DID-based Auth Protocols

Vurucu ve Akillica Alt Başlık

Abdulhamit Kumru

Blokzincir Laboratuvarı

2020

Auth Protocols & DID Auth

- OAuth & OpenID Connect
- Single Sign On
- Self-Issued OpenID Connect Provider DID Profile (did-siop, DIF)

OAuth

!!! not: bir uygulumaya tum izinleri vermektense sadece gerekli olan izinleri vermek, ben google a login olayim ama benim google a login oldugumu goren servis sadece izin verdigim ismim ve mailimi alabilsin

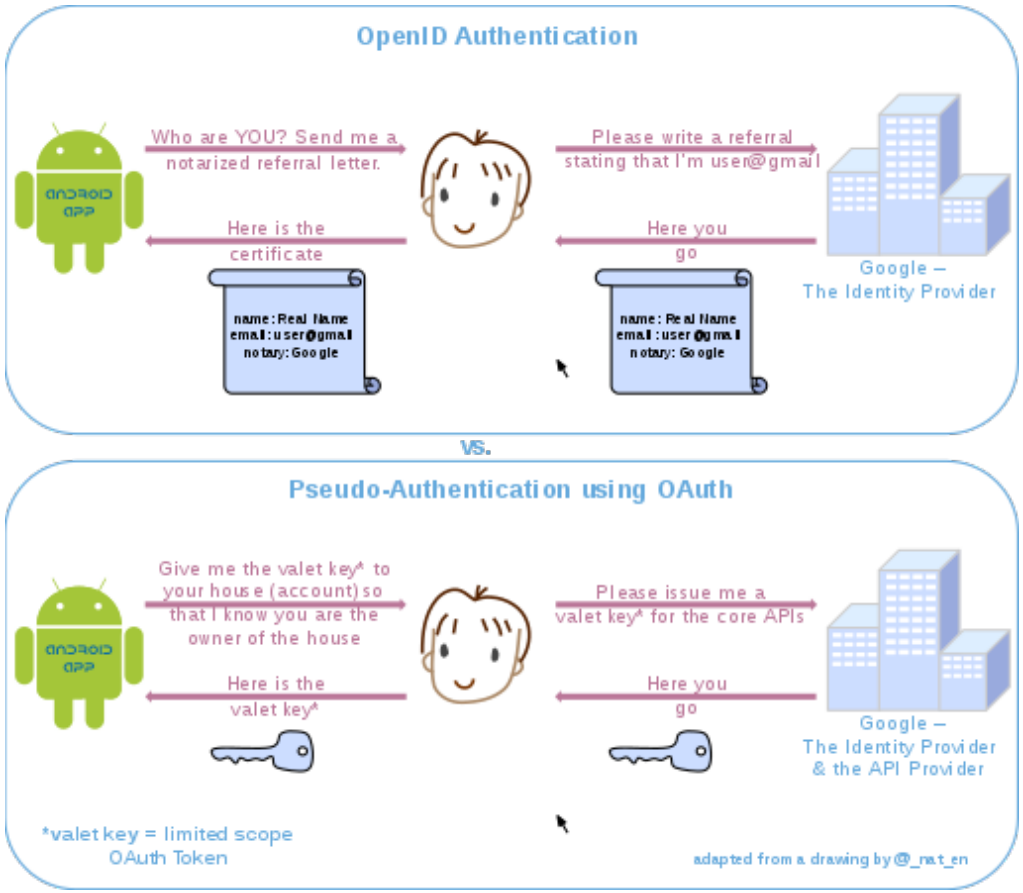
OAuth is an open standard for access delegation, commonly used as a way for Internet users to grant websites or applications access to their information on other websites but without giving them the passwords

OAuth

- OAuth is an authorization protocol, rather than an authentication protocol
- OAuth is directly related to OpenID Connect (OIDC), since OIDC is an authentication layer built on top of OAuth 2.0
- OAuth essentially allows access tokens to be issued to third-party clients by an authorization server, with the approval of the resource owner

