CIBA Prototype Implementation Practical Guide

For keycloak FAPI-SIG

Oct 2020

PROPOSED DRAFT

This document describes the CIBA prototype implementation(https://github.com/tnorimat/keycloak/tree/ciba-prototype-v1.0).

This prototype has been developed on April 2020 to study to which extend keycloak can support CIBA.

After this prototype had been developed, the design document for CIBA support has been written, reviewed and accepted by keycloak-community (https://github.com/keycloak/keycloak-community/blob/master/design/client-initiated-backchannel-authentication-flow.md) .

Therefore, please note that this prototype does not completely comply with this design document.

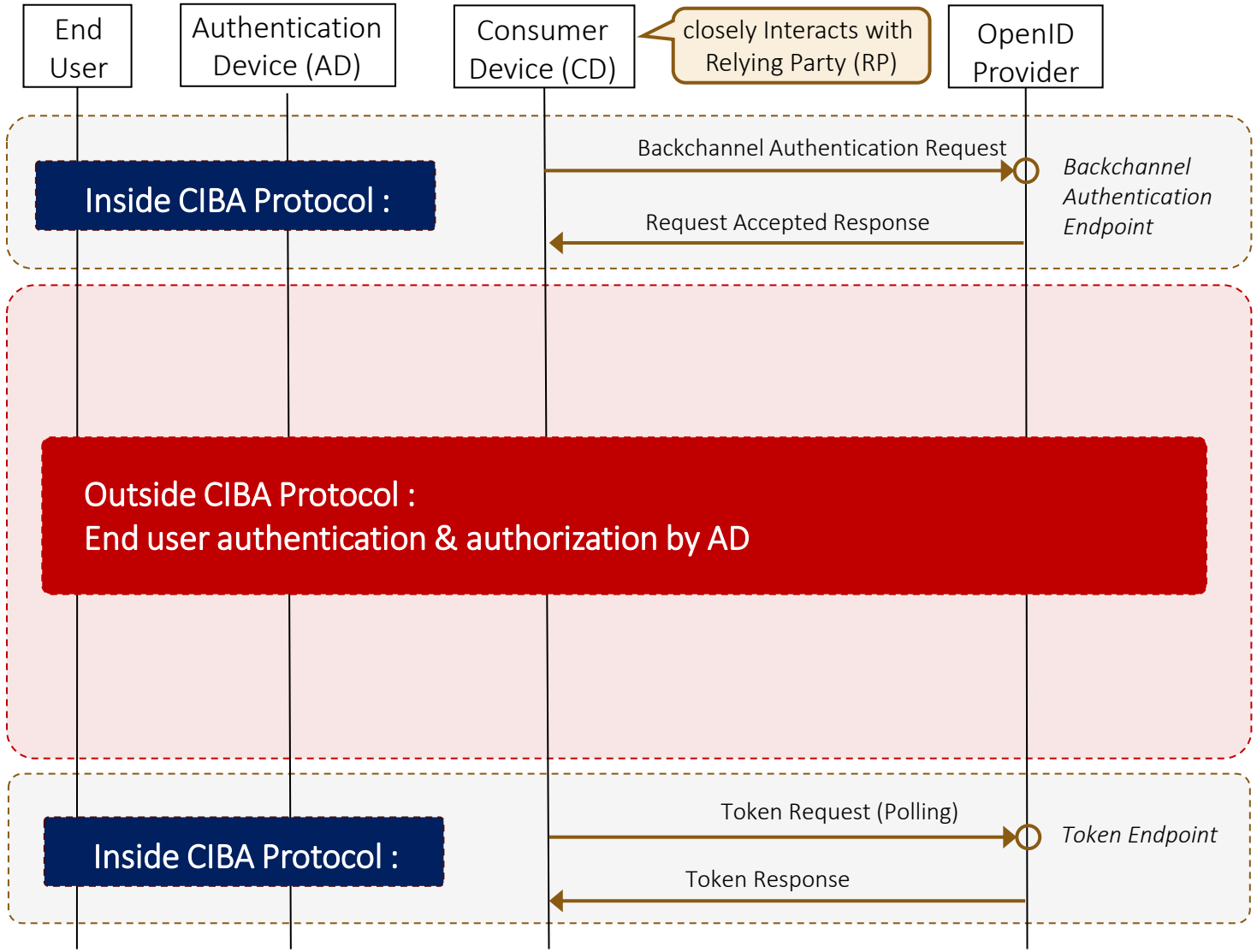
To contribute CIBA support to keycloak, we need to completely make this prototype comply with this design document. FAPI-SIG will work with this task.

