Contents

Intro

In this research document I will be talking about identification and authentication for simple websites. Websites have become a commonly used platform for companies to communicate and share their products through. Often, these websites have some form of login/user functionality for their clients. The main question for this research is how I user authentication can be done safely in the online web world.

What is identification and authentication?

Identification

User identification refers to the process of uniquely identifying and distinguishing individual users accessing online services, websites, or applications. Any application where you can have an account would need user identification. Think of your daily applications like Spotify, YouTube and WhatsApp. It involves assigning and managing unique identifiers to users to personalize their experience and enforce security measures.

User identification can take various forms, depending on the context and requirements of the web platform. Common practices involve usernames, email addresses and ID's as unique identifiers.

https://blog.admixer.com/user-id/

Authentication

Authentication in the online world refers to the process of verifying the identity of an individual or entity attempting to access a system, website, application, or online service. Authentication is important to protect sensitive information from being compromised and accessed by those who are not allowed to see this.

The most common form of user authentication is using credentials. This can include usernames, passwords, PINs, security tokens, or other forms of identification. Combining your user identification with authentication methods such as credentials ensures only you can access your private user information.

https://www.idrnd.ai/5-authentication-methods-that-can-prevent-the-next-breach/#:~:text=What%20Is%20User%20Authentication%3F,credentials%20like%20username%20and%20password.

Why is authentication important?

As mentioned before, the reason for proper authentication is to deny access to sensitive information against those who are not allowed to access it. According to the OWASP (Open Web Application Security Project), failure of identification and authentication is the seventh most common failure in web application security. This is an improvement from 4 years ago, as back then broken authentication was the second most common failure in web application security.

The damage flawed authentication can cause differs from web application. Your Netflix or Spotify account being hacked may be annoying but not the end of the world. The hacker might learn your real name and email address, but no real harm can come to you.

Other scenarios might be more harmful. Your bank account or medical records being accessed can have a severe impact on your life. This is why proper authentication is important to protect private information.

https://owasp.org/Top10/A07_2021-Identification_and_Authentication_Failures/

How to implement authentication

ASVS

The OWASP Application Security Verification Standard (ASVS) creates a guideline for developers to implement in their security. The ASVS was written to help normalize security practices for web applications and consists of a list with requirements for secure development.

The chapter about Authentication in the ASVS starts with password security. Passwords such as PINs, passwords and unlock patterns (such as one to open your phone) are categorized as “Memorized Secrets”. Memorized secrets are considered single-factor authentication. The OWASP strongly recommends using multi-factor authentication. Another way to improve on single-factor authentication is the use of Credential Service Providers (CSPs). CSPs provide federated identity for users. Users will often have an existing identity with big enterprises such as Google. Allowing users to authenticate themselves through a CSP like Google on a different website to increase the strength of their security.

The ASVS continues to talk about requirements passwords should have for a stronger authentication process. It lists commonly used tactics such as verifying a minimum password length, allowing users to change their password requiring their old password too and a strength meter for passwords.

The document continues describing common security protocols for other actions regarding authentication like credential storage and recovery, one-time verification and out of band verification. The ASVS is a must-read when having to work with application security as a developer.

https://owasp.org/www-project-application-security-verification-standard/

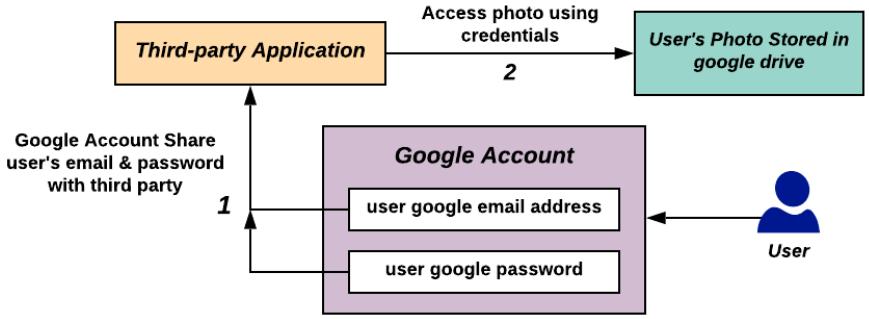
OWASP Application Security Verification Standard 4.0.3-en.pdf

OAuth

What is OAuth?

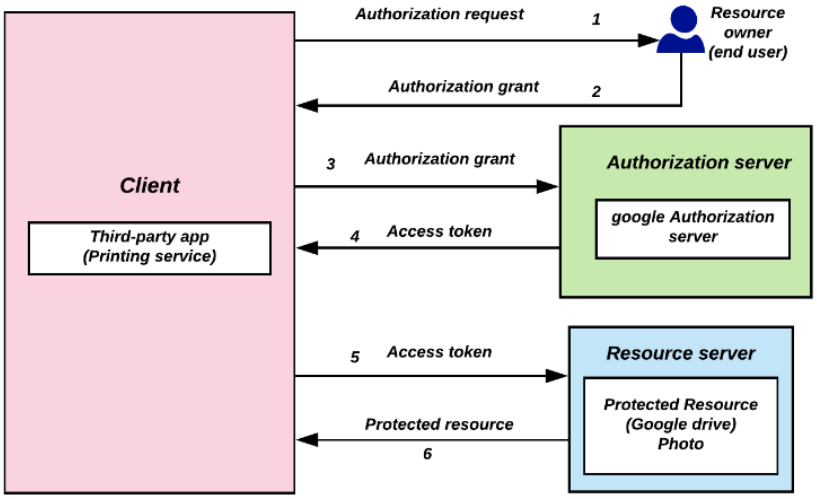
OAuth is an industry-standard protocol for authentication and authorization. It allows users to grant limited access to their resources on one website or application to another website or application without sharing their credentials.

OAuth first started in 2006 when third-party websites would ask for the user's username and password from another site to access theirs. Sharing your authentication from one site with another is a big security risk. Let's say you were trying to log in to a social media website using your Google account. Before OAuth, you would have to directly log-in with your Google account on their site. This would expose your authentication to their site, giving them access to use your account without your supervision.



Thanks to OAuth this is no longer the standard course of action. Instead of directly logging into your Google account on the third-party application, you are now prompted to log-in on a Google website itself.

The client (third-party application) initiates the OAuth flow by making an authorization request to the user. The user is then prompted to authenticate and authorize the client's access to their protected resources. This step usually involves presenting a login screen or consent screen. This screen is not from the client. It would be from the application you are trying to grant limited access to, like Google. Once the user provides their consent, the client makes a request to the authorization server to verify this request. If access was granted by the user, an access token will be given to the client. The client can then use this access token to retrieve the protected resource from the resource server on the user's behalf.



With this method, users never share their login credentials with the third-party application.

https://en.wikipedia.org/wiki/OAuth

https://auth0.com/intro-to-iam/what-is-oauth-2