Security

Thesis security description

Final Year

Alexandru Sulea
D Stream
#12315152
N April 2017

Contents

List of Tables

1 Overview

One of the many reasons for which Amazon Web Services was chosen to host the thesis project was that the service not only has a data centre located in Ireland but also that the service provides a large variety of other services such as machine learning and DNSSEC [1]. Amazon Web Services provides documentation on all of the provided services especially DNSSEC, most importantly there are clear instructions on how to adequately secure and protect ones website or server from malicious attacks.

Due to the statistics platform from the thesis storing and using user data in a mongo dB database, the data requires a number of security considerations, not only to keep it secure from being accessed publicly but also so that the wrong user signs into an account.

Firstly, with all login systems, a password, username and email are required to create an account. The email is required to ensure that the account creation is unique and not created by a bot. This is important, if a bot can create a profile on a platform then it can create n+1 profiles on a platform until the server crashes due to not being able to deal with the high, rapid volume of data being created. It also goes without saying that this data does not store anything important only, whatever presents the bot/s had programmed to set. It also does not contribute anything to the platform other than the entire thing being turned into a testing bed for click farm bots. The email guarantees on some part that the user is at least a real person, although the same person could have multiple emails and multiple accounts on the platform this is acceptable, the large population of the online community are too preoccupied to have two or more profiles on the same platform. The last reason why emails are important is for communication reason, the email is in many ways the only and primary way in which a platform can communicate with its users in an official manner.

The username is required to provide the users with an option to be public or private when they communicate to other users on the platform. This is an important aspect as many individuals around the globe may use a platform for communication or expression without the desire to be public about their identity. Other however may wish to use the platform to connect to other users for project work or CV building, thus would wish that they were easily identifiable by others. The username structure should not be unsupervised though, it should encompass the following pre-requisites:

- 1. Each username should be unique. the uniqueness is calculated based on the characters , numbers and symbols in the specific username.

  a. Thus, the first person to claim the username userone, will in effect own that username, thus if another person wishes to be named userone, then the second person will be informed that they cannot create an account with such a username due to that username already being in use.

  b. UserOne will be the same as userone, but combinations such as userone1 are considered different enough from the original to be regarded as different.

  c. UserOne will be the same as userone, but combinations such as userone:) are considered different enough from the original to be regarded as different.

- 2. A standard format for what usernames are appropriate from a semantic point of view should be clearly defined. The new usernames should meet al the standard name pre-requisites, such as:

  a. The username need to be longer than 3 characters.

  b. The username cannot be completely comprised of numbers

  c. The username cannot be completely comprised of symbols

  d. The username cannot be completely comprised of any one or two repeating characters, such as HAHAHAHA etc.

  e. The username should not be comprised of any derogatory or swear words.

  f. The username cannot be comprise of db instructions such as drop().

These username restrictions exist to avoid typo squatting of other peoples profiles by malicious attackers. Thus usernames will need to be unique in the view of the community so that one user cannot pretend to be another and post offensive or defamatory material.

The password is a very important aspect of any online website with users. The password is the key that opens the door to users account page and thus is one of, if not the most important aspect of online security. The platform in this case does not host any financial, medical or energy data, thus the security consideration does not specifically require two factor authentication or using phone numbers etc.. The password should at least fit the standard password model [2]. Thus a password should have a length of at least eight characters of which at least one character should be a capital and at least one character should be a number. An extra security implementation could be to also introduce symbols in the required characters section and encourage a password to be longer than fourteen characters in length [2] due to most password cracking software assuming that the password will be less than fourteen characters. The user should also be discouraged from using actual words in the password and the characters and symbols showing up on the left hand side of the keyboard at the beginning of the keyboard. This is due to most American targeting password-cracking software placing a higher probability on the left hand side characters of the keyboard showing up at the beginning of the password.

- In this manner the password is adequately protected from brute force password attacks where every type of combination is attempted until the password is cracked, this method is in theory full proof but due to the billions of combinations the attackers may end up years waiting for a result.
- The password is also adequately protected from dictionary attacks, as the user is not discouraged from using words and names as well as common phrases or combinations such as (qwerty) or (123456).
- The user should also be discouraged from using the email which was used to create the user account and anymore than 3 characters found in the email same.
- To avoid any storage issues, delimiting characters such as spaces, , commas, /, tabs and pound signs should also be rejected from being part of the password as they would cause confusion in the database which may create 3 tokens instead of the one.
- A mixture of characters which can be accessed by Shift and Alt , thus to use the full range of the character sections available on the keyboard.
- Lastly the password should especially reject the usual phrases such as, drop() or collection(). Having these pre-requisites in place can ensure that the password is robust enough to hold up against a relatively competent attack.

