Evaluating Performance Impacts in Identity Management based on Keycloak and OpenID Connect

Carlos D. S. Bunn[1], **Charles C. Miers**[1]

[1]Computer Science Department
Santa Catarina State University (UDESC)

carlos.bunn@edu.udesc.br,

charles.miers@udesc.br

Abstract. *Using third-party identity providers (IdP) to allow authentication and authorization lets web developers add different IdPs easily. Centralized authentication/authorization services, such as the robust Keycloak framework, provide a reliable and straightforward solution for web developers, ensuring easy and efficient authentication and authorization management. However, Keycloak may introduce some latency and increase the payload of the system traffic. We analyze Keycloak employing OpenID Connect and solely OpenID Connect performance, focusing on identity and token delegation and characterizing their impacts and behavior.*

1. Introduction

While identity management is a simple concept in theory, authorization and authentication are elements usually required to work together, providing essential security aspects and are expected to propose a good user experience [Wilson and Hingnikar 2022]. Tools and frameworks such as OpenID Connect and Keycloak are relevant, as they facilitate the implementation and configuration of robust security mechanisms on various platforms and applications [Silva and Thorgersenm 2021]. These technologies enhance security and are designed to improve the user experience by offering agile, practical, and integrated authentication and authorization methods. They ensure that only authorized users have access to sensitive service data without creating unnecessary barriers for the user. They aim to achieve an ideal balance between protection and usability, allowing users to enjoy secure services and minimum effort [Wilson and Hingnikar 2022].

Asymmetry and fluctuations are usual in computer networks, demanding research studies about assessing the quality of service under different operational demands [Dollimore and Kindbergr 1998]. In this context, benchmarks become a means to compare the performance of technologies such as OpenID Connect, both in isolation and in conjunction with Keycloak, obtaining information regarding latency and payload within applications. Our comparison is relevant to evaluating the impacts of integrating a centralized operation service in a specific architecture.

This work is organized as follows. Section 2 presents basic concepts about OpenID Connect and Keycloak Section 3 describes our analysis scenarios and metrics. Section 4, shows our results and analysis.

2. Fundamentals

OpenID Connect is a solution based on OpenID and OAuth 2.0, enabling users to authenticate in applications and services using third-party IDPs. It provides an additional layer of security to ensure that only authorized users have access [Wilson and Hingnikar 2022], turning it widely used in situations requiring identity management.

Keycloak is an open-source identity and access management solution, providing advanced authentication and authorization features for web applications and services [Silva and Thorgersenm 2021]. Supporting a variety of protocols (e.g., Single Sign-On (SSO), OpenID Connect, and SAML), Keycloak aims to simplify the administration of authentications and authorizations across multiple IdP services and applications. Although using Keycloak allows for easy integration with various authentication protocols, it is also a simple point of failure. Additionally, Keycloak acts as an intermediary and can increase communications latency. Thus, it cannot be denied that there is a tradeoff between performance and security, especially in computing applications with strong scalability requirements. Finally, it is necessary to sanitize inputs and effectively manage the communication [Dollimore and Kindbergr 1998], as these processes are crucial for authorization and identification.

3. Proposal

We defined a baseline scenario (Fig. 1), employing only OpenID Connect to measure our metrics on the simplest scenario.

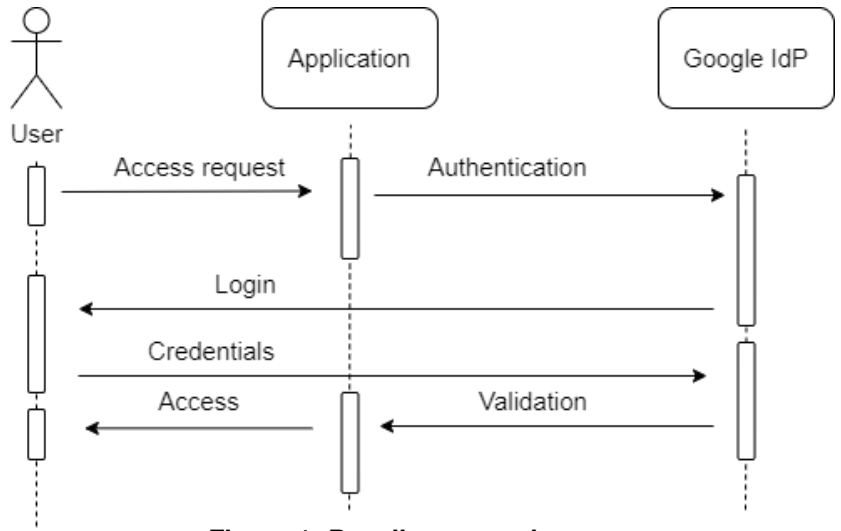

Figure 1. Baseline scenario.