Preface

Table of Contents

Overview

CIBA Flow : Protocol specified part only



CIBA specification does not specify how to do end user Authentication(AuthN) & Authorization(AuthZ) by AD.

To implement CIBA flow, we also need to specify this part (Outside CIBA Protocol) and implement it.

To do so, it's important that developer can realize their own AuthN & AuthZ by AD by implementing them as providers, which means that developer does not need to modify codes of the body of keycloak.

Considering that point, I've at first defined the interface to do end user AuthN & AuthZ by AD which is not dependent on the specific way of it. Also, I've prepared its reference implementation.

PROPOSED DRAFT

CIBA Flow : Interfaces on non-protocol specified part

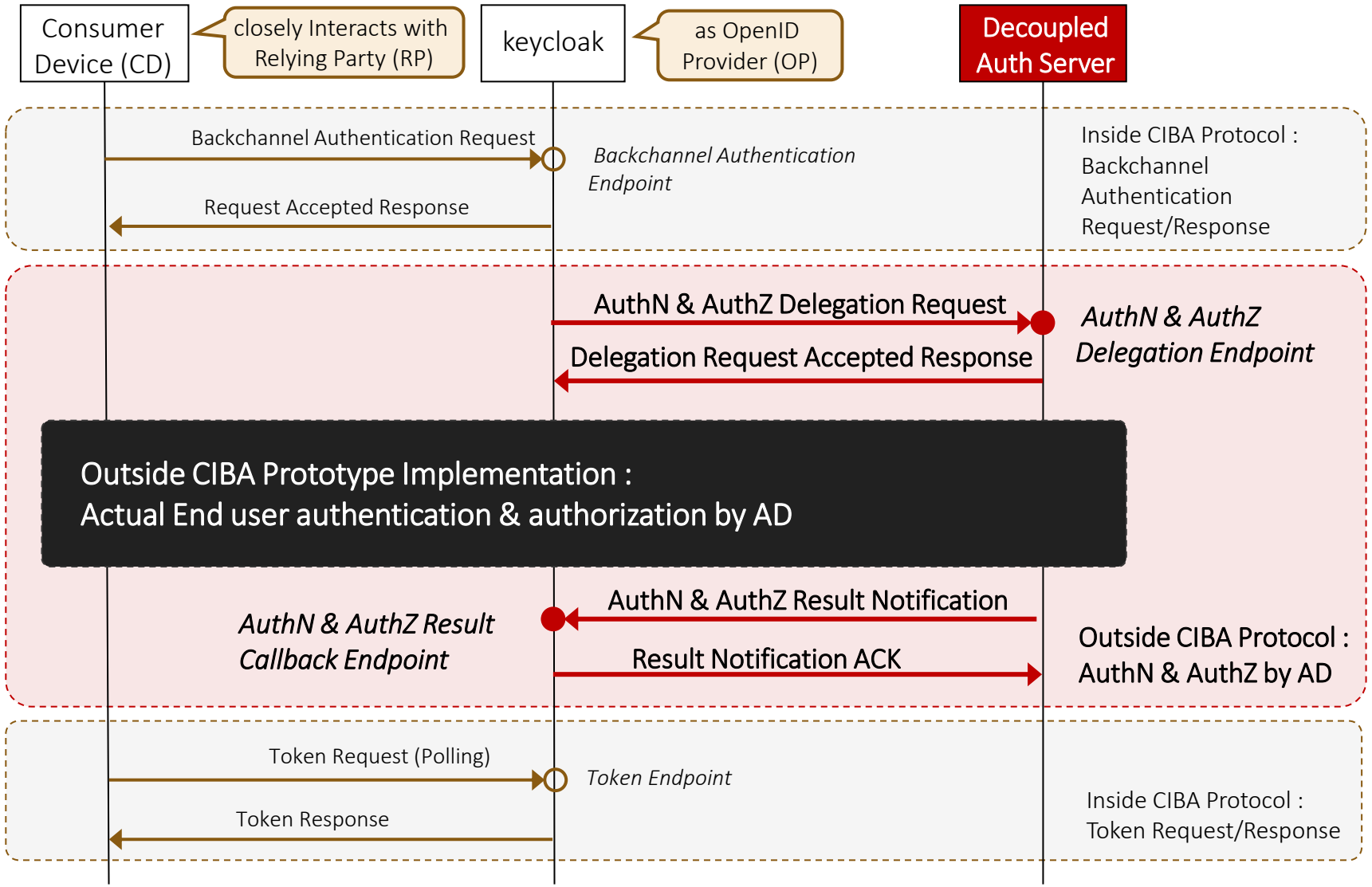
Keycloak itself does not do an end user AuthN & AuthZ. It is delegated to the other entity called “Decoupled Auth Server”.

