CS 6324: Information Security
Project 3 - Web Security

In this project, you will perform a series of attacks to exploit some vulnerabilities in an online bulletin-board, called *hackme*. Your attack will consist of several phases. You will conduct cross site scripting, cross site request forgery, and SQL injection attacks. You are also required to implement modifications to the web site to prevent such attacks.

Instructions

Due date & time 11:59pm CST on April 30, 2021. Submit your project files (the report and source files) to eLearning by the due time. The files you submit must be compressed to a single archive (.zip or .tar).

Additional Instructions

- For this project, you already have the source files for the website hackme compressed `hackme.zip` in your home directory in `fiona.utdallas.edu`.
- Each student will use their account on `fiona.utdallas.edu`. You will receive an email with this information.
- Use the UT Dallas VPN if you connect to the server from outside the campus: `https://oit.utdallas.edu/howto/vpn/`.
- Your home account will contain a folder called 'public_html'. This is your website's home directory, which you can use for testing your solution. You can access any file placed there online on:
  `http://fiona.utdallas.edu/~username/`
  (remember to keep your permissions appropriate, you may need to modify them so that the group 'www-data' has read and execute access).
- To set up you own version of the website, unzip `hackme.zip` to the web directory above.
- All your website instances will connect to the same mySQL database. The database name is `cs6324spring21`, the username to access the database is `cs6324spring21` and the password is `ebBUwdGQunTa8MFU`.
- Any code you write should run on `fiona` with no errors.
- The written portion of the project must be typed. Using Latex is recommended, but not required. The submitted document must be a PDF (no doc or docx are allowed)
- If you want to use late days for this project (with the penalty described in the course website), please notify TA (dongpeng.liu@utdallas.edu) by the due time. Otherwise, your account on `fiona` will be locked and you will not be able to modify your code.

1 (20 pts) Password and Session Management

Securely managing sessions and passwords is vital to prevent a number of attacks on a webpage. Most website designers, however, neglect to take a few important security measures when dealing with passwords and cookies. In this part, you are going to explore hackme's code and identify a set of exploitable vulnerabilities.

1. **Password Management** hackme manages passwords in an inherently insecure way. Familiarize yourself with the password management techniques employed by hackme by examining the following files: `index.php`, `register.php`, and `members.php`:

   - **Describe** how the passwords are stored, transmitted and authenticated.
   - **Identify** and **describe two** vulnerabilities in the password management system and explain **how** they can be exploited. (note: your answer should not involve the possibility of "weak passwords" and should not involve cookies)
   - **Describe** and **implement** techniques to fix the above vulnerabilities. Your description should be thorough and should indicate which files/scripts need modification and how they should be modified. You should then write code to fix the vulnerabilities. Your code will be graded on how well it fixes the vulnerabilities. No points will be given if the code gives runtime errors.

2. **Session Management** hackme relies on cookies to manage sessions. Familiarize yourself with the how cookies are managed in hackme.

   - **Describe** how cookies are used in hackme and how sessions are maintained. Include in your description what is stored in the cookies and what checks are performed on the cookies.
   - **Identify** and **describe three** vulnerabilities in the cookie management system and **how** they can be exploited.
   - **Describe** and **implement** techniques to fix the above vulnerabilities. Your description should be thorough and should indicate which files/scripts need modification and how they should be modified. You should then write code to fix the vulnerabilities. Your code will be graded on how well it fixes the vulnerabilities. You will not get any points for code that gives runtime errors.

2 (20 pts) XSS Attack

Cross Site Scripting (XSS) attacks are one of the most common attacks on websites and one of the most difficult to defend against. Popular websites, including twitter and Facebook, were once vulnerable to such attacks. Naturally, hackme is also vulnerable to XSS. In this part, you will perform XSS attacks to steal users' cookies.

1. Craft a special input string that can be used as part of a posting to the bulletin board to conduct a XSS attack. The attack should steal the cookies of any user who views your posting. The cookies must be stored in a file on the attacker's server (not storing the cookies will only get you partial credit). You need to:

   - Provide the *exact* input string you need to post to the webpage. The input should be typed and submitted in a file names (XSS_input.txt) (I should be able to copy your string and paste it in order to replicate your attack). To get full credit, your attack must be 100% hidden from the victim.
   - Explain what your string does and how it performs the attack. Also describe how the attacker can access the stolen cookies. Be precise in your explanation.
   - Provide any extra web pages, files, and/or scripts that are needed to conduct a successful attack. Provide a complete description of each item and its intended purpose. Include in your description the required linux/unix permission bits for each new file.

- Describe the exact vulnerabilit(y/ies) that made your attack possible.

2. **Describe** *and* **implement** a method to prevent this attack. Your description should be thorough and should indicate which files/scripts need modification and how they should be modified. You should then write code to fix the vulnerability. Your code will be graded on how well it fixes the vulnerability. You will not get any points for code that gives runtime errors.

3 (20 pts) XSRF Attack

Cross Site Request Forgery (XSRF) attacks are malicious exploits which allow unauthorized commands to be transmitted to a webpage on behalf of a victim user. The attack can be used to perform unauthorized transactions on the victim's behalf. In this part, you will perform a XSRF attack to post an advertisement to the website without the user's consent. While the user is still logged on to hackme, you need lure him/her to a malicious webpage that runs the attack.

