Autentikacija pomoću protokola OpenID Connect (OIDC)

Uvod

Uz već podržane protokole kao što su SAML i CAS, sustav AAI@EduHr nudi mogućnost autenticiranja korisnika pomoću protokola OpenID Connect (OIDC). OIDC je dodatni sloj na protokol OAuth2, pa se često može vidjeti referenca na taj protokol kao OAuth2 / OIDC. Prvenstvena namjena protokola OIDC je autentikacija (authn), dok je za OAuth2 to autorizacija (authz). OIDC donosi dodatne elemente u autorizacijski tijek protokola OAuth2, čime se omogućuje postupak autentikacije. Na primjer, OIDC ima predefinirane opsege (eng. scopes) za definiranje korisničkih atributa koji će se isporučivati, te ID token kao dodatni tip tokena uz pristupni token (eng. access token) iz OAuth2. ID token ima određeni format (JWT: JSON web token), dok je format pristupnog tokena nedefiniran. Upravo je dodatak ID tokena u autorizacijski tijek protokola OAuth2 dio kojim se omogućuje standardizirani postupak autentikacije korisnika.

U ovim uputama nastoji se sažeto opisati informacije potrebne za uspješno obavljanje autentikacije, no u slučaju nedoumica uvijek je dobro referirati se na originalne specifikacije protokola. Slijedi lista specifikacija koje su na neki način povezane uz postupak autentikacije pomoću protokola OIDC (redoslijed ne predstavlja važnost):

- OpenID Connect Core 1.0
- OpenID Connect Basic Client Implementer's Guide 1.0
- OpenID Connect Discovery 1.0
- The OAuth 2.0 Authorization Framework
- OAuth 2.0 Security Best Current Practice
- Proof Key for Code Exchange by OAuth Public Clients
- OAuth 2.0 for Native Apps
- JSON Web Token (JWT)
- JSON Web Signature (JWS)
- JSON Web Key (JWK)
- JSON Web Algorithms (JWA)
- OpenID Connect RP-Initiated Logout 1.0
- OpenID Connect Back-Channel Logout 1.0

AAI@EduHr podržava sljedeće autentikacijske tijekove iz OIDC specifikacije: tijek autorizacijskog koda (eng. Authorization Code Flow) i implicitni tijek (eng. implicit flow), dok hibridni tijek (eng. hybrid flow) trenutno nije podržan. Uz to, podržan je i implicitni OAuth2 tijek.

Slijedeć specifikacije, a možebitno i ove upute, implementacija autentikacije pomoću protokola OIDC se može učiniti kompliciranom. No, treba imati na umu da je protokol OIDC prilično popularan i raširen, te da već postoji mnoštvo klijenata napisanih u različitim programskim jezicima koji implementaciju svode na podešavanje konfiguracije i pozivanje nekoliko metoda, a rezultat su informacije o autenticiranom korisniku. Da biste dobili dojam kako to može na kraju izgledati, svakako bacite pogled na primjere implementacije.

Registar resura

Prije nego vaša aplikacija (usluga, resurs) može početi koristiti protokol OIDC za autentikaciju korisnika u sustavu AAI@EduHr, potrebno je obaviti registraciju OIDC klijenta, a time i samog resursa, tj. vaše usluge, u AAI@EduHr Registru resursa. Detaljne upute o tome kako registrirati resurs dostupne su na stranici Registar resursa.

Registracijom resursa i OIDC klijenta definirat će se:

- općenite informacije o resursu, tj. usluzi
- OIDC klijentske vjerodajnice:
    - ID klijenta (eng. client ID) - niz znakova koji jedinstveno određuje klijenta
    - tajni ključ klijenta (eng. client secret) - tajni niz znakova kojeg klijent koristi tijekom autentikacije krajnjeg korisnika (ne smije biti javno dostupan)
- tip OIDC klijenta - naznaka je li klijent povjerljiv (eng. confidential) ili javan (eng. public). Povjerljiv klijent je onaj koji može sigurno čuvati klijentske vjerodajnice tj. tajni ključ klijenta. To su npr. web aplikacije koje imaju dio aplikacije koji se izvršava na poslužitelju (eng. backend), pa se klijentske vjerodajnice mogu čuvati tako da im samo razvijatelji aplikacije imaju pristup. Javni klijent je onaj koji ne može sigurno čuvati klijentske vjerodajnice tj. tajni ključ klijenta (engl. client secret). To su npr. web aplikacije koje se izvršavaju u web pregledniku (npr. JavaScript aplikacije), te sve nativne aplikacije (Windows, Linux, Android, iOS...), jer se pretpostavlja da postoji mogućnost izvlačenja klijentskih vjerodajnica iz takvih aplikacija.
- lokacija za preusmjeravanje (eng. redirect URI) - lokacija na koju će poslužitelj vratiti odgovor na zahtjev za autentikacijom
- opsezi (eng. scopes) - definicija skupa tvrdnji (eng. claims), tj. korisničkih atributa koji će se isporučivati nakon uspješne autentikacije
- opis razloga korištenja traženih opsega i pripadajućih tvrdnji (korisničkih atributa)
- lokacija za odjavu u pozadinskom kanalu (eng. Back-Channel Logout URI) - opcionalno; lokacija na koju će se poslat token za odjavu (eng. Logout Token) nakon iniciranja odjave na autentikacijskom poslužitelju. Unijeti ako OIDC klijent podržava odjavu u pozadinskom kanalu prema specifikaciji "OpenID Connect Back-Channel Logout 1.0".
- lokacije za preusmjeravanje nakon pokretanja odjave (eng. Post-logout redirect URIs) - opcionalno; dozvoljene lokacije na koje će biti moguće preusmjeriti web preglednik nakon pokretanja odjave. Unijeti ako će OIDC klijent koristiti mogućnost pokretanja odjave od pouzdanje strane prema specifikaciji "RP-Initiated Logout 1.0"
- dozvoljena ishodišta za javne klijente (eng. Allowed origins for public clients) - dozvoljena ishodišta za CORS zahtjeve za javne klijente koji se izvode u web pregledniku. Potrebno ih je unijeti za javne klijente koji se izvode u web pregledniku (tipično za JavaScript aplikacije).

Pod OIDC klijentom podrazumijeva se aplikacija, dio aplikacije, modul, biblioteka ili slično, koja podnosi zahtjeve za autentikacijom krajnjeg korisnika na autentikacijski poslužitelj i obrađuje odgovore iz kojih dohvaća korisničke podatke. Nakon što obavite registraciju resursa i OIDC klijenta, možete iskoristiti vrijednosti definirane u Registru resursa za konfiguriranje OIDC klijenta u vašem izvornom kodu (postavljenje ID klijenta, tajnog ključa, lokacije za preusmjeravanje, opsega...).

Pošto je u postupku registracije OIDC klijenta potrebno odabrati određene opsege kojima se definiraju korisnički atributi koji će se isporučivati, preporučamo da svakako proučite poglavlje "Opsezi i tvrdnje (eng. Scopes and Claims)" ovih uputa.

AAI@EduHr OIDC konfiguracija

AAI@EduHr OIDC konfiguracijski URL je: https://login.aaiedu.hr/.well-known/openid-configuration

Na tom URL-u je dostupan JSON objekt koji sadrži sljedeća svojstva:

- krajnje točke (eng. endpoints) koje se koriste za autentikaciju krajnjeg korisnika - svojstva: 'authorization_endpoint', 'token_endpoint' i 'userinfo_endpoint'
- URL na JSON web skup ključeva (eng. JSON Web Key Set - JWKS) pomoću kojih se može provjeravati potpis u ID tokenu - svojstvo 'jwks_uri'
- podržani opsezi - svojstvo 'scopes_supported'

- podržani autentikacijski tijekovi - svojstvo 'response_types_supported'
- podržani tipovi identifikatora subjekta (eng. subject identifier) - svojstvo 'subject_types_supported'
- podržani algoritmi za potpisivanje ID tokena - svojstvo 'id_token_signing_alg_values_supported'
- podržane metode za generiranje parametra 'code_challenge' - svojstvo 'code_challenge_methods_supported'
- krajnja točka za prekidanje sjednice - svojstvo 'end_session_endpoint'
- podržane metode autentikacije klijenta na token krajnjoj točki - svojstvo 'token_endpoint_auth_methods_supported'
- indikator podrške za parametar 'request' - svojstvo 'request_parameter_supported'
- podržani načini izdavanja tokena - svojstvo 'grant_types_supported'
- indikator podrške za parametar 'claims' - svojstvo 'claims_parameter_supported'
- indikator podrške za odjavu u pozadinskom kanalu - svojstvo 'backchannel_logout_supported'
- indikator podrške za tvrdnju 'sid' (ID sjednice) u tokenu za odjavu - svojstvo 'backchannel_logout_session_supported'

Primjer sadržaja AAI@EduHr konfiguracije (dio vrijednosti u svojstvu 'scopes_supported' maknut radi preglednosti):

AAI@EduHr OIDC konfiguracija

```
{
  "issuer": "https://login.aaiedu.hr",
  "authorization_endpoint": "https://login.aaiedu.hr/sso/module.php/oidc/authorize.php",
  "token_endpoint": "https://login.aaiedu.hr/sso/module.php/oidc/token.php",
  "userinfo_endpoint": "https://login.aaiedu.hr/sso/module.php/oidc/userinfo.php",
  "end_session_endpoint": "https://login.aaiedu.hr/sso/module.php/oidc/logout.php",
  "jwks_uri": "https://login.aaiedu.hr/sso/module.php/oidc/jwks.php",
  "scopes_supported": [
    "openid",
    "profile",
    "email",
    "address",
    "phone",
    "hrEduPersonUniqueID",
    "...",
    "hrEduPersonCardNum"
  ],
  "response_types_supported": [
    "code",
    "token",
    "id_token",
    "id_token token"
  ],
  "subject_types_supported": [
    "public"
  ],
  "id_token_signing_alg_values_supported": [
    "RS256"
  ],
  "code_challenge_methods_supported": [
    "plain",
    "S256"
  ],
  "token_endpoint_auth_methods_supported": [
    "client_secret_post",
    "client_secret_basic"
  ],
  "request_parameter_supported": false,
  "grant_types_supported": [
    "authorization_code",
    "refresh_token"
  ],
  "claims_parameter_supported": true,
  "backchannel_logout_supported": true,
  "backchannel_logout_session_supported": true
}
```

JSON web skup ključeva

Svojstvo 'jwks_uri' sadrži URL na kojem je dostupan JSON web skup ključeva (eng. JSON Web Key Set - JWKS). Ukratko, JWKS je JSON objekt koji pod svojstvom 'keys' sadrži polje ključeva, dakle jedan ili više ključeva formatu JWK (JSON Web Key - JWK). To su ključevi među kojima će se nalaziti i onaj javni ključ koji se može koristiti za provjeru potpisa u ID tokenu.

Svaki JWK sadrži nekoliko generalnih svojstva koji ga opisuju:

Svojstvo	Opis,,,,,	Napomena
kty	Tip ključa (eng. key type), npr. 'RSA' ili 'EC'.	
kid	ID ključa (eng. key ID). Jedinstveno određuje svaki ključ u skupu ključeva.	Može se koristiti za određivanje javnog ključa za provjeru potpisa u ID tokenu (ID token u zaglavlju ima naznačen ID javnog ključa). Također, u slučaju predmemoriranja (eng. caching) javnih ključeva na resursu, omogućuje jednostavniju zamjenu ključeva (eng. key rollover).
use	Namjena ključa. Definira je li namjena ključa enkripcija podataka (vrijednost 'enc') ili provjera potpisa (vrijednost 'sig').	
alg	Oznaka algoritma koji se koristi s ključem, npr. 'RS256' (SHA256 with RSA), 'ES256' (SHA256 with ECDSA)...	

Ostala svojstva ovise o korištenom tipu i algoritmu ključa.

Primjer sadržaja JWKS-a:

JWKS

```
{
  "keys": [
    {
      "kty": "RSA",
      "n": "py7oHZwX71g3azW6La-cLUi-...-DS3FyOJnyY8bZS3SQ",
      "e": "AQAB",
      "kid": "69d8c46574",
      "use": "sig",
      "alg": "RS256"
    }
  ]
}
```

U slučaju nemogućnosti korištenja ključeva u formatu JWK, javni ključ u formatu PEM dostupan je na poveznici: https://login.aaiedu.hr/sso/module.php/saml/idp/certs.php/idp.crt

Trajanje kodova i tokena

Trajanje kodova i tokena je kako slijedi:

- autorizacijski kod (eng. authorization code): 1 minuta
- pristupni token (eng. access token) i ID token: 1 sat
- token za osvježavanje (eng. refresh token): 8 sati

Testno okruženje AAI@EduHr Lab

Za testno okruženje OIDC klijent je potrebno podesiti prema uputama na stranici: **Upute za korištenje AAI@EduHr Laba**

Opsezi i tvrdnje (eng. Scopes and Claims)

OIDC opsezi (eng. scopes) se koriste tijekom autentikacijskog postupka za autorizaciju pristupa pojedinim korisničkim atributima. Svaki opseg vraća skup korisničkih atributa koje u kontekstu protokola OIDC nazivamo tvrdnjama (eng. claims). Dakle, ovisno o tome koje korisničke atribute aplikacija zahtjeva, potrebno je koristiti različite opsege (minimalan skup različitih opsega tj. potrebno je zahtijevati samo potrebne opsege). Nakon uspješne autentikacije tvrdnje će biti dostupne na krajnjoj točki za dohvat korisničkih podataka (eng. 'userinfo' endpoint), a u slučaju kada u autentikacijskom postupku nije izdan pristupni token koji bi se iskoristio za dohvat tvrdnji, tvrdnje će biti direktno vraćene u ID tokenu.