This CIBA prototype does not treat any kind of the specific way of AuthN & AuthZ AD. It is up to actual implementation of Decoupled Auth Serer.

This CIBA prototype only define the interfaces between keycloak and Decoupled Auth Server.

However, to confirm this prototype works, the reference implementation of Decoupled Auth Server was prepared :

https://github.com/tnorimat/ciba-decoupled-authn-server



PROPOSED DRAFT

Entities

Entities in CIBA protocol sequence are the followings :

[Inside CIBA protocol]

- End User
- Authentication Device (AD)
- Consumer Device (CD)
- keycloak

[Outside CIBA protocol]

- Decoupled Auth Server

Entities

This CIBA prototype implementation provides the followings :

- keycloak
  - repository : tnorimat/keycloak
  - tag : ciba-prototype-v1.0
  - platform : Windows / RHEL
  - JDK version : 8
  - WildFly operating mode : standalone
  - Auth server on Arquillian integration test : undertow
- Decoupled Auth Server (Reference Implementation)
  - repository : tnorimat/ciba-decoupled-authn-server

It is only for test purpose. Not used for actual CIBA support to keycloak.

Scope

Scope

This document covers the followings :

- ■ Specification on Inside CIBA Protocol
- ■ Specification on Outside CIBA Protocol
- ■ Prerequisite of the prototype
- ■ Protocol sequence of CIBA Flow
- ■ Endpoint specification of CIBA Flow
- ■ HTTP Request/Response specification of CIBA Flow

Especially, this document does not cover the followings :

- □ Abnormal sequences of CIBA Flow
- □ Performance
- □ Security
- □ Usability

Functional Specification

Functional Specification

[Inside CIBA Protocol]

- Backchannel Authentication Request
  - Conveyance of end-user to be authenticated : login_hint
  - Value of login_hint : username in keycloak
  - Supported parameters : scope, binding_message
- Token Request
  - Mode : poll
  - Expiration of auth_req_id : supported (by expires_in)
  - Request throttling : supported (by interval)

Functional Specification

[Outside CIBA Protocol]

- Authentication by Authentication Device (AD)
  - The way of an authentication : Delegating to the server called Decoupled Auth Server
  - The way of an authentication request from keycloak : asynchronous
  - Conveyance of end-user to be authenticated : username in keycloak
  - Authorization : supported (whether it is required or not is notified from keycloak)
- Supported features by issued tokens
  - Token Refresh
  - Token Introspection
  - Token Revocation
  - User Info Request
  - Logout

Prerequisite

Prerequisite

[User]

Users authenticated by AD must be registered on keycloak in advance.

[Decoupled Auth Server]

Decoupled Auth Server must be registered on keycloak as a confidential client in advance.

[CD(Client)]

CD must be registered on keycloak as a confidential client in advance.

This CIBA prototype does not provider the software running as Client.

