Continuous Authentication and Authorization using Verifiable Credentials

Nikos Fotiou, fotiou@aueb.gr

Introduction

This document presents a few ideas stemmed from the project Enabling Zero Trust Architectures using OAuth2.0 and Verifiable Credentials (ZeroTrustVC), which was supported by eSSIF-lab. ZeroTrustVC implemented a solution for achieving continuous authorization of HTTP requests exploiting Verifiable Credentials and OAuth 2.0 capable of transparently protecting HTTP-based resources. In this project, which was motivated by the the Zero Trust principle that requires authentication and authorization of every request, we experimented with existing standards and active drafts for managing the lifecycle of VCs. Given the vivid activity in this area, we want to discuss our key insights from this project and propose specific research directions.

Use of OAuth 2.0 for issuing VCs

Although OAuth 2.0 is now being considered as a mechanism for issuing VCs, little attention has been given to the client credentials authorization grant. ZeroTrustVC considered a VC issuer that allowed users to self-"register" their wallets and assign them access rights; then a wallet can request a VC using client credentials authorization grant. This approach has many advantages:

- Using this grant, all communication is happening in the "back channel" using HTTP POST
- The client/wallet receives the VC immediately and there is no need for supporting redirection url (this includes custom schemes such as openid://)
- Using this grant there is no need to support PKCE
- There is no need for managing "access tokens" and "refresh tokens"

Use of proofs of possession as an alternative to VPs

Verifiable Presentations (VPs) are used by a VC holder in order to prove to a verifier that it posses a VC that includes specific claims. In the current VC data model, a VP "embeds" the corresponding VC, which is cumbersome when a VC is encoded as a JWT (e.g., this example from the VC data model specifications). In ZeroTrustVC we followed an alternative approach: we decoupled VPs from VCs and we leveraged existing solutions used for proving the possession of a public key. In particular, we relied on the OAuth 2.0 Demonstrating Proof-of-Possession at the Application Layer (DPoP) IETF-draft. Based on this draft, a user can include in an HTTP request a "proof of possession" of a key, which is just a digital signature over the request parameters and a random nonce. We believe that using a proof-of-possession which is decoupled from the VC not only results in more “elegant” outputs but it also enables re-use of existing related efforts (e.g., DPoP, FIDO2).

Use of browser-based wallets

ZeroTrustVC implemented a VC wallet as a Firefox browser extension. The main property of that wallet is that it is capable of detecting if a user tries to access a protected resource and if this is the case, it injects the corresponding VC and proof of possession as HTTP headers in the outgoing HTTP request. This is achieved by associating each VC with an “audience”, using the `aud` JWT claim. Although this might seem use-case specific, we argue that it is useful for the issuer to be able to specify the intended audience of a VC (e.g., a governmental issuer may want to specify that an issued VC should be used only with other governmental verifiers). This can be used a countermeasure against VC “phishing”.

Use of transparent VC verifiers

ZeroTrustVC implemented a VC verifier operating as an HTTPS proxy that intercepts the communication between a client and an HTTP(S)-based protected resource. The VC verifier is able to verify the validity, the status, and the ownership of a VC. Additionally, the VC verifier acts as a policy enforcement point by validating whether or not a VC can be used for executing a particular request over a protected resource. ZeroTrustVC verifier validates VCs based on rules expressed using JSON Path, which is a query language for JSON. Tools like JSON Path not only enable the configuration of VC verifiers but they can act as a building block for implementing VC/VP exchange protocols.

Acknowledgements

The work reported in this document has been supported in part by European Union’s Horizon 2020 research and innovation programme through subgrant Enabling Zero Trust Architectures using OAuth2.0 and Verifiable Credentials (ZeroTrustVC) of project eSSIF-lab, and by the Research Center of the Athens University of Economics and Business.