OAuth

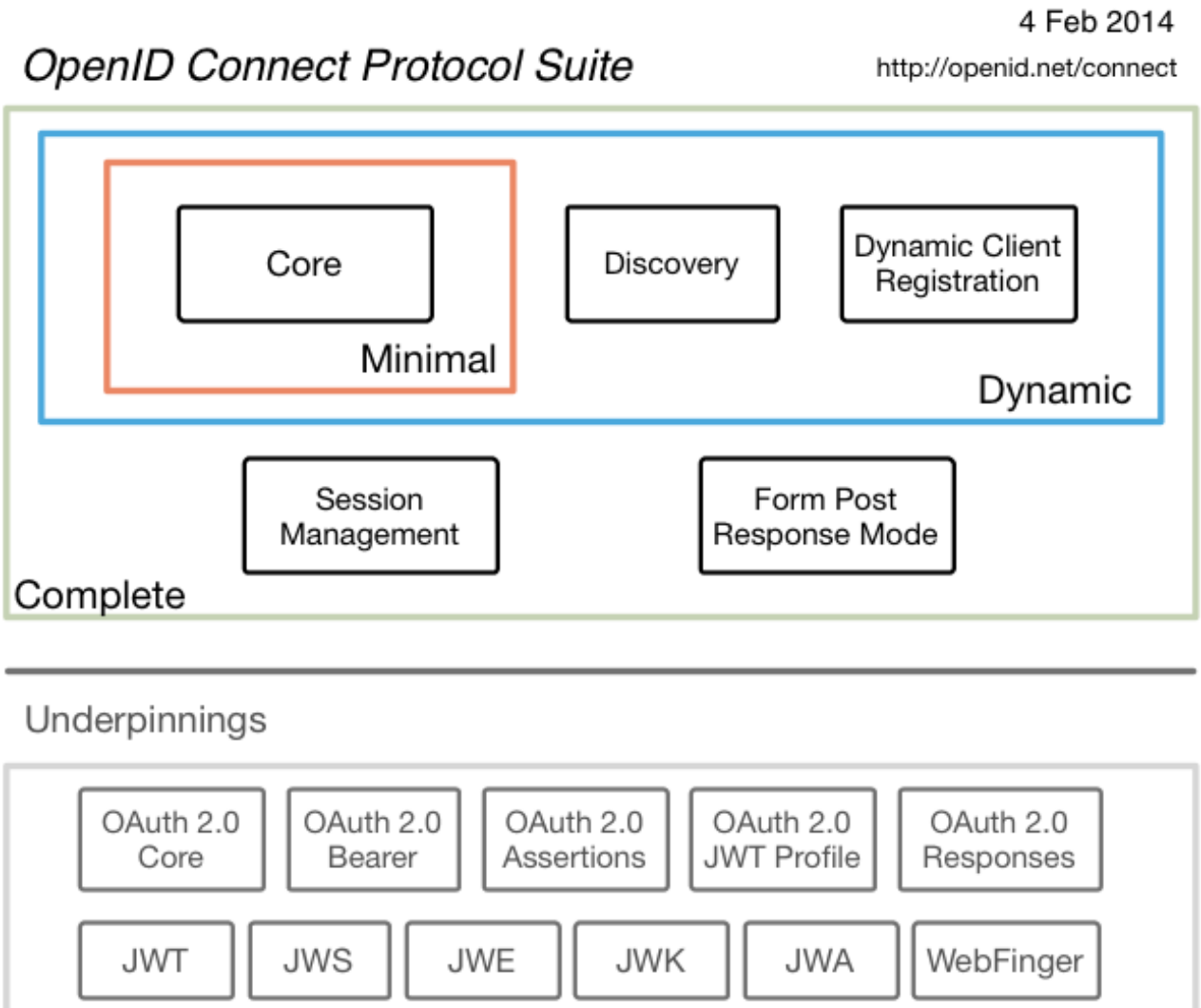
OpenID vs OAuth



OpenID Connect

- *OpenID Connect 1.0 is a simple identity layer on top of the OAuth 2.0 protocol.* It allows Clients to verify the identity of the End-User based on the authentication performed by an Authorization Server, as well as to obtain basic profile information about the End-User in an interoperable and REST-like manner
- In technical terms, OpenID Connect specifies a RESTful HTTP API, using JSON as a data format
- Based on OAuth 2.0, REST, JSON, JWT, JOSE

