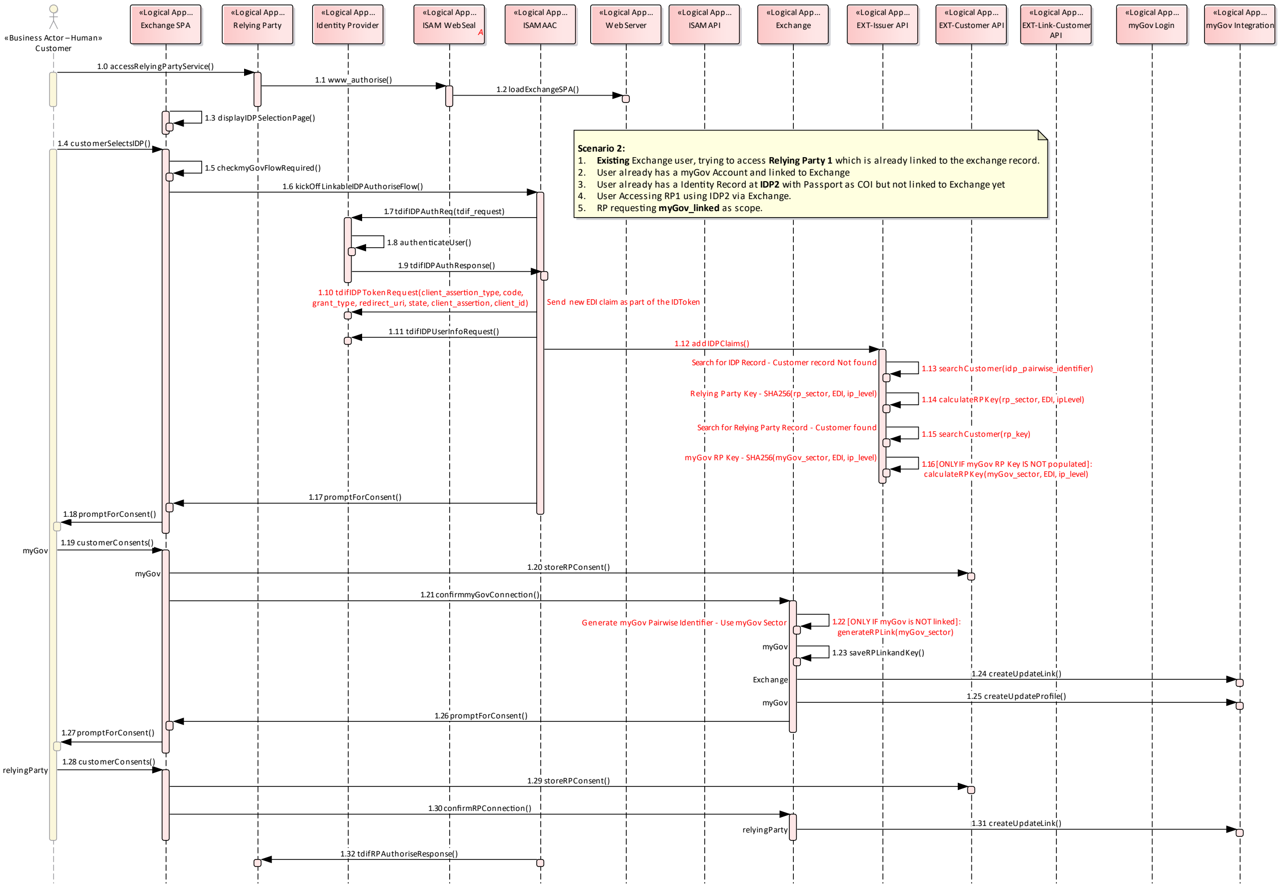
S2 - Existing Customer - Existing RP - New IDP - Existing myGov

Ram Challuri

15/11/2019 4:00:27 PM



Scenario 2:

1. **Existing** Exchange user, trying to access **Relying Party 1** which is already linked to the exchange record.
2. User already has a myGov Account and linked to Exchange
3. User already has a Identity Record at **IDP2** with Passport as COI but not linked to Exchange yet
4. User Accessing RP1 using IDP2 via Exchange.
5. RP requesting **myGov_linked** as scope.