AAI@EduHr podržava korištenje standardnih OIDC opsega, uz napomenu da neće isporučiti sve tvrdnje koje su definirane OIDC standardom. Naime, u nekim slučajevima nije moguće napraviti direktno mapiranje AAI@EduHr korisničkog atributa na pojedinu OIDC tvrdnju zbog drugačijeg formata podatka kojeg koristi AAI@EduHr od onog kojeg specificira OIDC standard. Na primjer, OIDC tvrdnja 'birthdate' je u formatu ' YYYY-MM-DD', dok je AAI@EduHr korisnički atribut 'hrEduPersonDateOfBirth' u formatu 'YYYYMMDD', i slično.

U radu sa OIDC opsezima, tvrdnjama i AAI@EduHR korisničkim atributima korisno je imati sačuvane sljedeće poveznice:

- standardni OIDC opsezi: https://openid.net/specs/openid-connect-core-1_0.html#ScopeClaims
- standardne OIDC tvrdnje: https://openid.net/specs/openid-connect-core-1_0.html#StandardClaims
- AAI@EduHr shema atributa: https://www.aaiedu.hr/o-sustavu/imenicke-sheme/shema

Standardni OIDC opsezi

Standardni OIDC opsezi su:

- openid
- offline_access
- profile
- email
- address
- phone

Prema protokolu OIDC, opseg 'openid' mora uvijek biti prisutan u autentikacijskom tijeku. Opseg 'openid' naznačuje autorizacijskom / autentikacijskom poslužitelju da je zahtjev koji dolazi zapravo autentikacijski zahtjev (a ne autorizacijski iz protokola OAuth2), te da poslužitelj treba izdati ID token koji će sadržavati tvrdnje o korisniku. Opseg 'openid' ne definira koje tvrdnje o korisniku će se isporučiti.

Opseg 'offline_access' naznačuje autentikacijskom poslužitelju da treba izdati i token za osvježavanje, uz ostale tokene. Opseg 'offline_access' također ne definira koje tvrdnje o korisniku će se isporučiti.

Za definiranje koje tvrdnje o korisniku će se isporučiti potrebno je koristiti dodatne opsege, npr. standardne OIDC opsege 'profile', 'email', ili neki poseban AAI@EduHr opseg (ili više njih).

Mapiranje standardnih OIDC tvrdnji na AAI@EduHr korisničke atribute napravljeno po uzoru na REFEDS wiki članak https://wiki.refeds.org/display/GROUPS/Mapping+SAML+attributes+to+OIDC+Claims. Dodatno, u slučaju kada neki AAI@EduHr korisnički atribut može imati više vrijednosti, a prema OIDC specifikaciji odgovarajuća OIDC tvrdnja mora imati jednu vrijednost, onda se kao vrijednost te OIDC tvrdnje uzima prva vrijednost AAI@EduHr korisničkog atributa koju vrati LDAP imenik. Na primjer, prema AAI@EduHr shemi, korisnički atribut 'mail' može imati više vrijednosti. Odgovarajuća OIDC tvrdnja je 'email', a prema specifikaciji ona može imati jednu vrijednost tipa 'string'. U tom slučaju, za OIDC tvrdnju 'email' uzet će se prva vrijednost AAI@EduHr korisničkog atributa 'mail' koju vrati LDAP imenik. Pritom je potrebno imati na umu da LDAP speficikacija ne garantira redoslijed atributa (https://www.ietf.org/rfc/rfc2251.txt, poglavlje 4.1.8. Attribute).

Slijedi popis standardnih OIDC opsega sa pripadajućim tvrdnjama i AAI@EduHr korisničkih atributa čija će vrijednost predstavljati OIDC tvrdnju (mapa OIDC tvrdnji na AAI@EduHr korisničke atribute).

OIDC opseg 'profile'

OIDC tvrdnja	AAI@EduHr korisnički atribut	Napomena
name	cn	
family_name	sn	
given_name	givenName	
middle_name		Nije podržano.
nickname	displayName	
preferred_username	hrEduPersonUniqueID	
profile	labeledURI	
picture		Nije podržano.
website		Nije podržano.
gender		Nije podržano.
birthdate		Nije podržano.

zoneinfo		Nije podržano.
locale		Nije podržano.
updated_at		Nije podržano.

OIDC opseg 'email'

OIDC tvrdnja	AAI@EduHr korisnički atribut	Napomena
email	mail	
email_verified		Nije podržano.

OIDC opseg 'address'

OIDC tvrdnja	AAI@EduHr korisnički atributi	Napomena
address	postalAddress, street, l, postalCode	Tvrdnja 'address' je JSON objekt. Mapa podržanih svojstava u objektu je kako slijedi (OIDC svojstvo u tvrdnji 'address' - AAI@EduHr korisnički atribut): • 'formatted' - 'postalAddress', • 'street_address' - 'street', • 'locality' - 'l', • 'postal_code' - 'postalCode'

OIDC opseg 'phone'

OIDC tvrdnja	AAI@EduHr korisnički atribut	Napomena
phone_number	mobile, telephoneNumber	Ako je dostupan atribut 'mobile', koristi se njegova vrijednost. Inače, ako je dostupan atribut 'telephoneNumber', koristi se njegova vrijednost. OIDC preporučuje format E.164, dok AAI@EduHR koristi format E.123. Tvrdnja se ipak isporučuje, jer se radi o preporučenom formatu, a ne zahtijevanom.
phone_number_verified		Nije podržano.

AAI@EduHr opsezi i tvrdnje

Kako bi se omogućila isporuka svih AAI@EduHr korisničkih atributa, uz standardne definirani su i posebni AAI@EduHr opsezi i tvrdnje. Uz to, AAI@EduHr opsezi omogućuju i isporuku svih vrijednosti onih atributa koji prema AAI@EduHr shemi mogu imati višestruke vrijednosti.

AAI@EduHr OIDC opsezi su definirani tako da svaki opseg sadrži jednu tvrdnju. Naziv opsega odgovara nazivu tvrdnje koju sadrži, a naziv tvrdnje odgovara nazivu AAI@EduHr korisničkog atributa kojeg tvrdnja predstavlja. Svaka AAI@EduHr tvrdnja će biti u obliku JSON polja (eng. array), neovisno o tome može li AAI@EduHr korisnički atribut kojeg predstavlja imati višestruke vrijednosti ili ne.

Slijedi popis AAI@EduHr opsega, tvrdnji te pripadajućih AAI@EduHr korisničkih atributa.

AAI@EduHr opseg	AAI@EduHr tvrdnja	AAI@EduHr korisnički atribut
hrEduPersonUniqueID	hrEduPersonUniqueID	hrEduPersonUniqueID
uid	uid	uid
cn	cn	cn
sn	sn	sn
givenName	givenName	givenName
mail	mail	mail

telephoneNumber	telephoneNumber	telephoneNumber
hrEduPersonExtensionNumber	hrEduPersonExtensionNumber	hrEduPersonExtensionNumber
mobile	mobile	mobile
facsimileTelephoneNumber	facsimileTelephoneNumber	facsimileTelephoneNumber
hrEduPersonUniqueNumber	hrEduPersonUniqueNumber	hrEduPersonUniqueNumber
hrEduPersonOIB	hrEduPersonOIB	hrEduPersonOIB
hrEduPersonDateOfBirth	hrEduPersonDateOfBirth	hrEduPersonDateOfBirth
hrEduPersonGender	hrEduPersonGender	hrEduPersonGender
jpegPhoto	jpegPhoto	jpegPhoto
userCertificate	userCertificate	userCertificate
labeledURI	labeledURI	labeledURI
hrEduPersonProfessionalStatus	hrEduPersonProfessionalStatus	hrEduPersonProfessionalStatus
hrEduPersonAcademicStatus	hrEduPersonAcademicStatus	hrEduPersonAcademicStatus
hrEduPersonScienceArea	hrEduPersonScienceArea	hrEduPersonScienceArea
hrEduPersonAffiliation	hrEduPersonAffiliation	hrEduPersonAffiliation
hrEduPersonPrimaryAffiliation	hrEduPersonPrimaryAffiliation	hrEduPersonPrimaryAffiliation
hrEduPersonStudentCategory	hrEduPersonStudentCategory	hrEduPersonStudentCategory
hrEduPersonExpireDate	hrEduPersonExpireDate	hrEduPersonExpireDate
hrEduPersonTitle	hrEduPersonTitle	hrEduPersonTitle
hrEduPersonRole	hrEduPersonRole	hrEduPersonRole
hrEduPersonStaffCategory	hrEduPersonStaffCategory	hrEduPersonStaffCategory
hrEduPersonGroupMember	hrEduPersonGroupMember	hrEduPersonGroupMember
o	o	o
hrEduPersonHomeOrg	hrEduPersonHomeOrg	hrEduPersonHomeOrg
ou	ou	ou
roomNumber	roomNumber	roomNumber
postalAddress	postalAddress	postalAddress
l	l	l
postalCode	postalCode	postalCode
street	street	street
homePostalAddress	homePostalAddress	homePostalAddress
homeTelephoneNumber	homeTelephoneNumber	homeTelephoneNumber
hrEduPersonCommURI	hrEduPersonCommURI	hrEduPersonCommURI
hrEduPersonPrivacy	hrEduPersonPrivacy	hrEduPersonPrivacy
hrEduPersonPersistentID	hrEduPersonPersistentID	hrEduPersonPersistentID
displayName	displayName	displayName
schacUserPresenceID	schacUserPresenceID	schacUserPresenceID

hrEduPersonCardNum	hrEduPersonCardNum	hrEduPersonCardNum

Stalne tvrdnje

ID token će sadržavati sljedeće, stalne tvrdnje koje opisuju autentikacijski događaj:

Tvrdnja	Opis	Napomena
iss	Identifikator izdavatelja (eng. issuer) ID tokena. Sadrži vrijednost AAI@EduHr identifikatora izdavatelja koji je inače dostupan na OIDC konfiguracijskom URL-u.	
sub	Identifikator subjekta (eng. subject identifier). Sadrži jedinstveni identifikator krajnjeg korisnika tj. korisnika koji se autenticirao.	Sadrži vrijednost AAI@EduHr korisničkog atributa 'hrEduPersonPersistentID'.
aud	Publika (eng. audience) kojoj je ID token namijenjen. Sadrži ID klijenta kojem je ID token namijenjen.	Sadrži jednu vrijednost tipa 'string'. OIDC specifikacija inače predviđa mogućnost vraćanja više vrijednosti u polju (više ID klijenata) ili jednostruku vrijednost (jedan ID klijenta).
jti	JWT ID, jedinstveni identifikator samog ID tokena. Može se koristiti za sprječavanje ponovnog korištenja već iskorištenog ID tokena.	
iat	Tvrdnja 'izdan u' (eng. issued at). Sadrži vremensku oznaku** kada je ID token izdan.	
exp	Vrijeme isteka (eng. expiration time). Sadrži vremensku oznaku** nakon koje ID token ne smije biti prihvaćen.	
nbf	Tvrdnja 'ne prije' (eng. not before). Sadrži vremensku oznaku** (eng. timestamp) prije koje ID token ne smije biti prihvaćen.	
nonce	Vrijednost tipa 'string' koja se koristi za povezivanje sjednice (eng. session) klijenta s ID tokenom, u svrhu sprječavanja napada ponovne reprodukcije (eng. replay attack). Vrijednost parametra 'nonce' kojeg klijent koristi tijekom autentikacije krajnjeg korisnika će biti proslijeđena u ID token. Nakon što klijent dobije ID token, mora provjeriti je li vrijednost 'nonce' u ID tokenu ista kao i ona korištena tijekom autentikacije.	
sid	Identifikator korisničke sjednice tipa 'string'.	

** vrijednost tvrdnji 'iat', 'exp' i 'nbf' je broj sekundi proteklih (ili koje će proteći) od 1. siječnja 1970. godine u 0 sati mjereno po standardu UTC (1970-01-01T0:0:0Z), do predmetne vremenske oznake.

ID token

ID token je primarni način isporuke tvrdnji o autentikacijskom događaju od autentikacijskog poslužitelja do resursa, u formatu JWT (JSON Web Token).