OpenID Connect



OpenID Connect

!!! not: openid idp isi nasil isliyor

OpenID provider (OP)

An identity provider, or OpenID provider (OP) is a service that specializes in registering OpenID URLs or XRIs. OpenID enables an end-user to communicate with a relying party

ID Token

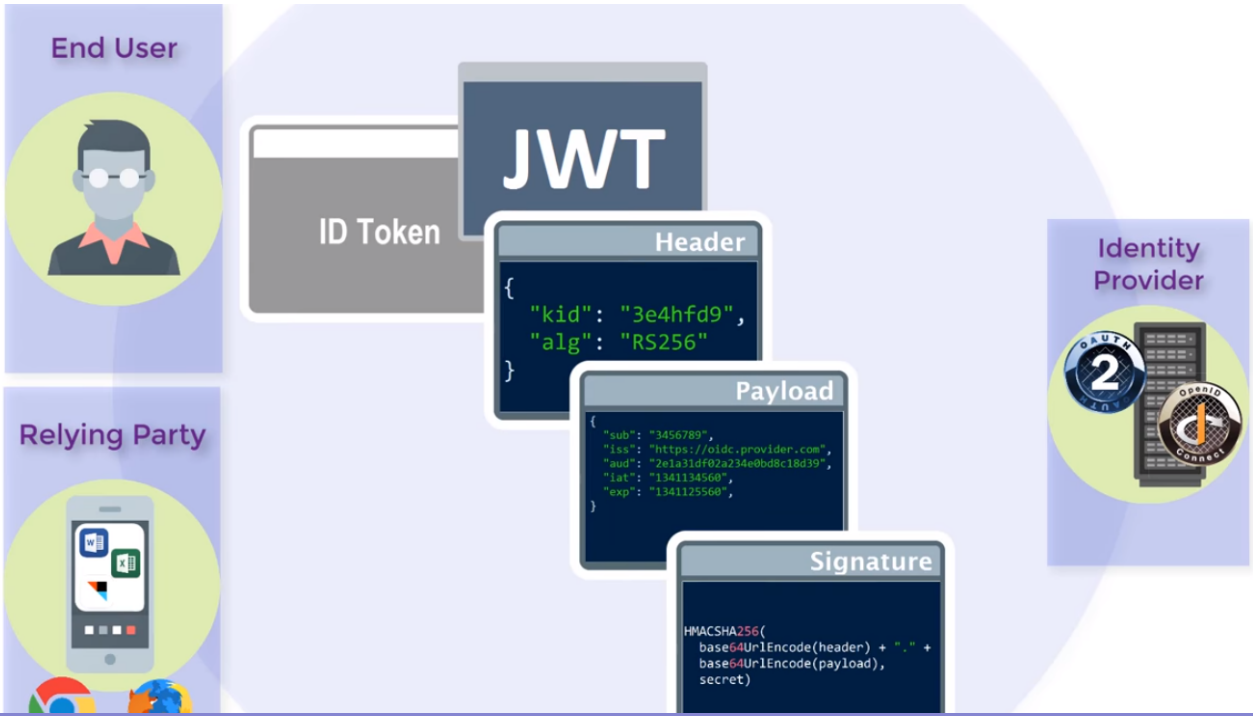
OpenID Connect

!!! not: videodan not al !!! id token contains a number of required attributes or clalims about that end user but also how end user was authenticated

JWT Token

OpenID Connect

!!! not: videodan not al !!! the claims in the token form part of payload !!! digitally signed using json web signature (integrity non reqpudiation) !!! header payload and signature are combined into a jwt and may also be encrypted with JWE



OpenID Connect

ID Token


```
{
    "iss": "https://self-issued.me",
    "nonce": "n-0S6_WzA2Mj",
    "exp": 1311281970,
    "iat": 1311280970,
    "sub_jwk" : {
        "crv":"secp256k1",
        "kid":"did:example:0xcd#verikey-1",
        "kty":"EC",
        "x":"7KEKZa5xJPh7WVqHJyUpb2MgEe3nA8Rk7eUlXsmBl-M",
        "y":"3zIgl_ml4RhapyEm5J7lvU-4f5jiBvZr4KgxUjEhl9o"
    },
    "sub": "9-aYUQ7mgL2SWQ_LNTeVN2rtw7xFP-3Y2EO9WV22cF0",
    "did": "did:example:0xcd"
}
```