«Business Actor – Human»
Customer
«Logical App...
Exchange SPA
«Logical App...
Relying Party
«Logical App...
Identity Provider
«Logical App...
ISAM WebSeal
A
«Logical App...
ISAM AAC
«Logical App...
Web Server
«Logical App...
ISAM API
«Logical App...
Exchange
«Logical App...
EXT-Issuer API
«Logical App...
EXT-Customer API
«Logical App...
EXT-Link-Customer API
«Logical App...
myGov Login
«Logical App...
myGov Integration
1.0 accessRelyingPartyService()
1.1 www_authorise()
1.2 loadExchangeSPA()
1.3 displayIDPSelectionPage()
1.4 customerSelectsIDP()
1.5 checkmyGovFlowRequired()
1.6 kickOff LinkableIDPAuthoriseFlow()
1.7 tdifIDPAuthReq(tdif_request)
1.8 authenticateUser()
1.9 tdifIDPAuthResponse()
1.10 tdifIDPTokenRequest(client_assertion_type, code, grant_type, redirect_uri, state, client_assertion, client_id)
Send new EDI claim as part of the IDToken
1.11 tdifIDPUserInfoRequest()
1.12 addIDPClaims()
Search for IDP Record - Customer record Not found
1.13 searchCustomer(idp_pairwise_identifier)
Relying Party Key - SHA256(rp_sector, EDI, ip_level)
1.14 calculateRPKey(rp_sector, EDI, ipLevel)
Search for Relying Party Record - Customer found
1.15 searchCustomer(rp_key)
myGov RP Key - SHA256(myGov_sector, EDI, ip_level)
1.16 [ONLY IF myGov RP Key IS NOT populated]: calculateRPKey(myGov_sector, EDI, ip_level)
1.17 promptForConsent()
1.18 promptForConsent()
myGov
1.19 customerConsents()
myGov
1.20 storeRPConsent()
1.21 confirmmyGovConnection()
Generate myGov Pairwise Identifier - Use myGov Sector
1.22 [ONLY IF myGov is NOT linked]: generateRPLink(myGov_sector)
myGov
1.23 saveRPLinkandKey()
Exchange
1.24 createUpdateLink()
myGov
1.25 createUpdateProfile()
1.26 promptForConsent()
1.27 promptForConsent()
relyingParty
1.28 customerConsents()
1.29 storeRPConsent()
1.30 confirmRPConnection()
relyingParty
1.31 createUpdateLink()
1.32 tdifRPAuthoriseResponse()

1.0 - accessRelyingPartyService() : (Customer ==> Relying Party)

Description:
Customer Requesting to Access Relying Service.

1.1 - www_authorise() : (Relying Party ==> ISAM WebSeal)

Description: OIDC
Relying party issues an authorize request with the appropriate scope and claims

Request:
```
https://<<idhub>>/authorise?response_type=code&client_id=rp&scope=openid profile email phone
tdif_business_authorisations&redirect_uri=<<hostname>>/rp/exchange&nonce=<<nonce>>&state=<<state>>&ac
r_values=urn:id.gov.au:tdif:acr:ip2:cl2&claims={"id_token":{"mygov_linked":{"essential":true}},"useri
nfo":{"mygov_linked":{"essential":true}}}
```

Response:
HTTP/1.1 200 OK

1.2 - loadExchangeSPA() : (ISAM WebSeal ==> WebServer)

Description:
Load Exchange SPA page

1.3 - displayIDPSelectionPage() : (Exchange SPA ==> Exchange SPA)

Description: UI action
Display the IDP Selection page to the user

1.4 - customerSelectsIDP() : (Customer ==> Exchange SPA)

Description: User action
Customer selects the preferred IDP from the list

1.5 - checkmyGovFlowRequired() : (Exchange SPA ==> Exchange SPA)

check if {"mygov_linked":{"essential":true}} is present in the Relying Party Authorisation request

1.6 - kickOffLinkableIDPAuthoriseFlow() : (Exchange SPA ==> ISAM AAC)

Description: API
Kickoff the IDP authorize flow

Security:
Bearer-Token: **ISAM_LEVEL1_JWT**

Request:
POST: /sso/sps/oidc/rp/IDENTITYHUB/kickoff/MYGOVIDP?acr_values=<<acr_values>>

Response:
HTTP/1.1 200 OK

BODY:
```
{"location":"/mga/sps/oauth/oauth20/authorize?scope=openid+email+link
 &state=<<state>>
 &client_id=rp_exchange_client
 &nonce=<<nonce>>
&redirect_uri=/sso/sps/oidc/rp/IDENTITYHUB/redirect/MYGOVIDP
 &acr_values=<<acr_values>>
 &response_type=code"}
```

1.7 - tdifIDPAuthReq(tdif_request) : (ISAM AAC ==> Identity Provider)

Description: API
Send TDIF OIDC Authorize request to Selected IDP as a front channel redirect.