Ukratko, ID token se sastoji od tri dijela:

- zaglavlje (eng. header)
- korisni podaci (eng. payload)
- potpis (eng. signature)

Zaglavlje i korisni podaci su Base64 URL kodirani JSON objekti (svaki zasebno), a treći dio je Base64 URL kodirani potpis generiran nad kodiranim zaglavljem i korisnim podacima koristeći privatni ključ i algoritam naznačen u zaglavlju. Svaki dio tokena je odvojen točkom, pa je krajnji oblik ID tokena niz znakova u formatu:

```
zzzzz.kkkkk.ppppp
```

Dio 'zzzzz' predstavlja zaglavlje, dio 'kkkkk' predstavlja korisne podatke, a dio 'ppppp' predstavlja potpis. Da bi se došlo do podataka, ID token se može rastaviti na tri dijela preko točke '.', pa Base64 URL dekodirati zaglavlje i korisne podatke. Potpis je potrebno iskoristiti kako bi se provjerilo da ID token tijekom puta nije promijenjen, te da je izdan od očekivanog izdavatelja.

Primjer potpunog ID tokena:

ID token

```
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImtpZCI6IjY5ZDhjNDY1NzQifQ.
eyJpc3MiOiJodHRwczovL2xvZ2luLmFhaWVkdS5ociIsImF1ZCI6IjZlNTUyOTUyMDk3ODJiN2IyIiwianRpIjoiYzQ0ZjRjZmZjYzg0Zjc5O
TBmN2ExZDViMmMiLCJuYmYiOjE2MDI2NzQ0NzAsImV4cCI6MTYwMjY3NTA3MCwic3ViIjoiYmZhMTYwNWJlNDRhNTBhN2MiLCJpYXQiOjE2MD
I2NzQ0NzAsInNpZCI6IjUydG45NW1lQkw1NzgyODJIN1YiLCJub25jZSI6ImR0bm1lQkw1SFZuaFFrSVIifQ.KRV5qrLOg-
xVJdmtFFbsZZt--ZkeqJ98_K8jLrDh_r_Jy4CApeIBiSyQNTusGtk6fdpVFrOP_KmpSSvF8On5_T3lDuxr7VmXlU-
HgL_Gq23mlZeeIUqCW681H1znaskD8-4Odtu2MkYoRerH3NfZEl9ysf1c0Ur-
7mn5S3hfytoYyclOFHEemjBFSKmiuW6_n4M0NCuw2hsbojj4lIu6qmV1UXBAHJGzJy0WTAbmyt8oMJG_rPRCgfcQlGnca_NnTiQzLP-
4OPhMUunt5v4WZUzS8KfR49iQfwBFUilKLwCMrrh4leboDXWlJeJH1HMKabmGgoQne6nuGXp-7LY3eg
```

Zaglavlje

Zaglavlje ID tokena sadrži sljedeće tvrdnje.

Tvrdnja	Opis	Napomena
typ	Tip tokena.	Vrijednost je 'JWT'.
alg	Algoritam korišten za potpisivanje tokena.	Vrijednost je jedna od vrijednosti JSON svojstva ' id_token_signing_alg_values_supported ' na OIDC konfiguracijskom URL-u.
kid	ID ključa (eng. key ID) koji je korišten za potpisivanje tokena.	Vrijednost je jednaka vrijednosti svojstva 'kid' od jednog javnog ključa dostupnom u JSON web skupu ključeva (JSON web skup ključeva dostupan je na URL-u navedenom u JSON svojstvu 'jwks_uri' na OIDC konfiguracijskom URL-u).

Primjer Base64 URL dekodiranog zaglavlja ID tokena:

Zaglavlje ID tokena

```
{
  "typ": "JWT",
  "alg": "RS256",
  "kid": "69d8c46574"
}
```

Korisni podaci

Drugi dio ID tokena sadrži korisne podatke, tj. tvrdnje o autentikacijskom događaju. Uz stalne tvrdnje ('iss', 'sub', ...), u slučaju kada tijekom autentikacijskog postupka nije izdan pristupni token (npr. u implicitnom tijeku u verziji kada se izdaje samo ID token), ovaj JSON objekt može sadržavati i korisničke podatke (ako se koriste opsezi koji definiraju isporuku korisničkih podataka).

Primjer Base64 URL dekodiranih korisnih podataka:

Korisni podaci u ID tokenu

```
{
  "iss": "https://login.aaiedu.hr",
  "aud": "6e55295209782b7b2",
  "jti": "c44f4cffcc84f7990f7a1d5b2c",
  "nbf": 1602674470,
  "exp": 1602675070,
  "sub": "bfa1605be44a50a7c",
  "iat": 1602674470,
  "sid": "52tn95meBL578282H7V",
  "nonce": "dtnmeBL5HVnhQkIR"
}
```

Potpis

Potpis je treći (zadnji) dio ID tokena i generiran je nad Base64 URL kodiranim zaglavljem i Base 64 URL kodiranim korisnim podacima, povezano točkom '.' (nad prvim i drugim dijelom ID tokena); dakle nad nizom znakova 'zzzzz.kkkkk', koristeći algoritam naznačen u zaglavlju ID tokena.

AAI@EduHr za generiranje potpisa koristi algoritam 'RS256'. U postupku kreiranja potpisa koristi se privatni ključ iz para javnog i privatnog RSA ključa. Za provjeru tog potpisa potrebno je iskoristiti javni ključ koji je dostupan u JSON web skupu ključeva. Odgovarajući javni ključ se može pronaći preko ID ključa ('kid') koji je naznačen u zaglavlju ID tokena te u pojedinom ključu u JSON web skupu ključeva.

Primjer Base64 URL kodiranog potpisa:

Potpis ID tokena

```
c5q4W3bmcxQepAHj9n7xxj8WaH0wqIaiFkOFB9UE3joeVUyU9hWuNsDfPJ_BZ7WfYnhyMrv29Ys-
dHs1oKqagvqFdk157IEwLnW1Dd5vNXvgGhMBhCTtseeQBDFS_DHN6KaksFDnGtFyWh3GXq-dWEBO86fpgSB3OuV9CD-
AQAAbjXsN4Mz9MyaajMkhxWgxh_7HTZMlg5hACv3Xn-wiI8N3IGDqWrgdB87Vo6n0T5TfVaUZQ8mw5ca5fKy-
aH5BP940LBb3bt6CvhZqxK0XFmD_M8hh-2MyRuPf2u6_P9HUORo1f8x8ZI30J2arhghPRcqEK9Sv_hdPxMy7WJJupg
```

Autentikacijski tijekovi

Podržani autentikacijski tijekovi (eng. flows) su:

- Tijek autorizacijskog koda (eng. Authorization Code Flow)
- Implicitni tijek (eng. Implicit Flow)

Uz to, podržan je i tijek osvježavanja tokena (eng. refresh token flow) koji omogućuje ponovno izdavanje pristupnog tokena i ID tokena bez ponovne autentikacije krajnjeg korisnika.

Svaki autentikacijski tijek se obavlja pomoću niza HTTP zahtjeva i preusmjeravanja (eng. redirections) između aplikacije (resursa) i autentikacijskog poslužitelja. Različiti autentikacijski tijekovi znače različit način slanja HTTP zahtjeva i različit način dohvata korisničkih podataka nakon autentikacije. Također, različiti tijekovi donose i različite razine sigurnosti autentikacijskog postupka. Na primjer, od trenutno podržana dva tijeka, tijek autorizacijskog koda ima veću razinu sigurnosti od implicitnog tijeka.

AAI@EduHr preporučuje korištenje tijeka autorizacijskog koda za autentikaciju korisnika, neovisno o tome koji tip klijenta se koristi ili u kojem programskom jeziku je klijent implementiran.

Krajnje točke na autentikacijskom poslužitelju koje se mogu koristiti su:

- autorizacijska krajnja točka (eng. authorization endpoint) - URL je u svojstvu 'authorization_endpoint' u JSON objektu na OIDC konfiguracijskom URL-u
- token krajnja točka (eng. token endpoint) - URL je u svojstvu 'token_endpoint' u JSON objektu na OIDC konfiguracijskom URL-u
- krajnja točka za korisničke podatke (eng. userinfo endpoint) - URL je u svojstvu 'userinfo_endpoint' u JSON objektu na OIDC konfiguracijskom URL-u

Ovisno o tome koji autentikacijski tijek se koristi, koristit će se različite krajnje točke i različiti HTTP zahtjevi. Autentikacijski tijek se određuje tako da se u početnom HTTP zahtjevu na autorizacijsku krajnju točku postavi parametar 'response_type' s jednom od vrijednosti:

- 'code' - definira korištenje tijeka autorizacijskog koda
- 'id_token token' ili 'id_token' - definira korištenje implicitnog tijeka
- 'token' - definira korištenje implicitnog OAuth2 tijeka

Autorizacijska i token krajnja točka se koriste u postupku autentikacije tj. za izdavanje ID tokena i / ili pristupnog tokena. Krajnja točka za korisničke po datke se koristi nakon što je autentikacija obavljena, a služi za dohvat korisničkih podataka pomoću pristupnog tokena. Slijedi više informacija o svakom autentikacijskom tijeku.

Podržane vrijednosti parametra 'response_type' su navedene u svojstvu 'response_types_supported' u JSON objektu na OIDC konfiguracijskom URL-u. Time se definiraju autentikacijski tijekovi koje podržava AAI@EduHr. Inače, protokol OIDC definira i sljedeće vrijednosti parametra 'response_type' tj. dodatne autentikacijske tijekove koje AAI@EduHr trenutno ne podržava:

- code id_token - hibridni tijek
- code token - hibridni tijek
- code id_token token - hibridni tijek
- none

Tijek autorizacijskog koda

Tijek autorizacijskog koda je glavni i preporučeni način autenticiranja krajnjih korisnika koristeći protokol OIDC. Tijek autorizacijskog koda u protokolu OIDC je nadogradnja postojećeg tijeka autorizacijskog koda u protokolu OAuth2. Glavne promjene koje u ovaj tijek donosi protokol OIDC u odnosu na OAuth2 su preddefinirani OIDC opsezi i mogućnost izdavanja ID tokena.

Detaljni opis tijeka dostupan je u specifikacijama:

- OIDC: https://openid.net/specs/openid-connect-core-1_0.html#CodeFlowAuth
- OAuth2: https://tools.ietf.org/html/rfc6749#section-4.1

Ukratko, tijek autorizacijskog koda se obavlja na način:

- klijent napravi autorizacijski HTTP zahtjev preusmjeravanjem web preglednika na autorizacijsku krajnju točku
- krajnji korisnik se autenticira
- autentikacijski poslužitelj preusmjeravanjem web preglednika na registriranu lokaciju za preusmjeravanje (eng. redirect URI) šalje klijentu kratkotrajni (eng. short-lived) autorizacijski kod (kao GET parametar 'code')
- klijent u pozadini (bez preusmjeravanja web preglednika) napravi HTTP zahtjev na token krajnju točku koristeći dobiveni autorizacijski kod, a u HTTP odgovoru dobije pristupni token (eng. access token), te ID token ako je korišten opseg 'openid' (što je preporučeno)

Dakle, klijent dobije autorizacijski kod kojeg onda u pozadinskom kanalu (eng. back channel) direktno zamjeni za pristupni token i ID token. To znači da pristupni token i ID token neće biti izloženi u tkz. prednjem kanalu (eng. front channel), dakle korisničkom agentu (eng. user-agent) kao što je web preglednik (a time i svim možebitnim malicioznim aplikacijama koje imaju pristup korisničkom agentu). Također, autorizacijski poslužitelj može autenticirati klijenta prije izdavanja tokena.

Slijedi opis svih HTTP zahtjeva.

1. HTTP zahtjev na autorizacijsku krajnju točku

> Prema protokolu OIDC, ovaj HTTP zahtjev može biti tipa GET ili POST. AAI@EduHr trenutno podržava samo tip GET.