1. Create a new webpage that runs the XSRF attack. The attack should post an advertisement for "awesome free stuff!!". You need to:

   - Describe the components of your webpage. Include a thorough description of the component performing the attack and explain **how** the attack is performed.
   - Make sure that the attack is 100% stealthy (hidden from the victim). The victim should only visit/view the malicious website for the attack to work. Attacks which require user interaction or which are not hidden (ex: cause redirection) will only get partial credit.
   - Identify a method of luring the victim to visit your malicious website while he/she is logged into hackme.

2. Describe the specific vulnerabilit(y/ies) in hackme that allowed XSRF attacks. Be precise in your description.

3. Describe **three** methods to prevent XSRF attacks on hackme. You need to be thorough in your description: mention **what** needs to be done, **how** to do it, and **why** it will prevent the attack.

4 (20 pts) SQL Injection Attack

For this part, you will use a private version of the website available on:

`http://fiona.utdallas.edu/hackme/`

Note that this version uses a different database instance which uses login credentials that are not available to you.

In an attempt to limit access to this bulletin board, we added one more verification step to the registration process. Now, only users that have been given a secret access key can register as users. When creating a new account, the user has to enter a secret key. The hash of this key is checked against a stored hash. If it matches, then the registration process continues normally. This is the only time the key is checked.

Unfortunately, the secret key was not given to you, and you cannot register on this website. For this attack, you will bypass this requirement by performing an SQL injection attack.

1. By crafting a special input string to one of the HTML forms, you need to perform an SQL injection attack to register a new user on the website. You need to:

   - Identify the webpage you are using for the attack (i.e. `index.php` or `register.php`)

- Provide the *exact* input to *every* field on the webpage; this should include your attack string (I should be able to copy your string and paste it in order to replicate your attack from a file you should send named SQL_string.txt). To get full credit, your attack should not return an SQL error.
- **Execute** the attack to register a new user. The username should be your NetID (ex: dxl200000). The first name and last name should be your name. The password can be anything.
- Login with new username, and **post** something on the bulletin board. The title of the post should be your full name.

2. **Describe** *and* **implement** a method to prevent the above SQL injection attack. Your description should be thorough and should indicate which files/scripts need modification and how they should be modified. You should then write code to fix the vulnerabilities. Your code will be graded on how well it fixes the vulnerabilities. You will not get any points for code that gives runtime errors.

3. The way the secret key is handled is inherently insecure. **Describe** a more secure method of providing the extra authentication step. That is, assuming that an adversary **can** perform the SQL injection attack, how can you prevent him from logging in to the website?

Note: You can assume that "magic quotes" are disabled in the php.ini file by the system administrator. You cannot override this setting.

5 (20 pts) Weak Passwords

For this part, you will use a private version of the website available on:

`http://fiona.utdallas.edu/hackme/`

Note that this version uses a different database instance which uses login credentials that are not available to you.

As you can tell, hackme does not check the strength of a user's passwords. The website only enforces the condition that passwords should be non-empty. As a result, 100 users registered accounts with very weak and/or common passwords. In this part, you will run a brute force dictionary attack to recover the passwords.

1. Read the paper "Testing Metrics for Password Creation Policies by Attacking Large Sets of Revealed Passwords" by Weir et al. (`pwdtest.pdf` is attached in eLearning).
   - Describe the current NIST standard for measuring the strength of passwords.
   - Briefly outline why, according to the paper, the NIST standard is ineffective.
   - Based on your understanding, suggest a set of rules that can be employed by hackme to prevent "weak passwords"

2. The file `users.txt` contains a list of 100 users with weak passwords. You need to perform a **brute-force *dictionary*** attack to recover these passwords. For this, you need to find a corpus of weak passwords (a number of them are available online).

   You should output your result in a text file called `pass.txt`. Each line should correspond to one user. You need to output the username, followed by a tab, followed by the password. The users should be in the same order as they appear in `users.txt`. If you are unable to recover a password for one user, then output the username alone on that line. That is, I should be able to run `diff` on your output and my answer file in order to get the number of incorrect answers you have. Failure to follow these rules will not get you any credit.

This part is worth 10 pts (i.e. 0.1 points per password) and points will be rounded up. That is, you only need to recover 91 passwords to get a full grade. If you recover all 100 passwords, you will get bonus points. Hint: No password is used twice. If you are missing a few passwords after your dictionary attack, think about bad password habits made by users.

6 Final Notes.

You should remember the followings:

- Even though you have your own version of the web site *hackme* they all share a common database, so make sure that you *play* nice.
- Make sure to compress all your solutions and follow the naming convention giving in the specification. The grading process will be automated and failing to follow the instruction might result in loosing points EVEN if you submitted the correct solution.
- Make sure you provide all the implementation of the fixes of the security vulnerabilities in *hackme* pointing out in your report to all the file you have changes and **what** you have changes and **why** it is important.
- Finally make sure you have a running final version in your account and clean up any backups you have made.