Security:
ISAM_LEVEL1_JWT

Request:
GET:
```
/proto-idp/oauth/authorize?claims=<<claims>>&state=<<state>>&client_id=PROTOIDP&nonce=<<nonce>>&redir
ect_uri=https://<<idhub>>/sso/sps/oidc/rp/IDENTITYHUB/redirect/PROTOIDP&acr_values=<<acr_values>>&res
ponse_type=code
```

Response:
HTTP/1.1 302 Found

1.8 - authenticateUser() : (Identity Provider ==> Identity Provider)

Description:
Authenticate the IDP User.

1.9 - tdifIDPAuthResponse() : (Identity Provider ==> ISAM AAC)

Description:
TDIF IDP Authorisation response

Security:
ISAM_LEVEL1_JWT

Request:
GET: /sso/sps/oidc/rp/IDENTITYHUB/redirect/PROTOIDP?code=<<code>>&state=<<state>>

Response:
HTTP/1.1 302
Redirect from IDP

1.10 - tdifIDPTokenRequest(client_assertion_type, code, grant_type, redirect_uri, state, client_assertion, client_id) : (ISAM AAC ==> Identity Provider)

Description:
Modify this to receive the new Claim EDI.

Request:
REST API Request URL : <<idhub>>/idp/oauth/token
Request Method : POST
Request params:

```
{client_assertion_type=<<client-assertion-type>>&code=APb2br&grant_type=authorization_code&redirect_u
ri=<<redirect_uri>>&state=2ilqGhrrGH&client_assertion=<<client_assertion>>&client_id=IDP}
```

Response:

```
{"access_token":"<<access_token>>","token_type":"bearer","expires_in":43199,"scope":"email openid
phone tdif_core","id_token":"<<id_token>>"}
```

1.11 - tdifIDPUserInfoRequest() : (ISAM AAC ==> Identity Provider)

Description: API
Retrieve IDP User information

Request:

Security:
Header: Bearer Token

Response:
HTTP/1.1 200 OK

1.12 - addIDPClaims() : (ISAM AAC ==> EXT-Issuer API)

Description: API
Store the new IDP (EDI) claims in the database

Security:
Header: ex-ext-iss-api-key: <<**ISAM_EXCHANGE_JWT**>>

Request:
PUT: <<exchange>>/api-ext-issuer/v1/identityproviders/<<idp_client_id>>/authclaims

BODY:

```
{"claims":{"sub":"<<sub>>",
 "aud":"PROTOIDP",
 "acr":"urn:id.gov.au:tdif:acr:ip2:cl2",
 "auth_time":1564557530,
 "kid":"1",
 "iss":"https://<<idhub>>/proto-idp",
 "exp":<<exp>>,
 "iat":<<iat>>,
 "nonce":"<<nonce>>",
 "jti":"<<jti>>",
 "phone_number":"000",
 "family_name":"Mayweather",
 "email_verified":true,
 "phone_number_verified":true,
 "updateAt":1564557531302,
 "email":"test@email.com",
 "given_name":"Mike",
 "birthdate":"<<birthdate>>"}}
```

Response:
HTTP/1.1 200 OK

1.13 - searchCustomer : (EXT-Issuer API ==> EXT-Issuer API)

1.14 - calculateRPKey : (EXT-Issuer API ==> EXT-Issuer API)

1.15 - searchCustomer : (EXT-Issuer API ==> EXT-Issuer API)

1.16 - calculateRPKey : (EXT-Issuer API ==> EXT-Issuer API)

1.17 - promptForConsent() : (ISAM AAC ==> Exchange SPA)

1.18 - promptForConsent : (Exchange SPA ==> Customer)

Redirect to SPA to display the consent page to share details with myGov

1.19 - customerConsents() : (Customer ==> Exchange SPA)

Description: User Action
Customer provides consent to share details with myGov

1.20 - storeRPConsent() : (Exchange SPA ==> EXT-Customer API)
Description: API
Store Relying Party Consent