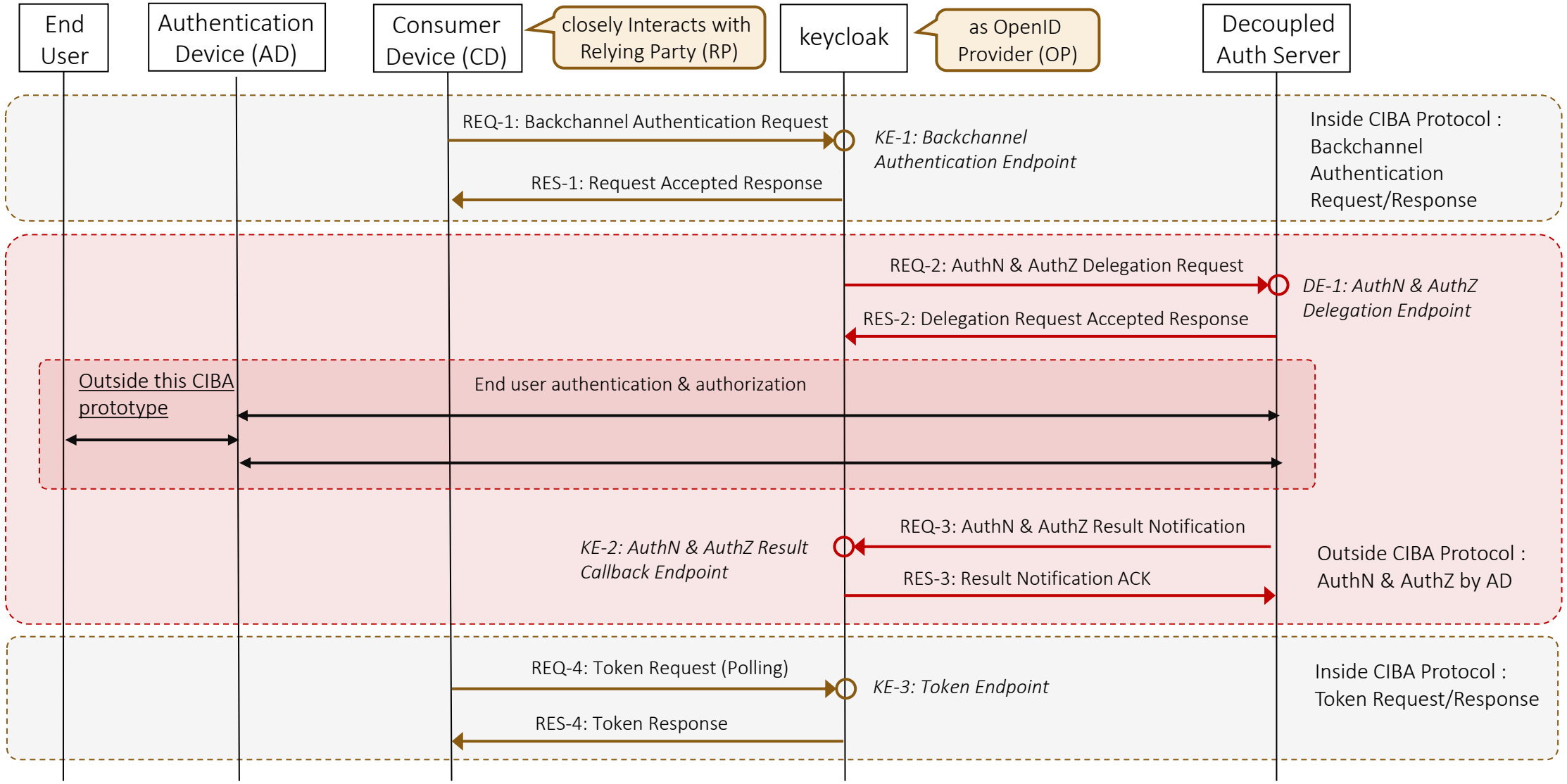
Please prepare it by yourself. (e.g. curl, postman)

[AD]

If using Decoupled Auth Server Reference Implementation provided by this CIBA prototype, AD is not required. You can control the result of AuthN & AuthZ by AD on this reference implementation.

Interface Specification

CIBA Flow : Endpoints and Messages



PROPOSED DRAFT

Sequence Overview

CIBA protocol sequence consists the following 2 parts:

- Inside CIBA Protocol (defined by CIBA specification)
  - Backchannel Authentication Request/Response
  - Token Request/Response
- Outside CIBA Protocol (NOT defined by CIBA specification)
  - AuthN & AuthZ by AD Request/Response

The order of running these part in CIBA protocol sequence is as follows :

1. Inside CIBA Protocol : Backchannel Authentication Request/Response
2. Outside CIBA Protocol : AuthN & AuthZ by AD Request/Response
3. Inside CIBA Protocol : Token Request/Response

Endpoints

Endpoints in CIBA protocol are the followings :

- On keycloak
  - KE-1: Backchannel Authentication Endpoint

    CD sends a backchannel authentication request to it.

    This Endpoint is defined by CIBA specification.
  - KE-2: AuthN & AuthZ Result Callback Endpoint

    Decoupled Auth Server sends the result of AuthN & AuthZ by AD to it.

