ZDMP Blog

Secure Authentication and Authorisation

By Leticia Montalvillo. IKERLAN Research Centre

Some questions for you

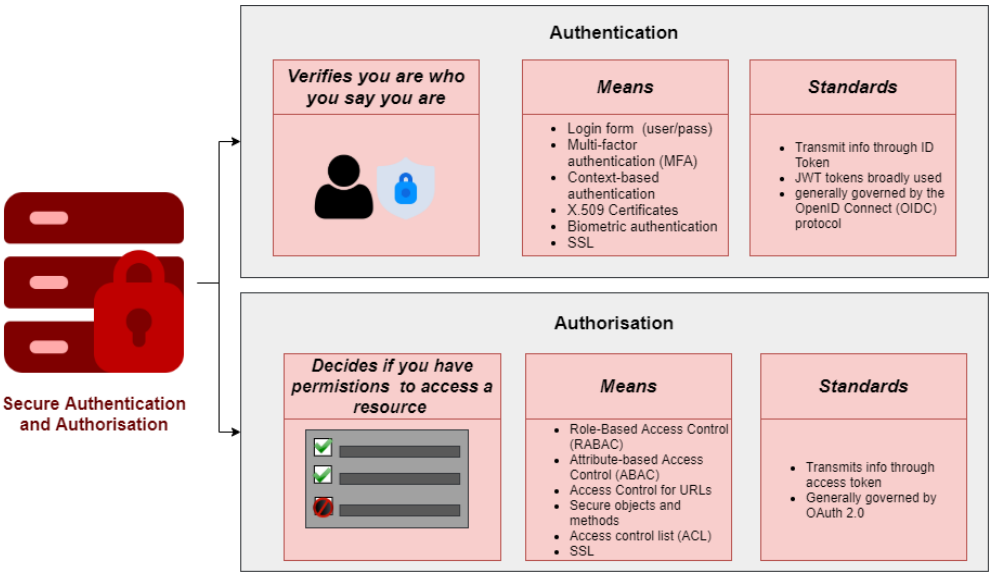
- Is your company providing secure authentication and authorisation mechanisms?
- Has any identity fraud happened due to stolen credentials within your organization?
- Do you implement additional authentication means to verify user's identity?
- Do you need provide different accesses for users based on their business roles? Has any data been disclosed to users within your company due to badly configured access?

Motivation

Authentication and authorisation are key to protect and secure an organization's data. These terms are often used interchangeably, although, fundamentally, these are different:

- **Authentication** is about validating user such as like username and password to verify user's identity. The system (authentication server) determines whether you are who you say you are, using your credentials. This is the first step towards accessing any protected resource a company is protecting. However, using username and password as the only way to verify the identity of a user is a weak approach. Indeed, there are already a considerable number of tools and techniques capable of breaking users' passwords relatively easily [1]. That is why multi-factor authentication mechanisms have become commonplace today
- **Authorisation** is performed after successful authentication and it is about validating the identity of unauthorized users trying to get access to an application, API, microservices, data, or other asset. In other words, authorisation is about granting an authenticated user the permission to perform a given action on a specific resource. The lack of appropriate authorisation mechanisms can cause the disclosure of sensitive data.

Both authentication and authorization are required to deal with sensitive data assets and need to go hand-by-hand. Without any of them, company data would be vulnerable to data breaches and unauthorized accesses. The figure below depicts commonly used means and standards for both secure authentication and authorisation processes:



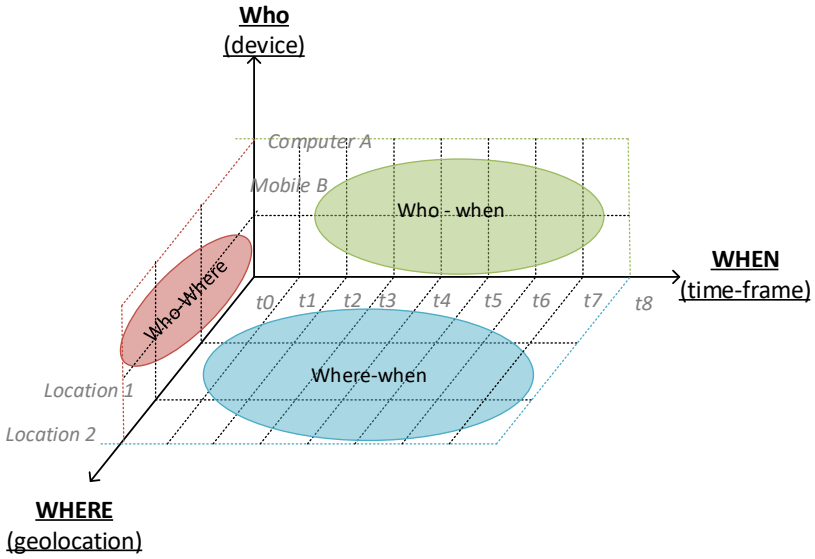
Measures for secure authentication and authorisation

Authentication is usually performed by a username and password, and sometimes in conjunction with other factors of authentication, which refers to the many ways to be authenticated (ie Multi-factor Authentication). Common protocols and standards for user authentication include OpenID Connect [2] and SAML [3]. For machines and devices, X.509 certificates provide the means to authenticate in the network.