Security:
Bearer-Token: **ISAM_LEVEL3_JWT**

Request:
PUT: <<idhub>>/api-ext-customer-ui/v1/rpauthentications/<<rp_auth_id>>/consent

BODY:
```
{"id":"<<rp_auth_id>>",
 "claimConsents":[],
 "abn":"<<abn>>",
 "triggerScope":<<triggerScope>>}
```

Response:
HTTP/1.1 201 Created

1.21 - confirmmyGovConnection : (Exchange SPA ==> Exchange)
Description: Internal API

Verify and confirm if a link exists between myGov and Exchange for the given user

1.22 - generateRPLink() : (Exchange ==> Exchange)
Generate new RP Pairwise Identifier using the relevant sector

1.23 - saveRPLinkandKey() : (Exchange ==> Exchange)
Save the generated RP Pairwise ID and key for myGov

1.24 - createUpdateLink(rpLinkID) : (Exchange ==> myGov Integration)
Description: API
Create a new myGov Relying Party Account Link / MBUN in the myGov system for a given relying party. The user account will be specified by the use of the "sub", "squ" and "sty" claims of the mgv-jwt JWT header

Security:
Bearer-Token: **EXCHANGE_MYGOV_JWT**

Request:
POST (Create): /accounts/links/

BODY: RelyingPartyLink object
```
{
 "relyingPartyId": "ATO|EXCHGE|...",
 "relyingPartyName": "Australian Taxation Office,
 "relyingPartyLinkDetails": {
   "id": "<<id>>",
   "status": "permanent|transient",
   "created": "<<created>>",
   "lastModified": "<<lastModified>>"
 }
}
```

Response:
HTTP/1.1 201 Relying party link successfully created

1.25 - createUpdateProfile() : (Exchange ==> myGov Integration)
Description: API
Create or Update User's myGov profile in myGov

Security:
Bearer-Token: **EXCHANGE_MYGOV_JWT**

Request:
POST (Create): /accounts/profile
PUT (Update): /accounts/profile

BODY: myGov Profile
```
{ "name": {
   "firstName": "string",
   "middleName": "string",
   "lastName": "string"
 },
 "dateOfBirth": "2019-11-05"
}
```

Response:
HTTP/1.1 204 Successfully Created or Updated

1.26 - promptForConsent() : (Exchange ==> Exchange SPA)
Redirect to SPA to display the consent page to share details with the relying party

1.27 - promptForConsent : (Exchange SPA ==> Customer)

Redirect to SPA to display the consent page to share details with the relying party

1.28 - customerConsents() : (Customer ==> Exchange SPA)

Description: User Action
Customer provides consent to share details with the relying party

1.29 - storeRPConsent() : (Exchange SPA ==> EXT-Customer API)

Description: API
Store Relying Party Consent

Security:
Bearer-Token: **ISAM_LEVEL3_JWT**

Request:
PUT: <<idhub>>/api-ext-customer-ui/v1/rpauthentications/<<rp_auth_id>>/consent

BODY:

```
{"id":"<<rp_auth_id>>",
 "claimConsents":[],
 "abn":"<<abn>>",
 "triggerScope":<<triggerScope>>}
```

Response:
HTTP/1.1 201 Created

1.30 - confirmRPConnection : (Exchange SPA ==> Exchange)

Description: Internal API

Verify and confirm if a link exists between the Relying Party and Exchange for the given user

1.31 - createUpdateLink(rpLinkID) : (Exchange ==> myGov Integration)

Description: API
Create a new myGov Relying Party Account Link / MBUN in the myGov system for a given relying party. The user account will be specified by the use of the "sub", "squ" and "sty" claims of the mgv-jwt JWT header

Security:
Bearer-Token: **EXCHANGE_MYGOV_JWT**

Request:
POST (Create): /accounts/links/

BODY: RelyingPartyLink object

```
{
 "relyingPartyId": "ATO|EXCHGE|...",
 "relyingPartyName": "Australian Taxation Office,
 "relyingPartyLinkDetails": {
   "id": "<<id>>",
   "status": "permanent|transient",
   "created": "<<created>>",
   "lastModified": "<<lastModified>>"
 }
}
```

Response:
HTTP/1.1 201 Relying party link successfully created

1.32 - tdifRPAuthoriseResponse() : (ISAM AAC ==> Relying Party)

Send an authorized response back to the Relying Party along with the RP claims