OpenID Connect

Security

- Registration between RP and OP is mandatory, can be done with public metadata exchange and selfregistration
- JSON messages can be signed and/or encrypted with the help of asymmetric keys (public keys published in JWKS) or symmetric keys (client secret)

Single Sign On

!!! yukardaki protokollerin tek olayi sso degil fakat genellikle bu amacla kullaniliyorlar. !!! not: open id den alinan id tokenler birden fazlo uygulamada login olabilir, session acabilir yanit sso !!! not: cas zaten sso implementasyonu, samlin ana kullanim amaci zaten sso implement etmek

Single sign-on (SSO) is an authentication scheme that allows a user to log in with a single ID and password to any of several related, yet independent, software systems.

Web Single Sign On protocols

- Based on the principle of an authentication server, a lot of SSO standards have been created:
  - CoSign (Weblogin), Pubcookie, Webauth, CAH, CAS, WebID, BrowserID (Persona), SAML, WS-*, Liberty alliance, SAML 2, Shibboleth, OpenID, OpenID Connect...
- But nowadays, only a few are really used:
  - CAS, SAML 2, OpenID Connect

CAS

The Central Authentication Service (CAS) is a single *sign-on* protocol for the web.

Its purpose is to permit a user to access multiple applications while providing their credentials (such as userid and password) only once. It also allows web applications to authenticate users without gaining access to a user's security credentials, such as a password.

Security

- No obligation to declare CAS clients in CAS server (open mode)
- Trust between CAS client and CAS server relies on CAS server certificate validation

SAML

Security Assertion Markup Language is an open standard for exchanging authentication and authorization data between parties, in particular, between an identity provider and a service provider

- An important use case that SAML addresses is web-browser single sign-on

Security

- Registration between SP and IDP is mandatory, can be done with public metadata exchange
- XML messages can be signed and/or encrypted with the help of asymmetric keys (public keys published in metadata)

Security

- Registration between RP and OP is mandatory, can be done with public metadata exchange and selfregistration

Protocol Comparison

!!! not: yani open id connect sso protokolu olarak ta one cikiyor !!! not: id token is gorur

- CAS: simple protocol, no strong security, fits internal usage
- SAML: complex protocol, very used for SaaS authentication, good security, well established
- ***OpenID Connect:*** easy adoption with new technologies (JSON/REST/OAuth2), mobile ready, good security, still not wide spread

self-issued openid connect provider

!!! not: Self-Issued OpenID Connect Provider bunu acikla, protokolde ne gibi farkliliklara yol acabilir !!! not: SIOP OP tan ne farki var, OP internette duruken, SIOP sende localde duruyor.

- A normal provider such as Google, is available at an HTTP endpoint. Requests to normal providers use the http:// protocol.
- A self-issued provider is usually installed on the end-user's device. Requests to self-issued providers use the openid:// protocol.

self-issued openid connect provider DID Profile (did-siop, DIF)

The work on DIF SIOP DID Profile specification has moved to OIDF AB WG to work on a new SIOP v2 specification that will either introduce breaking changes to the DIF SIOP DID Profile specification or will replace it with an implementation guide document on how to use SIOP v2 in an SSI context.

DID SIOP Terminology

DID SIOP

Term	Description
DID	Decentralized Identifier as per [DID]
DID Document	DID Document as per [DID]
SIOP DID	Self-Issued OpenID Connect Provider DID profile. Refers to a specific flavor of DID AuthN used in the OIDC SIOP flow.
JWT	JSON Web Token as per [RFC7797]
JWE	JSON Web Encryption as per [RFC7516]
JWS	JSON Web Signature as per [RFC7515]
JWK	JSON Web Key as per [RFC7517]
JWKS	JWK Set as per [RFC7517]
OIDC	OpenID Connect as per [OIDC.Core]
OIDC client	Used synonymously with Relying Party (see RP)
OP	OpenID Provider as per [OIDC.Core]
SIOP	Self-Issued OpenID Provider as per [OIDC.Core]
RP	Relying Party, as used in [OIDC.Core]
Identity Wallet	An Identity Wallet refers to a application that is under the control and acts on behalf of the DID holder. This Also known as an identity agent. The Identity Wallet can have different form factors such as a mobile app, browser extension/ plugin etc.
DID AuthN	Refers to a method of proofing control over a DID for the purpose of authentication.