Autentikacijski tijek počinje HTTP zahtjevom na autorizacijsku krajnju točku (eng. authorization endpoint). URL autorizacijske krajnje točke je dostupan u svojstvu 'authorization_endpoint' u JSON objektu na OIDC konfiguracijskom URL-u. Ovaj HTTP zahtjev se obavlja preusmjeravanjem korisničkog agenta (web preglednika) na autentikacijski poslužitelj preko unaprijed pripremljenog URL-a s točno definiranim GET parametrima. Primjer pripremljenog URL-a za preusmjeravanje korisničkog agenta (razlomljeno na više redova radi preglednosti):

```
https://login.aaiedu.hr/sso/module.php/oidc/authorize.php?
response_type=code&
client_id=neki-id-klijenta&
redirect_uri=https://neki-uri-za-preusmjeravanje.primjer.hr&
scope=openid profile&
state=neki-state-132&
nonce=neki-nonce-123
```

Ovisno o tome koji tip klijenta se koristi, URL mora sadržavati određene GET parametre. Slijedi opis obaveznih i opcionalnih GET parametra:

Parametar	Vrijednost	Napomena
response_type	'code'	Obavezan parametar. Vrijednost 'code' signalizira da se radi o tijeku autorizacijskog koda.
client_id	ID klijenta koji se dobije prilikom registracije klijenta.	Obavezan parametar.
redirect_uri	Jedna od vrijednosti lokacija za preusmjeravanje koje su definirane prilikom registracije klijenta.	Obavezan parametar.
scope	Lista standardnih ili AAI@EduHr opsega odvojenih razmakom. Moguće vrijednosti opsega su navedene u JSON svojstvu 'scopes_supported' na OIDC konfiguracijskom URL-u.	Obavezan parametar. Za obavljanje autentikacije po protokolu OIDC, jedan od opsega mora biti 'openid'. Ako opseg 'openid' nije naveden, obavit će se tijek autorizacijskog koda po protokolu OAuth2.

code_challenge_method	Jedna od sljedećih vrijednosti: 'plain', 'S256'.	Obavezan parametar za javne (eng. public) klijente. Kada je klijent sposoban koristiti metodu 'S256', mora ju koristiti umjesto metode 'plain'.
code_challenge	Ako parametar 'code_challenge_method' koristi vrijednost 'plain', onda je vrijednost parametra 'code_challenge' jednaka vrijednosti parametra 'code_verifier'. Ako parametar 'code_challenge_method' koristi vrijednost 'S256', onda je vrijednost parametra 'code_challenge' jednaka vrijednosti koja se dobije SHA256 raspršenjem (eng. hash) vrijednosti parametra 'code_verifier'.	Obavezan parametar za javne (eng. public) klijente. Za generiranje vrijednosti parametra 'code_challenge' koristi se parametar 'code_verifier', koji mora biti unaprijed pripremljen. Parametar 'code_verifier' se ne koristi u ovom zahtjevu, nego u idućem zahtjevu na token krajnju točku.
state	Ako se koristi, vrijednost mora biti niz znakova s dovoljnom entropijom, dakle kriptografski siguran (ne smije ga biti lako pogoditi). Preporučuje se napraviti npr. raspršivanje (eng. hash) kolačića sjednice (eng. session cookie), ili raspršivanje kriptografski slučajnog niza znakova (eng. random string) po svakom zahtjevu.	Opcionalan parametar, ali je preporučen zbog zaštite od napada krivotvorenja zahtjeva za više web lokacija (eng. Cross-Site Request Forgery - CSRF). Vrijednost parametra koja je dana u ovom zahtjevu će biti vraćena kao jedan od parametara kod preusmjeravanja na registriranu lokaciju za preusmjeravanje, pa ju je tamo potrebno i provjeriti (mora biti ista). Tako se osigurava da na lokaciji za preusmjeravanje stignu HTTP zahtjevi koje je klijent tražio (dakle, samo odgovori na autorizacijski zahtjev). Parametar 'state' se ponekad koristi i za postavljanje URL-a na kojeg je krajnjeg korisnika potrebno preusmjeriti nakon procesa autentikacije (implementirati po potrebi, uz napomenu da bi URL za preusmjeravanje također trebao sadržavati slučajan niz znakova, npr. kao GET parametar, kako bi se očuvao smisao korištenja parametra).
nonce	Ako se koristi, vrijednost mora biti niz znakova s dovoljnom entropijom, dakle kriptografski siguran (ne smije ga biti lako pogoditi). Preporučuje se napraviti npr. raspršivanje (eng. hash) kolačića sjednice (eng. session cookie), ili raspršivanje kriptografski slučajnog niza znakova (eng. random string) po svakom zahtjevu.	Opcionalan parametar, ali je preporučen zbog zaštite od napada ponovne reprodukcije (eng. replay attack). Vrijednost parametra koja je dana u ovom zahtjevu će biti vraćena kao jedna od tvrdnji u ID tokenu, pa ju je tamo potrebno i provjeriti (mora biti ista). Tako se povezuje autorizacijski zahtjev s izdanim ID tokenom, pa klijent može potvrditi da je to ID token koji je zatražen.
prompt	Trenutno je jedina podržana vrijednost 'login'. OIDC propisuje i sljedeće vrijednosti koje AAI@EduHr trenutno ne podržava: 'none', 'consent', 'select_account'.	Opcionalan parametar. Ako je dana vrijednost 'login', autentikacijski poslužitalj će ponovno autenticirati korisnika čak i kada postoji aktivna SSO sjednica (forsira se ponovna autentikacija korisnika).
claims	Prema specifikaciji: https://openid.net/specs/openid-connect-core-1_0.html#rfc.section.5.5	Opcionalan parametar koji omogućuje zahtjevanje specifičnoh tvrdnji koje će se vratiti sa krajnje točke za korisničke podatke i / ili u ID tokenu.

U nastavku je opis HTTP GET zahtjeva kojeg korisnički agent obavi nakon što ga se preusmjeri na pripremljeni URL:

```
GET {Authorization Endpoint}
  ?response_type=code             // Obavezno
  &client_id={Client ID}          // Obavezno
  &redirect_uri={Redirect URI}    // Obavezno
  &scope={Scopes}                 // Obavezno
  &state={Arbitrary String}       // Preporueno zbog zaštite od napada krivotvorenja zahtjeva za više web
lokacija (eng. Cross-Site Request Forgery - CSRF)
  &code_challenge={Challenge}     // Obavezno za javne (eng. public) klijente, za povjerljive (engl.
confidential) klijente se ne koristi
  &code_challenge_method={Method} // Obavezno za javne (eng. public) klijente, za povjerljive (engl.
confidential) klijente se ne koristi
  &nonce={Nonce}                  // Preporueno zbog zaštite od napada ponovne reprodukcije (eng. replay
attack)
  HTTP/1.1
HOST: {Authorization Server}
```

Parametri 'code_verifier', 'code_challenge' i 'code_challenge_method' su dio dodatnog standarda 'Proof Key for Code Exchange by OAuth Public Clients - PKCE' kojim se želi poboljšati sigurnost za javne klijente. Upravo zbog tog standarda, javnim klijentima se omogućuje i preporučuje korištenje tijeka autorizacijskog koda za autentikaciju, umjesto da se koristi implicitni tijek. Vrijednosti koje mogu poprimiti parametri 'code_verifier', 'code_challenge' i 'code_challenge_method' detaljno su opisan su u specifikaciji PKCE u poglavlju https://tools.ietf.org/html/rfc7636#section-4.

2. HTTP odgovor sa autorizacijske krajnje točke

Nakon što se krajnji korisnik autenticira, autentikacijski poslužitelj će generirati kratkotrajni autorizacijski kod i vratit ga klijentu preusmjeravanjem na lokaciju za preusmjeravanje koja je korištena u prvom zahtjevu na autorizacijsku krajnju točku. Također, ako je korišten parametar 'state', vratit će i njega. Vrijednost parametra 'state' je potrebno provjeriti na klijentskoj strani (mora biti isti onaj koji je korišten u prvom zahtjevu prema autorizacijskoj krajnjoj točki).

```
HTTP/1.1 302 Found
Location: {Redirect URI}
  ?code={Authorization Code}  // Uvijek ukljuen
  &state={Arbitrary String}   // Ukljuen ako je zahtjev na autorizacijsku krajnju toku imao parametar 'state'
```

3. HTTP zahtjev na token krajnju točku

Ovaj zahtjev odrađuje se u pozadinskom kanalu (eng. back channel) na način da klijent samostalno šalje HTTP POST zahtjev na token krajnju točku (ne koristi se preusmjeravanje web preglednika). URL token krajnje točke je dostupan u svojstvu 'token_endpoint' u JSON objektu na OIDC konfiguracijskom URL-u.

```
POST {Token Endpoint} HTTP/1.1
Host: {Authorization Server}
Content-Type: application/x-www-form-urlencoded

grant_type=authorization_code  // Obavezno
&client_id={Client ID}         // Obavezno
&client_secret={Client secret} // Obavezno za povjerljive (eng. confidential) klijente. Za javne (eng.
public) klijente se ne postavlja.
&code={Authorization Code}     // Obavezno
&redirect_uri={Redirect URI}   // Obavezno
&code_verifier={Verifier}      // Obavezno za javne (eng. public) klijente. Za povjerljive (eng.
confidential) klijente se ne postavlja.
```

Opis parametara:

Parametar	Vrijednost	Napomena
grant_type	'authorization_code'	Vrijednost 'authorization_code' signalizira da se koristi autorizacijski kod u svrhu izdavanja tokena.
client_id	ID klijenta koji se dobije prilikom registracije klijenta.	
client_secret	Tajni ključ klijenta koji se definira prilikom registracije klijenta.	Koristi se samo za povjerljive klijente (ne postavljati ako je klijent javan).
code	Autorizacijski kod dobiven od autentikacijskog poslužitelja.	
redirect_uri	Jedna od vrijednosti lokacija za preusmjeravanje koje su definirane prilikom registracije klijenta.	Vrijednost mora biti ista onoj koja je korištena u autorizacijskom zahtjevu.
code_verifier	Ako se koristi, vrijednost mora biti Base64 URL kodiran niz znakova minimalne duljine 43 i maksimalne duljine 128. Niz znakova mora biti s dovoljnom entropijom, dakle kriptografski siguran (ne smije ga biti lako pogoditi).	Obavezan za javne klijente. Parametar 'code_verifier' mora biti unaprijed definiran i pohranjen jer se koristi i za generiranje parametra 'code_challenge' koji se šalje u prethodnom, autorizacijskom zahtjevu na autentikacijski poslužitelj.

4. HTTP odgovor s token krajnje točke

U slučaju da je čitav tijek prošao u redu, ovaj HTTP odgovor sadrži krajnji rezultat autentikacije, tokene. Odgovor će uvijek sadržavati pristupni token (eng. access token). Ako je u autorizacijskom zahtjevu korišten opseg 'openid', sadržavat će i ID token. Uz njih, bit će izdan i token za osvježavanje (eng. refresh token).

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
Cache-Control: no-store
Pragma: no-cache

{
  "id_token": "{ID Token}",            // Ukljuen ako je scope sadržavao vrijednost 'openid', inae ne
  "access_token": "{Access Token}",    // Ukljuen
  "token_type": "{Token Type}",        // Ukljuen
  "expires_in": {Lifetime In Seconds}, // Ukljuen
  "refresh_token": "{Refresh Token}",  // Ukljuen ako je scope sadržavao vrijednost 'offline_access', inae ne
}
```

4.1 Validacija ID tokena

Dobiveni ID token je potrebno provjeriti na način:

- pomoću javnog ključa provjeriti ispravnost potpisa
- provjeriti je li token istekao (vrijednost tvrdnje 'exp' - vrijeme isteka, eng. expiration time)
- provjeriti može li se token početi koristiti (vrijednost tvrdnje 'nbf' - eng. not before)
- provjeriti je li vrijednost tvrdnje 'aud' (publika, eng. audience) jednaka vrijednosti 'client_id' (ID token je namijenjen određenom klijentu)
- provjeriti je li vrijednost tvrdnje 'iss' (izdavatelj, eng. issuer) jednaka vrijednosti svojstva 'issuer' u JSON objektu na OIDC konfiguracijskom URL-u
- provjeriti je li vrijednost tvrdnje 'nonce' jednaka vrijednosti koja je poslana u prvom zahtjevu prema autorizacijskoj krajnjoj točki

Ako bilo koja provjera ne uspije, potrebno je odbaciti ID token i prekinuti autentikacijski tijek.

5. HTTP zahtjev na krajnju točku za korisničke podatke

Korisnički podaci se mogu dohvatiti slanjem dodatnog HTTP zahtjeva na krajnju točku za korisničke podatke (eng. userinfo endpoint). URL krajnje točke za korisničke podatke je dostupan u svojstvu 'userinfo_endpoint' u JSON objektu na OIDC konfiguracijskom URL-u. HTTP zahtjev na krajnju točku za korisničke podatke se može slati pomoću metoda GET ili POST, a preporučena metoda je GET.

U slučaju metode GET, u HTTP zahtjevu je potrebno postaviti 'Authorization' HTTP zaglavlje koje će za vrijednost imati 'Bearer {Access Token}' (umjesto '{Access Token}' iskoristiti pristupni token dobiven u prethodnom zahtjevu prema token krajnjoj točki):

```
GET {Userinfo Endpoint} HTTP/1.1
Authorization: Bearer {Access Token}
Host: {Authorization Server}
```

U slučaju metode POST, u tijelu zahtjeva mora biti postavljen parametar 'access_token' koji za vrijednost ima pristupni token dobiven u prethodnom zahtjevu prema token krajnjoj točki:

```
POST {Userinfo Endpoint} HTTP/1.1
Host: {Authorization Server}
Content-Type: application/x-www-form-urlencoded

access_token={Access Token}
```

Primjer odgovora sa krajnje točke za korisničke podatke:

```
HTTP/1.1 200 OK
Content-Length: 25
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: application/json