The baseline scenario shows the authentication process in which a user, to gain access to the application, must initially request authentication through OpenID Connect, thereby strengthening security with third-party technology. Ensuring the security of the system, the authentication process involves the user's credentials for identification and validation. This stringent process is designed to grant access to the application only to authorized users.

Our extended scenario (Fig. 2) employs *Keycloak* to manage user identification through the application, as well as to manage the authentication system in conjunction

with OpenID Connect, thereby providing an additional layer of security for credential-related matters. In the extended scenario, for the user to gain access to the application, they must go through Keycloak to obtain access and then provide the credentials to OpenID Connect, which, in turn, grants access to the application.

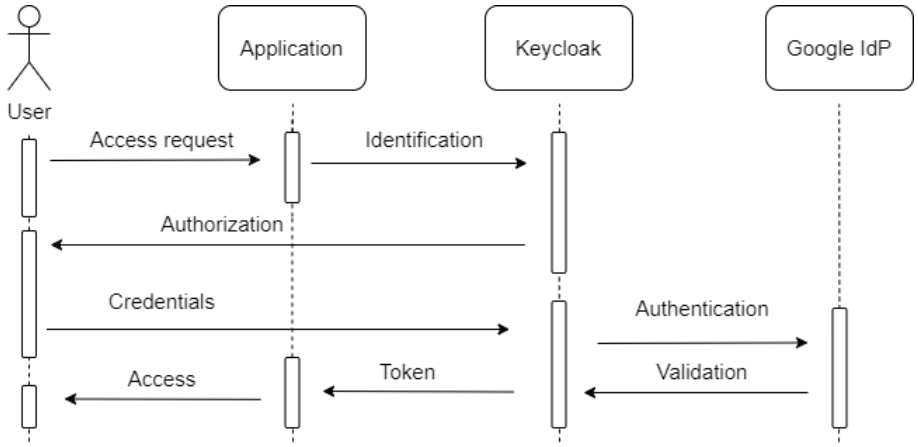

Figure 2. Test base scenario with Keycloak.

Observing how Keycloak is positioned between the application and IdP, it becomes essential to understand and discover the payload and latency involved in this process. The additional payload is necessary due to the interoperability of Keycloak, which has access to the JSON Web Token (JWT) used by OpenID Connect to carry private information of the system's users [Silva and Thorgersenm 2021]. Thus, we aim to measure data related to identification, authentication, and authorization during the access process, up to the release of the aspect requested by the user in the system, recording the measurement of latencies and payloads generated in this communication.

4. Testbed and Initial results

For the benchmarks, an architecture based on Kernel-based Virtual Machine (KVM) over GNU/Linux Ubuntu LTS 24.04 server was deployed on Intel i7 machine (8 cores) on LabP2D/UDESC. We deploy two VMs, Each VM has a flavor of 1 vCPU, 8Gb RAM, and 20Gb storage. OpenID Connect services and client applications were developed using Python3[1]. The application employs Keycloak version 23.0.7 and OpenID Connect managed by Google IdP. Prometheus was employed to collect the data, monitoring the requests occurring when the user accesses the application until the access is granted.

The results are divided into three parts: (i)Application, (ii) OIDC, and (iii) Keycloak. Thus, the Keycloak payload is increased due to the permission tokens and regular transactions of the framework. Each color represents the respective entity in the sequence diagram, evaluating the latency and payload. The payload consists of the data, commands, and files accumulated and transferred between the related entities. The specific content within the payload varies according to each entity that participates in forming the packet [Silva and Thorgersenm 2021]. Fig. 3 and 4 show our initial measurement results.