    This Endpoint is NOT defined by CIBA specification.
  - KE-3: Token Endpoint

    CD sends a token request to it.

    This Endpoint is defined by OAuth2 specification.
- On Decoupled Auth Server
  - DE-1: AuthN & AuthZ Delegation Endpoint

    Keycloak sends AuthN & AuthZ by AD delegation request to it.

    This Endpoint is NOT defined by CIBA specification.

KE-1: Backchannel Authentication Endpoint

[Overview]

Keycloak plays a role as HTTP Server while CD as HTTP Client.

CD sends a backchannel authentication request to keycloak.

Keycloak returns auth_req_id that identifies the corresponding CIBA flow.

CD uses it for token request afterwards.

If keycloak returns an abnormal response, the corresponding CIBA is aborted.

<URI>

http(s)://{host}:{port}/auth/realms/{realm}/protocol
/openid-connect/backchannelAuthn

<Authentication>

Required (Basic Authentication with client_id and client_secret as default)

REQ-1: Backchannel Authentication Request

<Method> : POST

<Content-Type> : application/x-www-form-urlencoded

<Parameters>

login_hint : REQUIRED

It identifies the end user for AuthN and AuthZ by AD.

Its value must be “username” of the user registered in keycloak.

scope : REQUIRED

“scope” parameter defined by OAuth2 specification.

binding_message : OPTIONAL

Its value is intended to be shown in both CD and AD’s UI.

RES-1: Request Accepted Response

[Normal Case]

<Status Code>

200 OK

<Content-Type>

application/json

<Parameters>

auth_req_id : REQUIRED

It identifies the CIBA flow. It can be used for token request.

expires_in : REQUIRED

It expresses the expiration time in sec for auth_req_id.

interval : OPTIONAL

It shows the interval for which CD needs to wait for token request.

RES-1: Request Accepted Response

[Abnormal Case 1]

<Case> : CD's client authentication failed.

<Status Code> : 401 Unauthorized

<Content-Type> : application/json

<Entities>

error : "unauthorized_client"

error_description : "invalid client secret"

[Abnormal Case 2]

<Case> : CD is not registered as a confidential client.

<Status Code> : 400 Bad Request

<Content-Type> : application/json

<Entities>

error : "unauthorized_client"

error_description : "INVALID_CREDENTIALS: Invalid client credentials"

RES-1: Request Accepted Response

[Abnormal Case 3]

<Case> : CD is registered as a confidential client but deactivated.

<Status Code> : 400 Bad Request

<Content-Type> : application/json

<Entities>

error : “unauthorized_client”

error_description : “Invalid client credentials”

[Abnormal Case 4]

<Case> : Required parameter “scope” is missing.

<Status Code> : 400 Bad Request

<Content-Type> : application/json

<Entities>

error : “invalid_request”

error_description : “missing parameter : scope”

RES-1: Request Accepted Response

[Abnormal Case 5]

<Case> : Required parameter “login_hint” is missing.

<Status Code> : 400 Bad Request

<Content-Type> : application/json

<Entities>

error : “invalid_request”

error_description : “missing parameter : login_hint”

[Abnormal Case 6]

<Case> : The user specified by “login_hint” does not exist.

<Status Code> : 400 Bad Request

<Content-Type> : application/json

<Entities>

error : “unknown_user_id”

error_description : “no user found”

RES-1: Request Accepted Response

[Abnormal Case 7]

<Case> : The user specified by “login_hint” is deactivated.

<Status Code> : 400 Bad Request

<Content-Type> : application/json

<Entities>

error : “unknown_user_id”

error_description : “user deactivated”

[Abnormal Case 8]

<Case> : Something unexpected error happened.

<Status Code> : 500 Internal Server Error

DE-1: AuthN & AuthZ Delegation Endpoint

[Overview]

Decoupled Auth Server plays a role as HTTP Server while keycloak as HTTP Client.

keycloak sends an AuthN & AuthZ delegation request to Decoupled Auth Server.

Decoupled Auth Server returns decoupled_auth_id to identify the context of AuthN & AuthZ by AD.

This endpoint is not defined by CIBA specification.

If keycloak returns an abnormal response, the corresponding CIBA is aborted.