{"sub":"bfa1605be44a50a7c","name":"Ivan Horvat","family_name":"Horvat","given_name":"Ivan","
preferred_username":"ihorvat@primjer.hr","email":"ivan.horvat@primjer.hr","hrEduPersonUniqueNumber":
["LOCAL_NO: 1234","OIB: 12345678912","JMBAG: 1234567891"]}
```

Implicitni tijek

Implicitni tijek koristi samo autorizacijsku krajnju točku koja ID token i eventualno pristupni token (eng. access token) (uopće se ne koristi token krajnja točka). Ako se izdaje pristupni token, ua dohvat podataka o korisniku koristi se krajnja točka za korisničke podatke (eng. userinfo endpoint), a ako ne, ID token će sadržavati i korisničke podatke. Ovaj tijek ne izdaje token za osvježavanje (eng. refresh token).

Zbog veće razine sigurnosti, AAI@EduHr preporučuje korištenje tijeka autorizacijskog koda umjesto implicitnog tijeka.

Ukratko, implicitni tijek se obavlja na način:

- klijent napravi autorizacijski HTTP zahtjev preusmjeravanjem web preglednika na autorizacijsku krajnju točku
- krajnji korisnik se autenticira
- autentikacijski poslužitelj preusmjeravanjem web preglednika na registriranu lokaciju za preusmjeravanje (eng. redirect URI) šalje klijentu ID token, te pristupni token (eng. access token) ako je zahtjevan

Dakle, Dakle, klijent dobije ID token (i eventualno pristupni token) u odgovoru sa autorizacijske kranje točke kao parametre u fragment dijelu URI-a (dio nakon znaka '#'). To znači da će ID token (i pristupni token) biti izloženi u tkz. prednjem kanalu (eng. front channel), dakle korisničkom agentu (eng. user-agent) kao što je web preglednik (a time i svim možebitnim malicioznim aplikacijama koje imaju pristup korisničkom agentu). Također, u implicitnom tijeku nema autentikacije klijenta prije izdavanja tokena (ne koristi se tajni ključ klijenta (eng. secret).

Slijedi opis svih HTTP zahtjeva.

1. HTTP zahtjev na autorizacijsku krajnju točku

Zahtjev na autorizacijsku krajnju točku u implicitnom tijeku se obavlja na isti način kao i u tijeku autorizacijskog koda, ali sa sljedećim razlikama u parametrima:

Parametar	Vrijednost	Napomena
response_type	'id_token token' ili 'id_token'	Obavezan parametar. Ako se koristi vrijednost 'id_token token', izdat će se i ID token i pristupni token. Ako se koristi vrijednost 'id_token', izdat će se samo ID token.
nonce	Vrijednost mora biti niz znakova s dovoljnom entropijom, dakle kriptografski siguran (ne smije ga biti lako pogoditi). Preporučuje se napraviti npr. raspršivanje (eng. hash) kolačića sjednice (eng. session cookie), ili raspršivanje kriptografski slučajnog niza znakova (eng. random string) po svakom zahtjevu.	Obavezan parametar. Vrijednost parametra koja je dana u ovom zahtjevu će biti vraćena kao jedna od tvrdnji u ID tokenu, pa ju je tamo potrebno i provjeriti (mora biti ista). Tako se povezuje autorizacijski zahtjev s izdanim ID tokenom, pa klijent može potvrditi da je to ID token koji je zatražen.

2. HTTP odgovor sa autorizacijske krajnje točke

Nakon što se krajnji korisnik autenticira, autentikacijski poslužitelj će generirati ID token (i eventualno pristupni token) i vratit ih klijentu preusmjeravanjem na lokaciju za preusmjeravanje koja je korištena u prvom zahtjevu na autorizacijsku krajnju točku. Također, ako je korišten parametar 'state', vratit će i njega. Vrijednost parametra 'state' je potrebno provjeriti na klijentskoj strani (mora biti isti onaj koji je korišten u prvom zahtjevu prema autorizacijskoj krajnjoj točki). Parametri će biti vraćeni u fragment dijelu URI-a (dio nakon znaka '#').

```
HTTP/1.1 302 Found
Location: {Redirect URI}
  #id_token={ID Token}  // Uvijek ukljuen
  &access_token={Access Token} // Ukljuen ako je parametar 'response_type' imao vrijednost 'id_token token',
inae ne
  &token_type='Bearer' // Ukljuen na isti nain kao i 'access_token'
  &expires_in=3600 // Vrijeme isteka pristupnog tokena u sekundama, ukljuen na isti nain kao i 'access_token'
  &state={Arbitrary String}   // Ukljuen ako je zahtjev na autorizacijsku krajnju toku imao parametar 'state'
```

2.1 Validacija ID tokena

ID token je potrebno validirati na isti način kao što je opisano u tijeku autorizacijskog koda, uz sljedeće napomene:

- tvrdnja 'nonce' će uvijek postojati u ID tokenu u implicitnom tijeku, pa ju je uvijek potrebno i provjeravati

2.2. Validacija pristupnog tokena (ako je izdan)

Pristupni token je potrebno validirati na sljedeći način:

- napraviti raspršivanje (eng. hash) okteta ASCII reprezentacije pristupnog tokena pomoću algoritma za raspršivanje koji je naveden u parametru 'alg' u zaglavlju ID tokena (npr. za algoritam RSA256 to će biti sha256).
- uzeti lijevu polovicu raspršene vrijednosti pa ju Base64 URL kodirati
- usporediti dobivenu vrijednost sa vrijednošću iz tvrdnje 'at_hash' u ID tokenu (moraju biti iste)

3. HTTP zahtjev na krajnju točku za korisničke podatke (ako je izdan pristupni token)

Ako je izdan pristupni token, potrebno ga je iskoristiti kao Bearer token u zahtjevu prema krajnjoj točki za korisničke podatke (za dohvat korisničkih podataka), na isti način kao i u tijeku autorizacijskog koda. Ako pristupni token nije izdan, korisnički podaci će biti sadržani u ID tokenu, pa ovaj zahtjev nije potrebno obavljati.

OAuth2 Implicitni tijek

Ovo je standardni OAuth2 implicitni tijek, što znači da se izdaje samo pristupni token (eng. access token). ID token se ne izdaje, neovisno o tome je li u parametru 'scope' navedena vrijednost 'openid'. Ovaj tijek koristi samo autorizacijsku krajnju točku koja izdaje pristupni token (uopće se ne koristi token krajnja točka). Za dohvat podataka o korisniku koristi se krajnja točka za korisničke podatke (eng. userinfo endpoint). Ovaj tijek ne izdaje token za osvježavanje (eng. refresh token).

Zbog veće razine sigurnosti, AAI@EduHr preporučuje korištenje tijeka autorizacijskog koda umjesto implicitnog tijeka.

Detaljan opis tijeka dostupan je u specifikaciji OAuth2: https://tools.ietf.org/html/rfc6749#section-4.2 .

Ukratko, implicitni tijek se obavlja na način:

- klijent napravi autorizacijski HTTP zahtjev preusmjeravanjem web preglednika na autorizacijsku krajnju točku
- krajnji korisnik se autenticira
- autentikacijski poslužitelj preusmjeravanjem web preglednika na registriranu lokaciju za preusmjeravanje (eng. redirect URI) šalje klijentu pristupni token (eng. access token) kao fragment u URL-u (dio nakon znaka #)
- klijent napravi HTTP zahtjev na krajnju točku za korisničke podatke pomoću pristupnog tokena

Slijedi opis pojedinih HTTP zahtjeva.

1. HTTP zahtjev na autorizacijsku krajnju točku

Autentikacijski tijek počinje HTTP zahtjevom na autorizacijsku krajnju točku (eng. authorization endpoint). URL autorizacijske krajnje točke je dostupan u svojstvu 'authorization_endpoint' u JSON objektu na OIDC konfiguracijskom URL-u. Ovaj HTTP zahtjev se obavlja preusmjeravanjem korisničkog agenta (web preglednika) na autentikacijski poslužitelj preko unaprijed pripremljenog URL-a s točno definiranim GET parametrima. Primjer pripremljenog URL-a za preusmjeravanje korisničkog agenta (razlomljeno na više redova radi preglednosti):

```
https://login.aaiedu.hr/sso/module.php/oidc/authorize.php?
response_type=code&
client_id=neki-id-klijenta&
redirect_uri=https://neki-uri-za-preusmjeravanje.primjer.hr&
scope=profile email&
state=neki-state-132
```

Ovisno o tome koji tip klijenta se koristi, URL mora sadržavati određene GET parametre. Slijedi opis obaveznih i opcionalnih GET parametra:

Parametar	Vrijednost	Napomena
response_type	'token'	Obavezan parametar. Vrijednost 'token' signalizira da se radi o OAuth2 implicitnom tijeku.
client_id	ID klijenta koji se dobije prilikom registracije klijenta.	Obavezan parametar.
redirect_uri	Jedna od vrijednosti lokacija za preusmjeravanje koje su definirane prilikom registracije klijenta.	Obavezan parametar.
scope	Lista standardnih ili AAI@EduHr opsega odvojenih razmakom. Moguće vrijednosti opsega su navedene u JSON svojstvu 'scopes_supported' na OIDC konfiguracijskom URL-u.	Obavezan parametar.
state	Ako se koristi, vrijednost mora biti niz znakova s dovoljnom entropijom, dakle kriptografski siguran (ne smije ga biti lako pogoditi). Preporučuje se napraviti npr. raspršivanje (eng. hash) kolačića sjednice (eng. session cookie), ili raspršivanje kriptografski slučajnog niza znakova (eng. random string) po svakom zahtjevu.	Opcionalan parametar, ali je preporučen zbog zaštite od napada krivotvorenja zahtjeva za više web lokacija (eng. Cross-Site Request Forgery - CSRF). Vrijednost parametra koja je dana u ovom zahtjevu će biti vraćena kao jedan od parametara kod preusmjeravanja na registriranu lokaciju za preusmjeravanje, pa ju je tamo potrebno i provjeriti (mora biti ista). Na taj način se osigurava da na lokaciji za preumjeravanje stignu HTTP zahtjevi koje je klijent tražio (dakle, samo odgovori na autorizacijski zahtjev). Parametar 'state' se ponekad koristi i za postavljanje URL-a na kojeg je kranjeg korisnika potrebno preusmjeriti nakon procesa autentikacije (implementirati po potrebi, uz napomenu da bi URL za preusmjeravanje također trebao sadržavati slučajan niz znakova, npr. kao GET parametar, kako bi se očuvao smisao korištenja parametra).

U nastavku je opis HTTP GET zahtjeva kojeg korisnički agent obavi nakon što ga se preusmjeri na pripremljeni URL:

```
GET {Authorization Endpoint}
  ?response_type=token          // Obavezno
  &client_id={Client ID}        // Obavezno
  &redirect_uri={Redirect URI}  // Obavezno
  &scope={Scopes}               // Obavezno
  &state={Arbitrary String}     // Preporueno zbog zaštite od napada krivotvorenja zahtjeva za više web
lokacija (eng. Cross-Site Request Forgery - CSRF)
  HTTP/1.1
HOST: {Authorization Server}
```

2. HTTP odgovor sa autorizacijske krajnje točke

Nakon što se krajnji korisnik autenticira, autentikacijski poslužitelj će generirati pristupni token i vratiti ga klijentu preusmjeravanjem na lokaciju za preusmjeravanje koja je korištena u prvom zahtjevu na autorizacijsku krajnju točku. Također, ako je korišten parametar 'state', vratit će i njega. Svi parametri će biti vraćeni u fragment dijelu URL-a (nakon znaka #). Vrijednost parametra 'state' je potrebno provjeriti na klijentskoj strani (mora biti isti onaj koji je korišten u prvom zahtjevu prema autorizacijskoj krajnjoj točki).

```
HTTP/1.1 302 Found
Location: {Redirect URI}
  #access_token={Access Token}       // Ukljuen
  &token_type={Token Type}           // Ukljuen
  &expires_in={Lifetime In Seconds}  // Ukljuen
  &state={Arbitrary String}          // Ukljuen ako je zahtjev na autorizacijsku krajnju toku imao parametar
'state'
```

3. HTTP zahtjev na krajnju točku za korisničke podatke

Korisnički podaci se dohvaćaju slanjem HTTP zahtjeva na krajnju točku za korisničke podatke (eng. userinfo endpoint). URL krajnje točke za korisničke podatke je dostupan u svojstvu 'userinfo_endpoint' u JSON objektu na OIDC konfiguracijskom URL-u. U HTTP zahtjevu je potrebno postaviti 'Authorization' HTTP zaglavlje koje će za vrijednost imati 'Bearer {Access Token}' (umjesto '{Access Token}' iskoristiti pristupni token dobiven u prethodnom zahtjevu prema autorizacijskoj krajnjoj točki).