[1]https://github.com/carlossbunn/OpenID-Connect

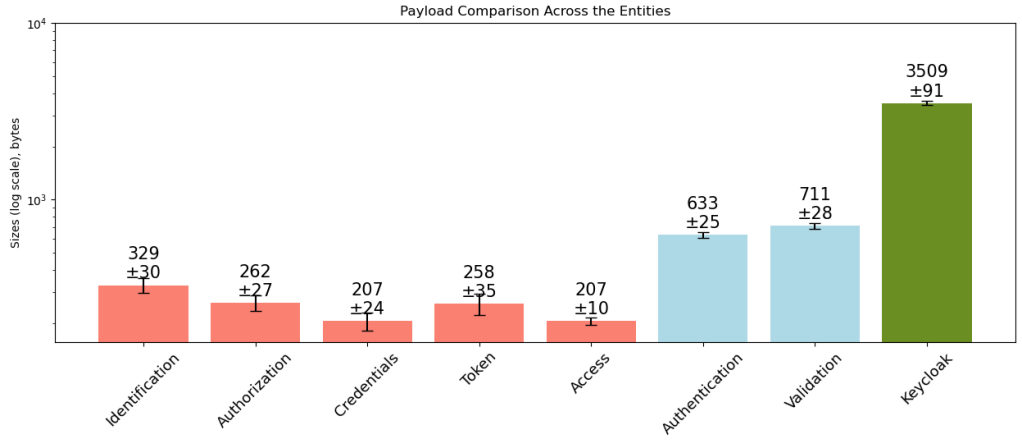

Figure 3. Payload results.

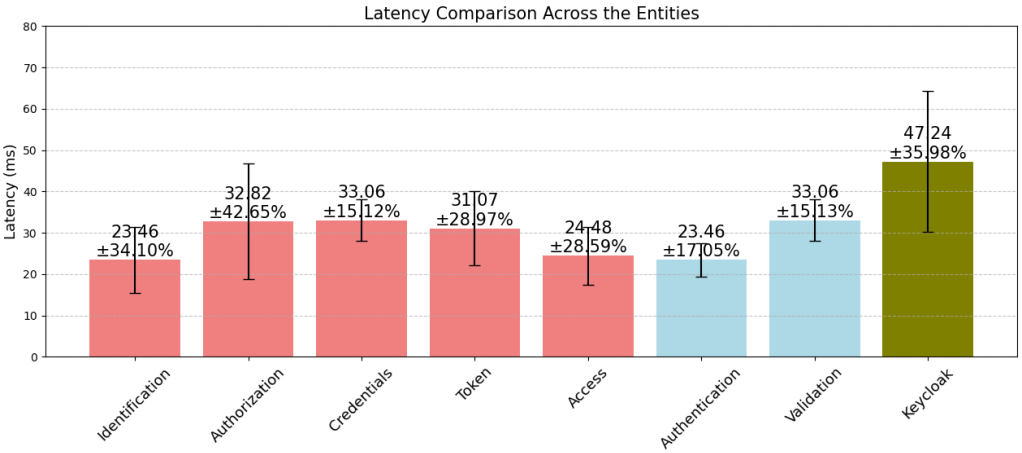

Figure 4. Latency results.

5. Considerations & Future work

As expected, in terms of Round Trip Time (RTT) traffic in the payload, Keycloak stands out in Credentials, presenting a significantly higher volume than OpenID Connect alone. This is due to the amount of information it needs to manage, including several claims, for SSO management. Being advantageous for web applications, Keycloak reduces the number of tokens in communication, managing network congestion more efficiently. A chain linking of the data transmitted by the entities will be implemented for future work.

Acknowledgement. This work was supported by FAPESC, and developed at LabP2D/UDESC.

References

Dollimore, J. and Kindbergr, T. (1998). *Fundamentos de Sistemas Distribuídos*, chapter 2, pages 38–67. Bookman, 4 edition.

Silva, I. P. and Thorgersenm, S. (2021). *Keycloak - Identity and Access Management for Modern Applications: Harness the power of Keycloak, OpenID Connect, and OAuth 2.0*. packet.

Wilson, Y. and Hingnikar, A. (2022). *Solving Identity Management in Modern Applications: Demystifying OAuth 2.0, OpenID Connect, and SAML 2.0*.