<URI>

http(s)://{host}:{port}/request-decoupled-authentication

<Authentication>

Nothing

REQ-2: AuthN & AuthZ Delegation Request

<Method> : POST

<Content-Type> : application/x-www-form-urlencoded

<Parameters>

decoupled_auth_id : REQUIRED

It identifies the context of AuthN & AuthZ by AD in Decoupled Auth Server.

user_info : REQUIRED

It identifies the end user for AuthN and AuthZ by AD.

Its value must be “username” of the user registered in keycloak.

scope : REQUIRED

“scope” parameter defined by OAuth2 specification.

is_consent_required : REQUIRED

It shows whether Decoupled Auth Server needs to get consent from the end user about scope.

default_client_scope : OPTIONAL

This scopes are the ones that Decoupled Auth Server needs to get consent from the end user.

binding_message : OPTIONAL

Its value is intended to be shown in both CD and AD’s UI.

RES-2: Delegation Request Accepted Response

[Normal Case]

<Status Code> : 200 OK

[Abnormal Case 1]

<Case> : Invalid input

<Status Code> : 400 Bad Request

[Abnormal Case 2]

<Case> : Something unexpected error happened in Decoupled Auth Server

<Status Code> : 500 Internal Server Error

KE-2: AuthN & AuthZ Result Callback Endpoint

[Overview]

keycloak plays a role as HTTP Server while Decoupled Auth Server as HTTP Client. Decoupled Auth Server sends the result of AuthN & AuthZ by AD to keycloak with decoupled_auth_id to identify the context of AuthN & AuthZ by AD.

This endpoint is not defined by CIBA specification.

If keycloak returns an abnormal response, the corresponding CIBA flow is aborted.

<URI>

http(s)://{host}:{port}/auth/realms/{realm}/protocol

/openid-connect/ext/ciba-decoupled-authn-callback

<Authentication>

Required (Basic Authentication with client_id and client_secret as default)

REQ-3: AuthN & AuthZ Result Notification

<Method> : POST

<Content-Type> : application/x-www-form-urlencoded

<Parameters>

decoupled_auth_id : REQUIRED

It identifies the context of AuthN and AuthZ by AD.

Its value must be “username” of the user registered in keycloak.

user_info : REQUIRED

It identifies the end user for AuthN and AuthZ by AD.

Its value must be “username” of the user registered in keycloak.

auth_result : REQUIRED

The result of AuthN and AuthZ by AD identified by decoupled_authid for the user identified by user_info

succeeded : Both AuthN and AuthZ have succeeded. (only AuthN if AuthZ is not required)

unauthorized : AuthN has succeeded but AuthZ has been denied.

cancelled : AuthN have been cancelled.

failed : AuthN have failed.

RES-3: Result Notification ACK

[Normal Case]

<Status Code> : 200 OK

[Abnormal Case 1]

<Case> : decoupled_auth_id format is invalid.

<Status Code> : 400 Bad Request

[Abnormal Case 2]

<Case> : decoupled_auth_id has already been used.

<Status Code> : 400 Bad Request

[Abnormal Case 2]

<Case> : decoupled_auth_id has not yet been issued.

<Status Code> : 400 Bad Request

RES-3: Result Notification ACK

[Abnormal Case 4]

<Case> : decoupled_auth_id has already expired.

<Status Code> : 400 Bad Request

[Abnormal Case 5]

<Case> : Something unexpected error happened in keycloak.

<Status Code> : 500 Internal Server Error

KE-3: Token Endpoint

[Overview]

Keycloak plays a role as HTTP Server while CD as HTTP Client.

CD sends a token request to keycloak with auth_req_id that identifies the corresponding CIBA flow.

If keycloak returns an abnormal response, the corresponding CIBA is aborted.

<URI>

http(s)://{host}:{port}/auth/realms/{realm}/protocol/openid-connect/token

<Authentication>

Required (Basic Authentication with client_id and client_secret as default)

REQ-4: Token Request (Polling)

<Method> : POST

<Content-Type> : application/x-www-form-urlencoded

<Parameters>

grant_type : REQUIRED

It must be “urn:openid:params:grant-type:ciba”.

auth_req_id : REQUIRED

It identifies the CIBA flow.

RES-4: Token Response

[Normal Case]

<Status Code> : 200 OK

<Content-Type> : application/json

<Parameters>

access_token: REQUIRED

Access token defined by OAuth2 specification.

expires_in : REQUIRED

Access token's expiration time defined by OAuth2 specification.

token_type : REQUIRED

Token type defined by OAuth2 specification. Its value is "bearer".

scope : OPTIONAL

Scope defined by OAuth2 specification.

refresh_token : OPTIONAL

Refresh token defined by OAuth2 specification.

refresh_expires_in : OPTIONAL

Refresh token's expiration time.

id_token : OPTIONAL

ID token defined by OIDC specification.