```
GET {Userinfo Endpoint} HTTP/1.1
Authorization: Bearer {Access Token}
Host: {Authorization Server}
```

Primjer odgovora s krajnje točke za korisničke podatke:

```
HTTP/1.1 200 OK
Content-Length: 25
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: application/json


{"sub":"bfa1605be44a50a7c","name":"Ivan Horvat","family_name":"Horvat","given_name":"Ivan","
preferred_username":"ihorvat@primjer.hr","email":"ivan.horvat@primjer.hr","hrEduPersonUniqueNumber":
["LOCAL_NO: 1234","OIB: 12345678912","JMBAG: 1234567891"]}
```

Tijek tokena za osvježavanje

Tijek tokena za osvježavanje (eng. refresh token flow) omogućuje ponovni dohvat pristupnog i ID tokena, bez potrebe ponovne autentikacije krajnjeg korisnika. Za osvježavanje tokena koristi se token krajnja točka čiji je URL dostupan u svojstvu 'token_endpoint' u JSON objektu na OIDC konfiguracijskom URL-u. Klijent može zatražiti izdavanje tokena za osvježavanje koristeći opseg 'offilne_access' u tijeku autorizacijskog koda.

Slijedi opis pojedinih HTTP zahtjeva.

1. HTTP zahtjev na token krajnju točku

Opis potrebnih POST parametara:

Parametar	Vrijednost	Napomena
grant_type	'refresh_token'	Obavezan parametar. Vrijednost 'refresh_token' signalizira da se radi o tijeku tokena za osvježavanje.
refresh_token	Vrijednost tokena za osvježavanje dobivena u tijeku autorizacijskog koda.	Obavezan parametar.
client_id	ID klijenta koji se dobije prilikom registracije klijenta.	Obavezan parametar.
client_secret	Tajni ključ klijenta koji se definira prilikom registracije klijenta.	Obavezan parametar.
scope	Ako se koristi, vrijednost mora biti ili dio originalno odabranih opsega ili svi originalni opsezi. Ne mogu se dodavati novi opsezi koji nisu bili u originalnom autorizacijskom zahtjevu.	Opcionalan parametar. Ako se ne koristi, podrazumijevana vrijednost je originalno korištena lista opsega u autorizacijskom zahtjevu.

Primjer HTTP POST zahtjeva:

```
POST {Token Endpoint} HTTP/1.1
Host: {Authorization Server}
Content-Type: application/x-www-form-urlecoded

grant_type=refresh_token        // Obavezno
&refresh_token={Refresh Token}  // Obavezno
&client_id={Client ID}          // Obavezno
&client_secret={Client secret}  // Obavezno
&scope={Scopes}                 // Opcionalno
```

2. HTTP odgovor s token krajnje točke

Odgovor će uvijek sadržavati novi pristupni token i token za osvježavanje. Ako je u originalnom zahtjevu korišten opseg 'openid', odgovor će sadržavati i osvježeni ID token. ID token će imati tvrdnje 'iat' i 'nbf' postavljene na vremensku oznaku (eng. timestamp) u trenutku osvježavanja, a 'exp' na novu vremensku oznaku isteka. Ostale tvrdnje će imati vrijednosti kao i u originalnom ID tokenu.

Primjer odgovora:

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
Cache-Control: no-store
Pragma: no-cache

{
  "access_token": "{Access Token}",    // Ukljuen
  "token_type": "{Token Type}",        // Ukljuen
  "id_token": "{ID Token}",            // Ukljuen ako je originalni autorizacijski zahtjev u parametru scope
sadržavao vrijednost 'openid', inae ne
  "expires_in": {Lifetime In Seconds}, // Ukljuen
  "refresh_token": "{Refresh Token}",  // Ukljuen
}
```

Tijekovi za odjavu

Sustav AAI@EduHr trenutno podržava sljedeće specifikacije vezane za odjavu korisnika:

- odjava pokrenuta od pouzdane strane (eng. Relying Party Initiated Logout 1.0): https://openid.net/specs/openid-connect-rpinitiated-1_0.html
- odjava u pozadinskom kanalu (eng. Back-Channel Logout 1.0): https://openid.net/specs/openid-connect-backchannel-1_0.html

Pokretanje odjave od pouzdanje strane omogućuju resursima (uslugama) standardizirano slanje zahtjeva za odjavom prema autentikacijskom poslužitelju. Nakon što autentikacijski poslužitelj dobije takav zahtjev, prekinut će korisničku sjednicu te će o tome obavijestiti druge resurse (pouzdane strane) putem protokola za odjavu u pozadinskom kanalu (ako pouzdana strana podržava taj protokol).

Uz spomenute specifikacije, OIDC propisuje i sljedeće specifikacije vezane za održavanje sjednice i odjavu korisnika, a koje sustav AAI@EduHr trenutno ne podržava:

- upravljanje sjednicama (eng. Session Management 1.0): https://openid.net/specs/openid-connect-session-1_0.html
- odjava u prednjem kanalu (eng. Front-Channel Logout 1.0): https://openid.net/specs/openid-connect-frontchannel-1_0.html

Pokretanje odjave od pouzdane strane

Ukratko, pokretanje odjave od pouzdane strane obavlja se preusmjeravanjem korisničkog agenta na krajnju točku za prekidanje sjednice. URL za krajnju točku za prekidanje sjednice je dostupan u svojstvu 'end_session_endpoint' u JSON objektu na OIDC konfiguracijskom URL-u.

1. HTTP zahtjev na krajnju točku za prekidanje sjednice

HTTP zahtjev na krajnju točku za prekidanje sjednice može se obaviti metodama GET ili POST, a zahtjev može imate sljedeće parametre:

Parametar	Vrijednost	Napomena
id_token_hint	ID token koji je autentikacijski poslužitelj prethodno izdao pouzdanoj strani.	Opcionalan parametar, ali je preporučen jer autentikacijskom poslužitelju služi kao indikator identiteta korisnika kojeg je potrebno odjaviti i kao indikator o sjednici korisnika koju je potrebno prekinuti. Ako ovaj parametar nije dan, prekinut će se trenutno aktivna sjednica korisnika na autentikacijskom poslužitelju (ako postoji).
post_logout_redirect_uri	Jedna od vrijednosti lokacija za preusmjeravanje nakon pokretanja odjave koje su definirane prilikom registracije klijenta.	Opcionalan parametar. Ako se koristi, parametar 'id_token_hint' je obavezan.

state	Ako se koristi, vrijednost mora biti niz znakova s dovoljnom entropijom, dakle kriptografski siguran (ne smije ga biti lako pogoditi). Preporučuje se napraviti npr. raspršivanje (eng. hash) kolačića sjednice (eng. session cookie), ili raspršivanje kriptografski slučajnog niza znakova (eng. random string) po svakom zahtjevu.	Opcionalan parametar, ali je preporučen zbog zaštite od napada krivotvorenja zahtjeva za više web lokacija (eng. Cross-Site Request Forgery - CSRF). Vrijednost parametra koja je dana u ovom zahtjevu će biti vraćena kao jedan od parametara kod preusmjeravanja na registriranu lokaciju za preusmjeravanje nakon pokretanja odjave, pa ju je tamo potrebno i provjeriti (mora biti ista). Tako se osigurava da na lokaciji za preusmjeravanje nakon pokretanja odjavestignu HTTP zahtjevi koje je klijent tražio (dakle, samo odgovori na zahtjev za odjavom). Parametar 'state' se ponekad koristi i za postavljanje URL-a na kojeg je krajnjeg korisnika potrebno preusmjeriti nakon procesa odjave (implementirati po potrebi, uz napomenu da bi URL za preusmjeravanje također trebao sadržavati slučajan niz znakova, npr. kao GET parametar, kako bi se očuvao smisao korištenja parametra).

U nastavku je primjer HTTP GET zahtjeva kojeg korisnički agent (npr. web preglednik) obavi nakon što ga se preusmjeri na pripremljeni URL:

```
GET {Logout Endpoint}
  ?id_token_hint={ID token}                                    // Preporueno
  &post_logout_redirect_uri={Post-Logout Redirect URI}         // Opcionalno
  &state={Arbitrary String}                                    // Opcionalno
  HTTP/1.1
HOST: {Authorization Server}
```

Nakon što autentikacijski poslužitelj dobije zahtjev za odjavom, krenut će obavještavati o tom događaju sve pouzdane strane na kojima se korisnik prijavio (ako pouzdana strana podržava primanje takvih obavijesti). Zbog toga, pouzdana strana koja je pokrenula odjavu sama bira hoće li i lokalno odjaviti korisnika prije slanja zahtjeva za odjavom prema autentikacijskom poslužitelju ili ne. Naime, ako pouzdana strana koja je inicirala odjavu ima implementiranu podršku za npr. odjavom u pozadinskom kanalu, također će dobit obavijest o odjavi sa autentikacijskog poslužitelja.

2. (opcionalno) Preusmjeravanje na pouzdanu stranu nakon odjave

Ako je u zahtjevu za odjavom dan parametar 'post_logout_redirect_uri' (i parametar 'id_token_hint'), autentikacijski poslužitelj će nakon odjave preusmjeriti korisnički agent na URI iz tog parametra, zajedno sa vrijednošću 'state' (ako je parametar 'state' dan u zahtjevu za odjavom. URI iz parametra 'post_logout_redirect_uri' mora biti prethodno registriran u AAI@EduHr registru resursa.

Odjava u pozadinskom kanalu

Odjava u pozadinskom kanalu je način odjave u kojoj autentikacijski poslužitelj javlja pouzdanoj strani o potrebi za odjavom korisnika direktnom komunikacijom (bez korištenja korisničkog agenta kao što je web preglednik).

Pošto se u ovom postupku odjave ne koristi korisnički agent kao što je web preglednik, nema načina da se iskoristi informacija o stanju sjednice pomoću npr. kolačića iz web preglednika. Zbog toga, sve potrebne informacije o stanju sjednice moraju se direktno razmijeniti između autentikacijskog poslužitelja i pouzdane strane. Također, prekidanje lokalne sjednice na pouzdanoj strani može biti malo kompliciranije, pošto nema mogućnosti da se npr. jednostavno obriše kolačić lokalne sjednice u web pregledniku. Zbog toga, nema generalnog recepta kako prekinuti lokalnu sjednicu na pouzdanoj strani, nego je to specifično obzirom na programsko okruženje i način vođenja lokalne sjednice.

Ako pouzdana strana podržava odjavu u pozadinskom kanalu, mora registrirati lokaciju za odjavu (URI) u pozadinskom kanalu prilikom registracije OIDC klijenta u AAI@EduHr registru resursa. Lokacija za odjavu u pozadinskom kanalu na koji će autentikacijski poslužitelj pouzdanoj strani slati zahtjeve za odjavom u pozadinskom kanalu uvijek treba biti direktno dostupan autentikacijskom poslužitelju. Ako lokacija za odjavu u pozadinskom kanalu nije registrirana, autentikacijski poslužitelj neće slati zahtjeve za odjavom toj pouzdanoj strani.

Token za odjavu

Token za odjavu (engl. Logout Token) je JWT token kojeg će autentikacijski poslužitelj poslati pouzdanoj strani kao informaciju o potrebi za odjavom korisnika. Token za odjavu je po svojoj strukturi sličan ID tokenu koji se koristi prilikom autentikacije.

Primjer tvrdnji u token za odjavu u dijelu s korisnim podacima:

```
{
  "iss": "https://login.aaiedu.hr",
  "sub": "bfa1605be44a50a7c",
  "aud": "6e55295209782b7b2",
  "iat": 1602674470,
  "jti": "c44f4cffcc84f7990f7a1d5b2c",
  "sid": "52tn95meBL578282H7V",
  "events": {
    "http://schemas.openid.net/event/backchannel-logout": {}
  }
}
```

Za razliku od ID tokena, token za odjavu sadržava tvrdnju 'events' kao JSON objekt koji ima svojstvo 'http://schemas.openid.net/event/backchannel-logout'. Time se token označuje kao token za odjavu. Također, za razliku od ID tokena, token za odjavu nikad neće sadržavati tvrdnju 'nonce'.

Token za odjavu će sadržavati ili tvrdnju 'sub' ili tvrdnju 'sid', ili obje spomenute tvrdnje (mora sadržabati barem jednu od njih). Ako je tvrdnja 'sid' prisutna, potrebno je prekinuti sjednicu na pouzdanoj strani sa identifikatorom iz te tvrdnje. Ako tvrdnja 'sid' nije prisutna, potrebno je prekinuti sve sjednice na pouzdanoj strani za korisnika iz tvrdnje 'sub'.

Token za odjavu će imati isto zaglavlje kao i ID token, te će biti potpisan. Primljeni token za odjavu je potrebno provjeriti na sličan način kao i ID token (provjera potpisa, ispravnosti tvrdnje 'iss', itd.), uz nekoliko dopuna:

- provjeriti postoji li tvrdnja 'events' sa svojstvom 'http://schemas.openid.net/event/backchannel-logout'
- provjeriti da ne postoji tvrdnja 'nonce'
- provjeriti jesu li tvrdnje 'iss', 'sub', 'sid' istovjetne onima iz ID tokena za trenutnu ili nedavno postojeću korisničku sjednicu
- provjeriti postoje li barem jedna od tvrdnji 'sid' ili 'sub', ili obje

Ostale provjere mogu biti vezane za ostale tvrdnje kako slijedi:

- 'iat' - kada je token izdan. Token ne smije biti iz budućnosti ili izdan predaleko u prošlosti (npr. ne stariji od 2 minute)
- 'jti' - jedinstveni identifikator tokena,, npr. u smislu je li nedavno korišten isti token
- 'iss' - je li token izdan od očekivanog izdavatelja
- 'aud' - odgovara li vrijednost identifikatoru klijenta

Ako bilo koja od provjera ne prođe, token za odjavu je potrebno odbaciti kao nevaljan.