Once the password, username and email pass the verification process the user should be met with a javascript message informing them that the account has been successfully created but not verified. This is a very important second step employed by all user creation platforms today for account verification. This tep exists to not only verify that the new account was created by a human and not the newest version of bots, but also to verify that the user has access to the email they provided and can be communicated with. If this process fails, the user should not be allowed to sign into the account due to the not having access to the email they provided or the email no longer being active.

To perform email verification an email service such as REDIS is used to send emails to the different users, REDIS is easy to integrate with other services but has a built in module for Gmail which is used in this project. Thus when a user signs up for an account the user receives a verification email using REDIS which attaches a token to the email, this token is a unique random number based on the unique number associated with the account given by MongoDb, this verification token is encompassed in a link to the verification side of the platform such as https://someaddress/verification/¡uniquetokenid¿ this link will redirect the user to the verification page which will perform a post request to the server with the unique id. If the server verifies the id then the user is allowed to sign in, otherwise the user is redirected to a failed to verify page. The unique id can fail to be verified if the expiration time passes, this is important due to the aim of not having a large volume of unverified profiles in the system. Thus if an individual wishes to sign up and create a profile they will need to verify their profile within 24 hours or else the token will expire and the profile will be deleted from the system. Once the user has been verified, the setting in the database under the verified attribute should be changed from unverified to verified.

After the user has signed in, cookies should be used to cache the password and make it easier to gain access to the platform without having to repeatedly write in the password and email. The session handlers are also important in ensuring that for example user x does not share the link to their profile and in effect allow other to explore their profile without being verified as that specific user. Thus with session handlers, if a user does share their respective profile http, then anyone but the rightful user will be redirected to a sign in page.

On the backend side, the AWS instance ports should be carefully set, the website should not have any unnecessary ports open and https should be selected instead of http as the platform hosts user data. The https sign can also re-assure users that the website is legitimate as well as somewhat secure. Moreover the Mongodb database port should be changed from the standard 27017 to another less conspicuous port to avoid a malicious web crawler dropping the database through an open port.

The specific domain will also need a domain name, Amazon web services provides an initial aws address of where the instance is stored but not a specific .com domain name. Choosing a domain name can also be considered an important part of security but also marketing. The domain name must be distinctive enough from all other existing domain names, but descriptive enough to what is being offered by this address. Like in many other things of course here are always exceptions such as goDaddy which sells domain names but does not have market or sell in its own name. Typo squatting is a passive attack, where an attacker will try to phish for users by naming their domain similar to that of another popular domain, thus instead of facebook.com someone may buy fecebook.com, the distinction is small enough that it may be missed by someone, who might log into fecebook.com and give the attacker their personal email along with their password. The attacker will then email the user a verify message, only this message will grant the attacker access to the users computer and so on. Unfortunately there is very little that can be done about this other than buy different combinations of ones domain. A .com domain is always more trusted by users than say a .io due to these domains being there from the inception of the early commercial internet, however it is a lot harder to find the chosen domain names with a .com extension.

The last security concern would be a dos or ddos attack, this can be easily harder form by accessing the aws services for Cloudfare or Aiprotect which can use a wide variety of distributed server s across a large network together with advanced filters to mitigate or reduce the effects of such attacks. On https enabled pages this may be a lot harder to perform as htps uses tcp which is a lot easier to clog up than an attack on a http page.

Beyond the official security measures, the users need to be reminded to never write down their passwords and to never give anyone credentials to their personal profiles or click on emails from the platform which seem suspicious.

References

[1] A. W. Services. (2017) Amazon web services. [Online]. Available: http://docs.aws.amazon.com/Route53/latest/DeveloperGuide/domain-configure-dnssec.html

[2] Microsoft. (2017) Password requirements. [Online]. Available: https://msdn.microsoft.com/en-us/library/cc875839.aspx