RES-4: Token Response

[Abnormal Case 1]

<Case> : CD's client authentication failed.

<Status Code> : 401 Unauthorized

<Content-Type> : application/json

<Entity>

error : "unauthorized_client"

error_description : "invalid client secret"

[Abnormal Case 2]

<Case> : auth_req_id is missing.

<Status Code> : 400 Bad Request

<Content-Type> : application/json

<Entity>

error : "invalid_request"

error_description : "Missing parameter: auth_req_id"

RES-4: Token Response

[Abnormal Case 3]

<Case> : auth_req_id format is invalid.

<Status Code> : 400 Bad Request

<Content-Type> : application/json

<Entity>

error : “invalid_grant”

error_description : “Invalid Auth Req ID”

[Abnormal Case 4]

<Case> : auth_req_id has already been used.

<Status Code> : 400 Bad Request

<Content-Type> : application/json

<Entity>

error : “invalid_grant”

error_description : “Invalid Auth Req ID”

RES-4: Token Response

[Abnormal Case 5]

<Case> : auth_req_id has not yet been issued.

<Status Code> : 400 Bad Request

<Content-Type> : application/json

<Entity>

error : “invalid_grant”

error_description : “Invalid Auth Req ID”

[Abnormal Case 6]

<Case> : auth_req_id has already expired.

<Status Code> : 400 Bad Request

<Content-Type> : application/json

<Entity>

error : “expired_token”

error_description : “Auth Req ID has expired.”

RES-4: Token Response

[Abnormal Case 7]

<Case> : CD send a request without waiting for the time specified by the parameter “interval”.

<Status Code> : 400 Bad Request

<Content-Type> : application/json

<Entity>

error : “slow_down”

error_description : “Too early to access.”

Notes : add +5 sec to the interval as the penalty for too much early access.

[Abnormal Case 8]

<Case> : AuthN & AuthZ by AD has not yet been completed.

<Status Code> : 400 Bad Request

<Content-Type> : application/json

<Entity>

error : “authorization_pending”

error_description : “The authorization request is still pending as the end-user hasn’t yet been authenticated.”

RES-4: Token Response

[Abnormal Case 9]

<Case> : AuthN & AuthZ by AD has been time out.

<Status Code> : 400 Bad Request

<Content-Type> : application/json

<Entity>

error : “access_denied”

error_description : “authentication timed out.”

[Abnormal Case 10]

<Case> : AuthN & AuthZ by AD has failed.

<Status Code> : 400 Bad Request

<Content-Type> : application/json

<Entity>

error : “access_denied”

error_description : “authentication failed.”

RES-4: Token Response

[Abnormal Case 11]

<Case> : AuthN & AuthZ by AD has been cancelled.

<Status Code> : 400 Bad Request

<Content-Type> : application/json

<Entity>

error : “access_denied”

error_description : “authentication cancelled.”

[Abnormal Case 12]

<Case> : AuthZ by AD has been denied.

<Status Code> : 400 Bad Request

<Content-Type> : application/json

<Entity>

error : “access_denied”

error_description : “not authorized.”

RES-4: Token Response

[Abnormal Case 13]

<Case> : Unexpected error happened on AuthN & AuthZ by AD.

<Status Code> : 400 Bad Request

<Content-Type> : application/json

<Entity>

error : “invalid_grant”

error_description : “unknown authentication result.”

[Abnormal Case 14]

<Case> : AuthN & AuthZ by AD has been succeeded but the creation of corresponding user session failed.

<Status Code> : 400 Bad Request

<Content-Type> : application/json

<Entity>

error : “invalid_grant”

error_description : “user session not found.”

RES-4: Token Response

[Abnormal Case 15]

<Case> : Different user that CD does not required to be authenticated has been authenticated by AD.

<Status Code> : 400 Bad Request

<Content-Type> : application/json

<Entity>

error : “invalid_grant”

error_description : “different user authenticated.”

[Abnormal Case 16]

<Case> : Different CD that keycloak did not send auth_req_id sends a token request.

