with a clever way to avoid having to remember every user's password.

When the user creates a new account, his password is stored in a Web cookie. When he comes to `BakeORama.com` again and types in his username and password, the site pulls the cookie from his browser and compares the typed-in password with the password stored in the cookie. If the two passwords match, access is granted.

Problem 3a (4 points)

Describe how you can log into another user's account on `BakeORama.com`. (Assume that the victim's computer is offline and inaccessible; all you know is his username.)

Problem 3b (5 points)

Design an authentication scheme in which passwords are stored in cookies, but the attack you discovered in Problem 3a is no longer feasible.

Problem 4

Molvanian Telecom has rolled out Molvania's first Web-based email system. After the user authenticates to the system's Web server, the server stores a cookie (called `SessionCookie`) in the user's browser so that all subsequent requests from this user do not require authentication.

The new email messages are displayed in the user's Web browser using the following HTML template:

```
<HTML>
<BODY>
--- Headers appear here ---
<DIV ID="msg">
--- Email message is displayed here ---
</DIV>
</BODY>
</HTML>
```

Problem 4a (5 points)

Give an example of an email message that you could send to a user of this Web-based email system and that would allow you to read all of that user's email.

Problem 4b (5 points)

How would you modify the Web-based email system to prevent the attack you discovered in Problem 4a?

Problem 5 (6 points)

Here is a new scheme for encrypting messages that are shorter than 128 bits. The scheme is based on the Rijndael algorithm, which is believed to be a secure block cipher. First, pad the message to 128 bits, obtaining a 128-bit block m . Next, generate a 128-bit random string r , apply plain Rijndael (*i.e.*, no initialization vectors) to encipher r with the symmetric key, and XOR the result with the message. The ciphertext is the pair $(r, \text{Rijndael}(r, \text{key}) \oplus m)$.

Is this encryption scheme secure against the chosen-plaintext attack? Explain your answer.

Problem 6 (8 points)

MMACs (Molvanian Message Authentication Codes) are intended to provide authentication and integrity for email messages between Molvanian diplomatic missions. All missions share the secret key K . Each message M sent by one mission to another is accompanied by a MMAC, which is constructed as $HBH(K, M)$, where HBH is a certain hash function.

HBH (stands for Home-Brewed Hash) is a hash function invented by Molvanian cryptologists. They took SHA-1 (which is assumed to be one-way and collision-resistant—at least for the purposes of this problem) and used it as a building block to create HBH . They even *proved* that HBH , too, is one-way and collision-resistant.

As it turns out, MMAC is completely broken. Someone eavesdropping on even one MMAC-protected message call can gather enough information to forge valid MMACs in the future, that is, to send any message she wants accompanied by a forged MMAC which will

pass verification by any Molvanian mission.

How is HBH constructed? (Important: your construction must be one-way, collision-resistant, and still make the above scheme vulnerable to forging.)