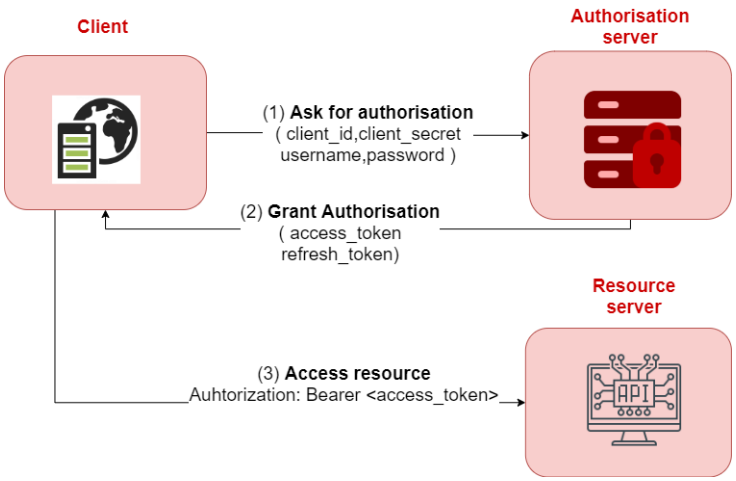
There is a trend towards additional authentication methods to avoid the use of passwords and to strengthen the authentication process. The more factors that are used for authentication, the less are the chances for an intruder to

log-in. One of these additional authentication methods are OTP (One Time Passcodes) which are alphanumeric series that are connected by text message, email, or generated by means of OTP generator applications such as Google Authenticator or Microsoft Authenticator. Hence, once the user introduces their user/password credentials, if these are proven correct, the user is asked to enter the OTP code which is sent/generated eg via their mobile phone.

More modern user authentication mechanisms include biometric authentication which proves the identity of users by scanning of their fingerprints (or eyes), and context-based authentication, which can stablish a set of parameters that define the context of a user and can elucidate any intrusion when any of these parameters change. Context-based authentication can harvest user's context in the following dimensions (see next figure): The "who" dimension (the device being used to login), the "when" dimension (the usual timeframe of login), and the "where" dimension (geo-location of the device performing the login). If any login attempt does not fall within the user's common context (a login from different country, device, or timeframe) the user would be required to prove their identity with an additional factor.



Authorisation starts after the authentication process completes, ie deciding if the authenticated user has permissions to access the requested resources. One approach is to implement the OAuth protocol [4], an open authorization standard that provides secure access to server resources. Following the OAuth 2.0 standard, clients (applications requesting access to protected resource) need first to be granted access by the authorisation server. The following diagram depicts the commonly followed process:



The client (the application requesting access to a given protected resource) needs to request authorisation from the Authorisation server which includes sending several parameters along with the request. The authorisation server then validates the request, and if everything is fine, the client receives an access token (short-lived token containing authorisation information) and a refresh token (token that serves for requesting additional access tokens before access tokens expire). The client then accesses the protected resource and passes along the access token already issued by the

authorisation server. The resource server would check the access token information to grant or deny access to the protected resource the client wants to access. Commonly, access control to this resource are managed by previously defined policy, where user roles and attributes play a central role, eg certain user roles are granted access to only certain resources. Indeed, common access control mechanisms are Role-based Access Control (RBAC) and Attribute-based Access Controls (ABAC). When following the OAuth protocol additional considerations need to be considered to enable a secure authorisation process [3]:

- **Always use SSL**. Using OAuth 2.0 without SLL is just like sending a password in a plaintext across an insecure Wi-Fi connection
- **Always check the SSL certificate** to protect from the man-in-the-middle attacks
- **Do not store client secrets in the database in plaintext**; store the hashed value instead
- **Always use refresh tokens** and make access tokens short-lived

What will ZDMP achieve

ZDMP will follow OpenID protocol for authentication means and will implement context-based authentication together with multi-factor authentication for secure authentication. To enable a secure authorisation process, OAuth 2.0 protocol will be implemented together with Role-based Access Control (RBAC) and Attribute-based Access Control (ABAC) mechanisms, for managing access to protected resource.

ZDMP Links

• **Architecture Component(s)**	Secure Authorization Secure Authentication
• **Work Package**	WP65– ZDMP Core Services
• **Tasks**	T6.2 – Secure Business Cloud

References

[1] Hu, Gongzhu. (2018). On Password Strength: A Survey and Analysis. 10.1007/978-3-319-62048-0_12.
[2] OpenID Connect. https://openid.net/connect/
[3] SAML specification. http://saml.xml.org/saml-specifications
[4] OAuth. https://oauth.net/2/
[5] https://www.esecurityplanet.com/mobile-security/5-tips-on-using-oauth-2.0-for-secure-authorization.html