CS177: Computer Security
Prof. Stefano Tessaro

UC Santa Barbara
Fall 2018

Homework 5

Posted: Wednesday, November 14, 2018 – 11:59pm
Due: **Tuesday**, November 27, 2018 – 11:59pm – online submission only!

Task 1 – Web Security & Cookies[1] (6 points)

CryptoTube is a major player in the emerging market for streaming cryptography-related contents. It enables two different types of subscription products to its registered users: *gold access* ($10/month) and *silver access* ($5/month).

Upon login, the server at `www.cryptotube.com` checks the membership type for the user in a database, and remembers it by setting a cookie:

access-type=**access**; domain=www.cryptotube.com; path=/

where **access** can be either of `silver` or `gold`. (Further cookies are set to keep track of the successful login and a session identifier.) Then, upon accessing each web page on `https://www.cryptotube.com/`, the server checks the `access-type` value in the cookie (which is sent along with the HTTP request), and determines which contents are displayed.

a) Why is the above approach to distinguish "gold" from "silver" users likely not effective? Explain!

After learning of the above problem, CryptoTube engineers have adopted the following solution. The CryptoTube server stores (locally) a secret key K for HMAC. Instead of letting *access-type* equal either of *silver* or *gold*, it now equals the string

$$type \;\|\; session\ ID \;\|\; tag$$

where *type* is ether *silver* or *gold*, *session ID* is the ID of the current session (contained in a different cookie), and *tag* is the output of HMAC with secret key K and input *type* || *session ID*. (As usual, || denotes concatenation of strings here.)

Then, upon accessing any web page on `https://www.cryptotube.com/`, the server looks at the `access-type` value in the cookie (which is sent along with the HTTPS request) to recover *type* || *session ID* || *tag*, and checks whether *session ID* matches the current session (which has been sent along in a different cookie), and if *tag* is correct (i.e., it uses the secret key K to recompute the tag, and checks if it equals *tag*). If so, it displays the contents according to *type*, and if not, it gives an error message.

b) Explain why this is a better solution, and in particular bypasses any attack you described in **a)**.

[1] This question was part of the Fall 2017 final exam.

Task 2 – Find the Secret Phrase (8 points)

Retrieve the hidden secret phrase from the following website:

```
http://192.35.222.247/hw5/pt1/
```

Explain in detail what you did, and include the secret phrase in your solution.

Task 3 – A Curious Administrator (16 points)

Retrieve the administrator's password for the following website:

```
http://192.35.222.247/hw5/pt2/
```

We heard that the admin is addicted to the site and checks everyone's comments very frequently. Unfortunately, the admin is also very easy to offend, and ends up deleting all accounts and all comments after reading them!

Explain in detail what you did, and include the password in your solution. Please read the hints below!

Hints. Please read carefully the following hints:

- For the simplest approach, you will need a little knowledge of Javascript and HTML. Please refer to

  ```
  http://www.w3schools.com/html/
  http://www.w3schools.com/js/
  ```

  for tutorials, but feel free to ask questions to the TAs, lecturer, or on Piazza.

- You may need to set up some small "attacker web site" allowing you (for example) to capture a string. To this end, we have provided some web-space for you on our class VM. Using your account for HW4, you can create a `public_html` subdirectory which will be accessible as

  ```
  http://192.35.222.247/~yourusername/
  ```

  The server supports php7. Please be aware that (1) the Apache webserver runs as `www-data`, and that user needs to be able to access your `public_html` directory and the files you want to be accessible (and editable) by the webserver. Note that by default, files and directory are created so that this is not possible, so you have to change access rights.

- A small php script called `capture.php` allowing you to store a certain value in a file, is provided on Piazza. It can be invoked e.g. as

  ```
  http://192.35.222.247/~yourusername/capture.php?value=hello
  ```

and ends up appending to a file `data.txt` (you may need to create first) the value `hello`.

- You are not allowed to retrieve the data from the accounts of fellow 177 students who are not careful and may make it accessible.
- You have to set access permissions of files so that other students can not read them, but the webserver can access what necessary. In particular, we recommend you execute:

  ```
  chmod 701 /home/yourusername
  chmod 701 /home/yourusername/public_html
  chmod 602 /home/yourusername/public_html/data.txt
  ```

  where the latter command refers to the file you use for `capture.php`.