1. Zahtjev za odjavom u pozadinskom kanalu

Autentikacijski poslužitelj šalje HTTP zahtjev koristeći metodu POST na registriranu lokaciju za odjavu u pozadinskom kanalu pouzdane strane. Tijelo zahtjeva sadrži parametar 'logout_token' čija je vrijednost Base64 URL kodirani token za odjavu. Na taj način autentikacijski poslužitelj javlja pouzdanoj strani kojeg korisnika treba lokalno odjaviti.

```
POST {Back-Channel Logout Endpoint} HTTP/1.1
Host: {Relying Party Host}
Content-Type: application/x-www-form-urlecoded

logout_token={Logout Token}  // Obavezno
```

2. Odgovor na zahtjev za odjavom u pozadinskom kanalu

Pouzdana strana odgovara na zahtjev za odjavom u pozadinskom kanalu kako slijedi:

- ako je odjava uspjela, treba vratiti odgovor HTTP 200 OK
- ako odjava nije uspjela, treba vratiti odgovor HTTP 501 Not Implemented
- ako je zahtjev za odjavom neispravan, treba vratiti odgovor HTTP 400 Bad Request

Primjeri implementacije

Slijedi nekoliko primjera implementacije autentikacije u nekim programskim jezicima koristeći protokol OIDC. Navedeni primjeri su dani u pokazne svrhe te ne predstavljaju obavezu korištenja spomenutih programskih paketa, biblioteka, razvojnih okvira ili slično. No, svakako je preporuka koristiti neki postojeći, popularniji OIDC programski paket, jer oni tipično omogućuju laganiju i sigurniju implementaciju autentikacije. Tijekom stvaranja primjera korištene su tada aktualne verzije paketa, pa je moguće da se način implementacije pomoću njih u međuvremenu promijenio. Za najnovije upute potrebno je proučiti dokumentaciju samog programskog paketa koji se koristi.

Radi jednostavnosti prikaza samog koda, u primjerima su napravljene neke stvari koje bi u stvarnim aplikacijama trebalo izbjegavati, npr.:

- postupak autentikacije pokrenut je automatski, dok bi se u stvarnim aplikacijama postupak autentikacije mogao pokretati klikom na gumb 'Prijava', 'AAI@EduHr prijava', ili slično
- primjeri su napravljeni do trenutka dohvata korisničkih podataka, dok je u stvarnim aplikacijama dobivene korisničke podatke potrebno dodatno obraditi u smislu autorizacije, stvaranje korisničke sjednice (eng. session), ili slično
- konfiguracijske postavke su tvrdo kodirane (eng. hardcoded), dok bi ih u stvarnim aplikacijama trebalo dohvaćati iz varijabli okruženja (eng. environment variables), ili slično

PHP

Svi primjeri PHP paketa se mogu instalirati pomoću alata Composer.

jumbojett / OpenID-Connect-PHP

Repozitorij paketa: https://github.com/jumbojett/OpenID-Connect-PHP

U primjeru, čitav kod je sadržan u jednoj datoteci 'index.php'. U njoj se započinje postupak autentikacije, te također služi i kao lokacija za preusmjeravanje na koju će autentikacijski poslužitelj vratiti autorizacijski kod. Paket ima mogućnost automatskog dohvata OIDC konfiguracije sa OIDC konfiguracijskog URL-a (nije potrebno zasebno definirati krajnje točke). U trenutku pisanja primjera, ovaj paket je mogao koristiti samo certifikat u PEM formatu (ne koristi funkcionalnost dohvata javnog ključa iz JSON web skupa ključeva).

index.php

```
<?php
require 'vendor/autoload.php';

if (session_status() == PHP_SESSION_NONE) {
    session_start();
}

// TODO In real app, check if the user is already authenticated locally.
// If so, prevent further processing, show warning message, redirect to another page, or similar.

// Do the OIDC authentication.
$issuer = 'https://login.aaiedu.hr';
$clientId = 'YOUR_CLIENT_ID';
$clientSecret = 'YOUR_CLIENT_SECRET';

$oidcClient = new \Jumbojett\OpenIDConnectClient($issuer, $clientId, $clientSecret);

// Provide the path to the public certificate from OpenID Provider (OP)
$oidcClient->setCertPath(__DIR__ . '/../storage/idp.crt');
$oidcClient->addScope(['openid', 'profile', 'hrEduPersonUniqueNumber']);
$oidcClient->setRedirectURL('https://your-app.example.org/index.php');

// Start authentication or handle incoming authorization code.
$oidcClient->authenticate();

// Use appropriate methods to get user data.
// TODO In real app, log in the user locally, show success message, or similar.
var_dump($oidcClient->requestUserInfo('sub'));
var_dump($oidcClient->getVerifiedClaims());
```

steverhoades / oauth2-openid-connect-client

Repozitorij paketa: https://github.com/steverhoades/oauth2-openid-connect-client

U primjeru postoje tri datoteke:

- bootstrap.php - zajednički kod
- index.php - pokretanje autentikacije slanjem autorizacijskog zahtjeva
- callback.php - obrada dobivenog autorizacijskog koda i slanje zahtjeva za tokene

U trenutku pisanja primjera, ovaj paket je mogao koristiti samo certifikat u PEM formatu (ne koristi funkcionalnost dohvata javnog ključa iz JSON web skupa ključeva). Potrebno je ručno unijeti URL-ove za krajnje točke (nema mogućnost automatskog dohvata OIDC konfiguracije sa OIDC konfiguracijskog URL-a).

bootstrap.php

```
<?php
require 'vendor/autoload.php';

if (session_status() == PHP_SESSION_NONE) {
    session_start();
}

$oidcClient = new \OpenIDConnectClient\OpenIDConnectProvider(
    [
        'clientId' => 'YOUR_CLIENT_ID',
        'clientSecret' => 'YOUR_CLIENT_SECRET',
        'idTokenIssuer' => 'https://login.aaiedu.hr',
        'redirectUri' => 'https://your-app.example.org/callback.php',
        'urlAuthorize' => 'https://login.aaiedu.hr/sso/module.php/oidc/authorize.php',
        'urlAccessToken' => 'https://login.aaiedu.hr/sso/module.php/oidc/access_token.php',
        'urlResourceOwnerDetails' => 'https://login.aaiedu.hr/sso/module.php/oidc/userinfo.php',
        'publicKey' => 'file://' . __DIR__ . '/../storage/idp.crt',
    ],
    [
        'signer' => new \Lcobucci\JWT\Signer\Rsa\Sha256()
    ]
);

$oidcStateSessionKey = 'OIDC_STATE';
```

index.php

```
<?php
require 'bootstrap.php';

// TODO In real app, check if the user is already authenticated locally.
// If so, prevent further processing, show warning message, redirect to another page, or similar.

// Do the OIDC authentication.

// We must set desired scopes here, otherwise 'openid' scope will be used by default.
$redirectUrl = $oidcClient->getAuthorizationUrl(['scope' => 'openid profile hrEduPersonUniqueNumber']);

// Save state so we can validate it later.
$_SESSION[$oidcStateSessionKey] = $oidcClient->getState();

// Do the authorization request...
header('Location: ' . $redirectUrl);
exit();
```

callback.php

```
<?php
require 'bootstrap.php';

// TODO In real app, check if the user is already authenticated locally.
// If so, prevent further processing, show warning message, redirect to another page, or similar.

// We have to manually validate state
if (!isset($_GET['state']) || (! isset($_SESSION[$oidcStateSessionKey])) || $_GET['state'] != $_SESSION
[$oidcStateSessionKey]) {
    throw new \Exception('State parameter not valid.');
}

// State parameter is validated. We can remove it so it can't be reused (and this page can't be refreshed).
unset($_SESSION[$oidcStateSessionKey]);

// Get tokens...
try {
    $token = $oidcClient->getAccessToken('authorization_code', [
        'code' => $_GET['code']
    ]);
} catch (\Exception $e) {
    // TODO In real app log error, show error message, redirect to another page, or similar.
    throw $e;
}

// Use appropriate methods to get user data.
// TODO In real app, log in the user locally, show success message, or similar.
$accessToken = $token->getToken();
$refreshToken = $token->getRefreshToken();
$expires = $token->getExpires();
$hasExpired = $token->hasExpired();
$idToken = $token->getIdToken();
$email = $idToken->getClaim('email', false);
$allClaims = $idToken->getClaims();

var_dump($allClaims);
```

cicnavi / oidc-client-php

Repozitorij paketa: https://github.com/cicnavi/oidc-client-php

Paket ima mogućnost automatskog dohvata OIDC konfiguracije sa OIDC konfiguracijskog URL-a (nije potrebno zasebno definirati krajnje točke), te automatskog dohvata i obnove javnog ključa iz JSON web skupa ključeva.

Primjer korištenja vidljiv je u README datoteci u repozitoriju samog paketa: https://github.com/cicnavi/oidc-client-php

JavaScript

IdentityModel / oidc-client-js

Repozitorij paketa: https://github.com/IdentityModel/oidc-client-js

Paket je moguće instalirati pomoću alata 'npm' ili je moguće koristiti kompiliranu verziju dostupnu u mapi (eng. folder) 'dist'. Zbog mogućnosti različitog načina organiziranja JS koda te zbog tipično većeg broja potrebnih JS i HTML datoteka prilikom dizajniranja aplikacija, u nastavku je naveden samo dio koda relevantan za obavljanje OIDC autentikacije. Paket omogućuje korištenje tijeka autorizacijskog koda za javne klijente pomoću PKCE parametara, pa je iskorišten takav način autentikacije (takav način je preporučen umjesto implicitnog tijeka).

```
// TODO Make 'UserManager' class available in current context

// Example config to use for UserManager instance creation
const config = {
    authority: "https://login.aaiedu.hr/.well-known/openid-configuration",
    client_id: "YOUR_CLIENT_ID",
    redirect_uri: "https://your-app.example.org/callback",
    response_type : "code",
    scope: "openid profile hrEduPersonUniqueNumber",
    loadUserInfo: false // Prevent additional request to userinfo endpoint, since all inforomation is in ID
token
};

const userManager = new UserManager(config);

// Get user from local storage, if available (session storage is used by default).
// Local user will be available if the authentication process was already performed.
let user = null;
userManager.getUser().then(localUser => {user = localUser});

// Method used to initiate an authentication process. This will trigger a redirect
// to the authentication server (authorization request).
userManager.signinRedirect();

// Method used to handle incoming authorization code on the callback URI, and then to fetch tokens.
// The result is user object.
userManager.getUser().then(authenticatedUser => {user = authenticatedUser});

// User object contains tokens, claims...
console.log(user, user.profile);
```

Više primjera dostupno je u repozitoriju paketa: https://github.com/IdentityModel/oidc-client-js/tree/dev/samples

.NET

Microsoft.AspNetCore.Authentication.OpenIdConnect

NuGet paket: https://www.nuget.org/packages/Microsoft.AspNetCore.Authentication.OpenIdConnect

Unutar aplikacije potrebno je instalirati nuget paket: Microsoft.AspNetCore.Authentication.OpenIdConnect. Paket može koristiti tijek aturizacijskog koda i ima mogućnost automatskog dohvata OIDC konfiguracije sa OIDC konfiguracijskog URL-a, uključujući dohvat javnog ključa iz JSON web skupa ključeva. Lokacija za preusmjeravanje mora biti tipa 'https://neka-domena.neki-tld/{opcionalna-putanja}/signin-oidc'. Dakle, 'signin-oidc' je obavezan na kraju, jer je to middleware za .NET koji odrađuje cijeli postupak autentikacije. Korištenje opsega 'openid' je obavezano.

Datoteka appsettings.json:

appsettings.json

```
"OpenId": {
    "ClientId": "YOUR_CLIENT_ID",
    "ClientSecret": "YOUR_CLIENT_SECRET",
    "Authority": "https://login.aaiedu.hr/"
  }
```

Datoteke Startup.cs (kod koji nije relevantan za OIDC je uklonjen radi bolje čitljivosti):

Startup.cs

```
using Microsoft.AspNetCore.Authentication.Cookies;
using Microsoft.AspNetCore.Authentication.OpenIdConnect;

// ***
public void ConfigureServices(IServiceCollection services)
{
    // ***
    services.AddAuthentication(options => {
                options.DefaultScheme = CookieAuthenticationDefaults.AuthenticationScheme;
                options.DefaultChallengeScheme = OpenIdConnectDefaults.AuthenticationScheme;
            })
            .AddCookie()
            .AddOpenIdConnect(o =>
            {
                o.ResponseType = "code";
                o.ClientId = Configuration["OpenId:ClientId"];
                o.ClientSecret = Configuration["OpenId:ClientSecret"];
                o.Authority = Configuration["OpenId:Authority"];
                o.Scope.Add("openid");
                o.Scope.Add("hrEduPersonUniqueID");
                o.Scope.Add("cn");
                o.Scope.Add("mail");
                o.Scope.Add("hrEduPersonOIB");
                o.Scope.Add("hrEduPersonPersistentID");
                o.SignInScheme = "Cookies";
                o.TokenValidationParameters = new TokenValidationParameters
                {
                    ValidateIssuer = true,
                    ValidateIssuerSigningKey = true
                };
            });
    services.AddAuthorization();
    // ***
}

public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
{
    // ***
    app.UseAuthentication();
    app.UseAuthorization();
    // ***
}
```

Primjer dohvata korisničkih podataka:

```
var principal = HttpContext.User as ClaimsPrincipal;

foreach (var claim in principal.Claims)
{
    Console.WriteLine("Claim: " + claim.Type + " : " + claim.Value);
}

// *** or...
string commonName = User.FindFirst("cn").Value;
```

Hvala kolegi Vidu Kašiću na suradnji u pripremi upute za paket Microsoft.AspNetCore.Authentication.OpenIdConnect.

mod_auth_openidc

Repozitorij modula: https://github.com/zmartzone/mod_auth_openidc

mod_auth_openidc je modul Apache HTTP poslužitelja koji omogućuje provjeru autentičnosti i autorizacije kroz implementaciju OpenID Connect (OIDC) protokola. Modul funkcionira kao OpenID Connect Relying Party (RP) koji provjera autentičnost korisnika na OpenID Connect Provider-u, u ovom slučaju AAI autorizacijskom poslužitelju.