Introduction

DID SIOP

!!! not: While this specification focuses on the integration of Identity Wallets in the form of browser extensions/ plugins, or smartphone apps, it does not prevent implementers using the proposed flow in different scenarios as well, e.g., between two web services with pre-populated DIDs.
!!! not: cevir, nota ekle

An everyday use case that the Decentralized Identity community identified is the sign-up or login with web applications. Nowadays, this is often achieved through social login schemes such as Google Sign-In. *While the Decentralized Identity community has serious concerns about social login, the underlying protocol, OIDC, does not have these flaws by design.* SIOP DID provides great potential by leveraging an Identity Wallet, e.g., as a smartphone app, on the web. This will increase and preserve the user's privacy by preventing third-parties from having the ability to track which web applications a user is interacting with.

Introduction

DID SIOP

Purpose

The main purpose is to sign up with/ login to an RP (Relaying Party), i.e., web application. It assumes the user operates a mobile or desktop browser or a browser-based app that can respond to SIOP requests according to this specification.

Goals

- Staying backward compatible with existing OIDC clients and OPs (OpenID Provider) that implement the SIOP specification which is part of the OIDC core specification as per [OIDC.Core] to reach a broader community.
- Adding validation rules for OIDC clients that have DID AuthN support to make full use of DIDs.
- Not relying on any intermediary such as a traditional centralized public or private OP while still being OIDC-compliant.

Protocol Flow

DID SIOP

- First, the user clicks on the sign up or login UX element. The RP will then generate the redirect to openid:// which will be handled by the SIOP.
- The SIOP will generate the based on the specific DID method that is supported. The will be signed and optionally encrypted and will be provided according to the requested response mode.

Protocol Flow

DID SIOP

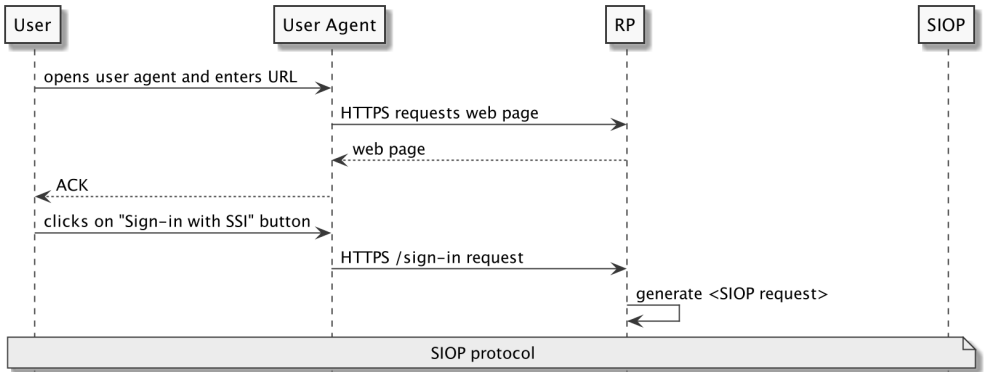
!!! not: detaylari anla not al !!! not: turkcelerini not al

- Unlike the OIDC Authorization Code Flow as per [OIDC.Core], the **SIOP will not return an access token to the RP**
- **SIOP also differs from Authorization Code Flow by not relying on a centralized and known OP**. The SIOP can be unknown to the RP until the user starts to interact with the RP using its Identity Wallet
- OIDC Authorization Code Flow is *still a useful approach* and *should be used whenever the OP is known, and OP discovery is possible, e.g.*, exchanged or pre-populated DID Document containing an openid element in the service section.
- *The SIOP flow allows to integrate Identity Wallets with plain OIDC clients if they implemented the SIOP specification*. In contrast, using DID AuthN as the authentication means in the OIDC Authorization Code Flow would *require integration with the OP vendor itself*.

Protocol Flow

DID SIOP

Example SIOP flow



Protocol Flow

DID SIOP

Example SIOP flow