Prilikom pristupa zaštićenom sadržaju modul će od korisnika zatražiti sigurnosni token. Ukoliko token nije pronađen, modul će upit usmjeriti prema AAI autorizacijskom poslužitelju koji će nakon uspješne prijave dodijeliti token i time dozvoliti pristup.

Konfiguraciji se može pristupiti tek nakon što se u registru AAI resursa postave OIDC parametri. Kod otvaranja resursa prikazat će se vrijednosti **Client ID** i **Secret** (OIDCClientID i OIDCClientSecret). Pod **Lokacije za preusmjeravanje** upišite adresu gdje će korisnik biti preusmjeren nakon uspješne provjere autentičnosti (OIDCRedirectURI). Za kraj treba još odabrati **openid** i **profile** (OIDCScope) među raspoloživim standardnim opsezima.

Na samome poslužitelju je potrebno instalirati programsku podršku kroz paket libapache2-mod-auth-openidc na Debianu ili mod_auth_openidc na CentOS-u i potom omogućiti auth_openidc modul.

Konfiguracija se postavlja u Apache virtualnom hostu koristeći vrijednosti postavljene u prethodnim koracima.

Parametar	Opis	Primjer
`OIDCProviderMetadataURL*`	Konfiguracijska poveznica AAI autorizacijskog poslužitelja	https://fed-lab.aaiedu.hr/.well-known/openid-configuration
`OIDCRedirectURI`	**Lokacija za preusmjeravanje** postavljena u AAI resursu	`https://neki.host.hr/protected`
`OIDCClientID`	**Client ID** vrijednost postavljena u AAI resursu	123-123
`OIDCClientSecret`	**Secret** vrijednost postavljena u AAI resursu	123123123
`OIDCCryptoPassphrase`	Proizvoljna lozinka za zaštitu kolačića i predmemorijskih zapisa	123123123123
`OIDCRemoteUserClaim`	Vrijednost u kojoj se nalazi REMOTE_USER varijabla prijavljenog korisnika	preferred_username
`OIDCScope`	**Opseg** postavljen u AAI resursu	openid profile
`Location`	Konfiguracijski element zaštićene putanje	<Location "/"> Require valid-user AuthType openid-connect </Location>

OIDCProviderMetadataURL* poveznice po vrsti odabranog resursa:

Vrsta resursa	**`OIDCProviderMetadataURL`**
test	https://fed-lab.aaiedu.hr/.well-known/openid-configuration
produkcija	https://login.aaiedu.hr/.well-known/openid-configuration

Primjer Apache konfiguracije:

```
OIDCProviderMetadataURL https://fed-lab.aaiedu.hr/.well-known/openid-configuration
OIDCRedirectURI https://neki.host.hr/protected
OIDCClientID 123-123
OIDCClientSecret 123123123
OIDCCryptoPassphrase 123123123123
OIDCRemoteUserClaim preferred_username
OIDCScope "openid profile"

<Location "/">
  Require valid-user
  AuthType openid-connect
</Location>
```

Hvala kolegi Mariu Goljaku na pripremi uputa za mod_auth_openidc.

Dnevnik promjena (eng. changelog)

listopad 2022.

- dodana podrška za opseg 'offline_access'. Do sada, token za osvježavanje se uvijek izdavao u autentikacijskom tijeku autorizacijskog koda. Radi dodatnog usklađivanja sa OIDC specifikacijom, od sada token za osvježavanje se izdaje samo ako ga klijent "zatraži" pomoću opsega (eng. scope) 'offline_access' (https://openid.net/specs/openid-connect-core-1_0.html#OfflineAccess). Ovime je također ispravljena greška u kojoj je autentikacijski poslužitelj neispravno javljao da opseg 'offline_access' nije podržan, iako je uvijek vraćao token za osvježavanje. Implementacije koje se u autentikacijskom postupku oslanjaju na token za osvježavanje trebaju osigurati da u registru resursa https://registar.aaiedu.hr/ OIDC klijent ima registriran opseg 'offline_access'. Autentikacijski poslužitelj neće izdati token za osvježavanje ako klijent nema registran opseg 'offline_access'. Ova promjena nema utjecaja na klijente koji se ne oslanjanju na token za osvježavanje u atuentikacijskom postupku.

srpanj 2022.

Promjene koje mijenjaju postojeći način obavljanja autentikacije i mogu prouzročiti prekide (eng. breaking changes):

- ako se u nekom autentikacijskom tijeku izda pristupni token, ID token više ne sadrži tvrdnje o autenticiranom korisniku, nego samo tvrdnje o autentikacijskom događaju. Do sada, ID token je uvijek sadržavao i tvrdnje o autentikacijskom događaju i tvrdnje o korisniku, te je HTTP zahtjev na krajnju točku za korisničke podatke bio opcionalan. Od sada, ako je u autentikacijskom tijeku izdan i pristupni token, njega je potrebno iskoristiti za dohvat korisničkih podataka sa krajnje točke za korisničke podatke. Primjer autentikacijskog tijeka u kojem se izdaje pristupni token i koji je implementiran u AAI@EduHr je tijek autorizacijskog koda. Dakle, ova promjena utječe na one OIDC klijente koji su koristili tijek autorizacijskog koda i koji su podatke o korisniku dohvaćali samo iz ID tokena, te nisu radili poseban HTTP zahtjev na točku za korisničke podatke. Takvi klijenti će morati implementirati dohvat korisničkih podataka sa krajnje točke za korisničke podatke. Promjena je napravljena zbog usklađivanja sa OIDC specifikacijom (https://openid.net/specs/openid-connect-core-1_0.html#rfc.section.5.4).
- promijenjen je URI za 'token_endpoint' (krajnja točka za izdavanje tokena). Pošto se 'token_endpoint' koristi u samom autentikacijskom postupku, problem može nastati sa onim klijentima koji imaju zahardkodiranu vrijednost samog URI-a. Također, problem može nastati sa onim klijentima koji programatski dohvaćaju vrijednost sa AAI@EduHr OIDC konfiguracijskog URL-a (npr. https://login.aaiedu.hr/.well-known/openid-configuration za produkciju ili https://fed-lab.aaiedu.hr/.well-known/openid-configuration za testno okruženje), ali dohvaćenu konfiguraciju na neko vrijeme stavljaju u predmemoriju (eng. caching). Takvi klijenti će morati ažurirati URI za 'token_endpoint'. Kako bi se izbjegli prekidi u radu samih usluga, sustav AAI@EduHr će u prijelaznom razdoblju omogućiti korištenje i jednog i drugog URI-ja (starog i novog). Tijekom tog razdoblja potrebno je osigurati da klijenit koriste novi URI za 'token_endpoint'.

Nove značajke:

- implementiran OIDC implicitni tijek (https://openid.net/specs/openid-connect-core-1_0.html#ImplicitFlowAuth).
- implementirana mogućnost pokretanja odjave korisnika prema specifikacijama "RP-Initiated Logout 1.0" (https://openid.net/specs/openid-connect-rpinitiated-1_0.html) i "Back-Channel Logout 1.0" (https://openid.net/specs/openid-connect-backchannel-1_0.html). Sukladno tome, prilikom registracije klijenata sada je moguće definirati dozvoljene lokacije za preusmjeravanje nakon pokretanja odjave i lokaciju za odjavu u pozadinskom kanalu.
- poboljšan prikaz i objašnjenje grešaka u autentikacijskom postupku. Do sada, u slučaju greške u autentikacijskom postupku prikazala bi se stranica sa porukom o neobrađenoj iznimci (eng. unhandled exception). Od sada, prikaz i objašnjenje grešaka je više u skladu sa OAuth2 i OIDC specifikacijom (https://datatracker.ietf.org/doc/html/rfc6749#section-5.2, https://openid.net/specs/openid-connect-core-1_0.html#AuthError). Na primjer, u slučaju da za određeni autentikacijski tijek nisu dani svi potrebni parametri prikazat će se poruka o neispravnom zahtjevu. U slučaju neispravnog identifikatora klijenta, lokacije za preusmjeravanje, tajnog ključa ili slično, prikazat će se poruka o neuspjeloj validaciji ili autentikaciji klijenta. U slučaju da je klijent uspješno validiran / autenticiran, ali problem postoji u nekom drugom parametru (npr. neispravan opseg ili slično), na lokaciju za preusmjeravanje će se vratiti poruka o grešci preko 'error' i 'error_description' parametra, kako spomenute specifikacije i predviđaju.
- implementirana podrška za parametar 'prompt' sa vrijednošću 'login' u autentikacijskom zahtjevu pomoću kojeg se može forsirati ponovna autentikacija korisnika, ako korisnik već ima aktivnu SSO sjednicu (https://openid.net/specs/openid-connect-core-1_0.html#AuthRequest).
- implementirana podrška za parametar 'max_age' u autentikacijskom zahtjevu pomoću kojeg se može definirati maksimalno proteklo vrijeme u sekundama od kad se korisnik autenticirao, ako korisnik već ima aktivno SSO sjednicu (https://openid.net/specs/openid-connect-core-1_0.html#AuthRequest)
- implementirano zahtjevanje pojedinih tvrdnji pomoću parametra 'claims' (https://openid.net/specs/openid-connect-core-1_0.html#ClaimsParameter)
- prilikom registracije javnog klijenta omogućeno definiranje dozvoljenih ishodišta (eng. allowed origins) sa kojih će biti dopušteno slati zahtjeve prema krajnjoj točki za korisničke podatke. Ovo je prvenstveno namijenjeno JavaScript klijentima tj. klijentima koje se izvršavaju u web pregledniku pa je za njih potrebno poštivati mehanizam CORS (Cross-Origin Resource Sharing https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS)
- omogućeno korištenje standardne tvrdnje 'address' (https://openid.net/specs/openid-connect-core-1_0.html#AddressClaim)
- u zahtjevu prema krajnjoj točki za korisničke podatke omogućeno slanje pristupnog tokena u tijelu HTTP zahtjeva (HTTP POST metoda, https://datatracker.ietf.org/doc/html/rfc6750#section-2.2)
- dodane tvrdnje 'issuer' i 'kid' u pristupni token
- dodana tvrdnja 'at_hash' u ID token - raspršena vrijednot pristupnog tokena (eng. access token hash value)
- dodan tvrdnja 'sid' u ID token - identifikator sjednice (eng. session ID)
- na OIDC konfiguracijskom URL-u, u JSON-u su ažurirana ili dodana sljedeća svojstva:
    - token_endpoint - promijenjen URI
    - end_session_endpoint - novo svojstvo (URI)
    - response_types_supported - promijenjeno, tj. u polje dodane nove vrijednosti
    - token_endpoint_auth_methods_supported - novo svojstvo (polje)
    - request_parameter_supported - novo svojstvo (boolean)
    - grant_types_supported - novo svojstvo (polje)

    - claims_parameter_supported - novo svojstvo (boolean)
    - backchannel_logout_supported - novo svojstvo (boolean)
    - backchannel_logout_session_supported - novo svojstvo (boolean)

Ispravke:

- u pristupnom tokenu osigurano postavljanje vremenskih oznaka u cijelobrojnom obliku
- u slučaju ponovne upotrebe istog autorizacijskog koda, svi tokeni izdani na temelju tog autorizacijskog koda će biti označeni kao nevaljani (https://tools.ietf.org/html/rfc6749#section-4.1.2)

studeni 2021.

- dodan primjer implementacije pomoću Apache modula mod_auth_openidc

travanj 2021.

- dodan opis korištenja u testnom okruženju AAI@EduHr Lab
- dodan primjer implementacije u PHP-u pomoću paketa cicnavi/oidc-client-php

veljača 2021.

- dodan primjer imlementacije u okruženju .NET pomoću NuGet paketa Microsoft.AspNetCore.Authentication.OpenIdConnect