<Status Code> : 400 Bad Request

<Content-Type> : application/json

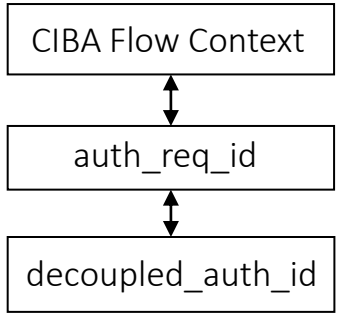
<Entity>

error : “invalid_grant”

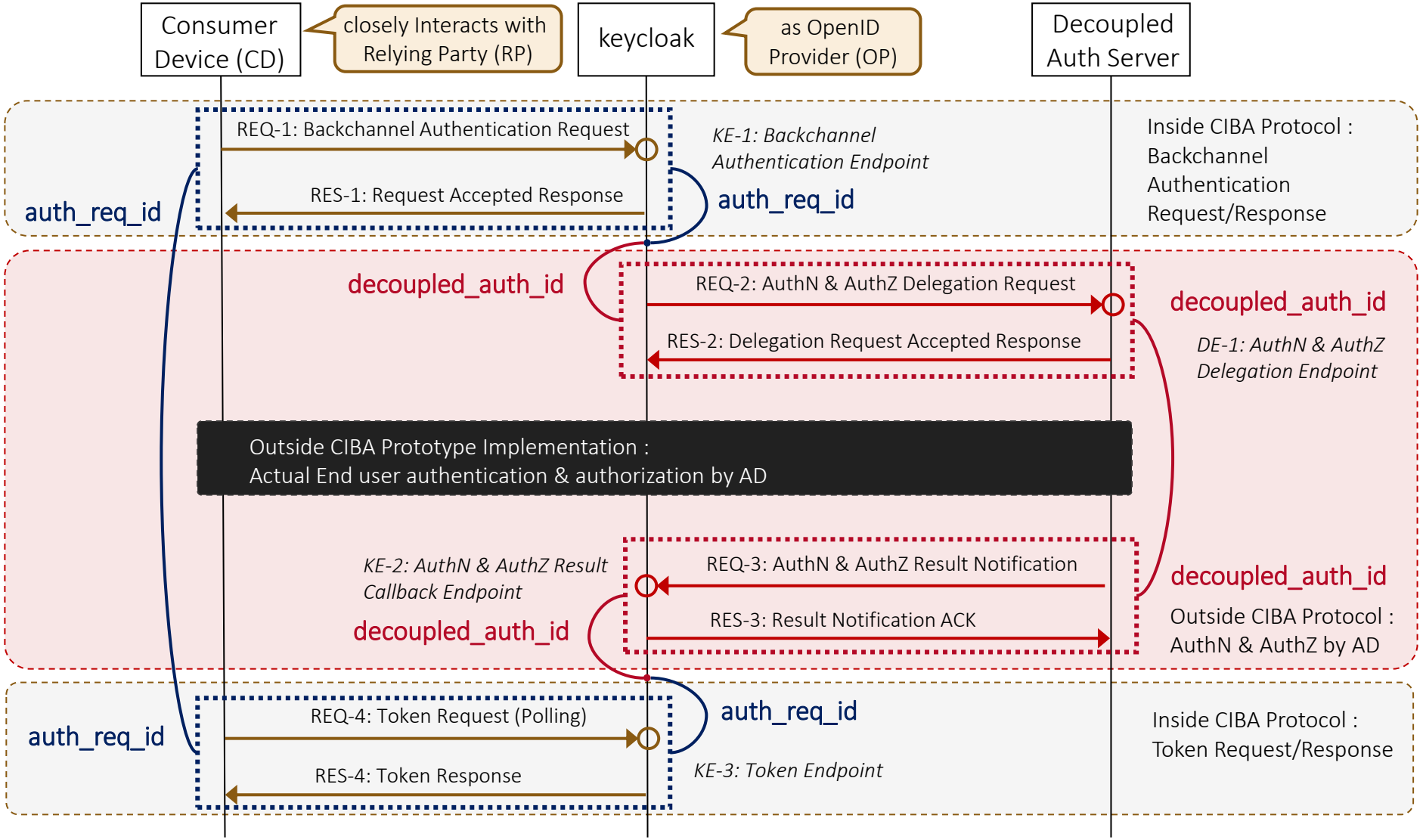
error_description : “unauthorized client.”

Internals

CIBA Flow : Session Binding

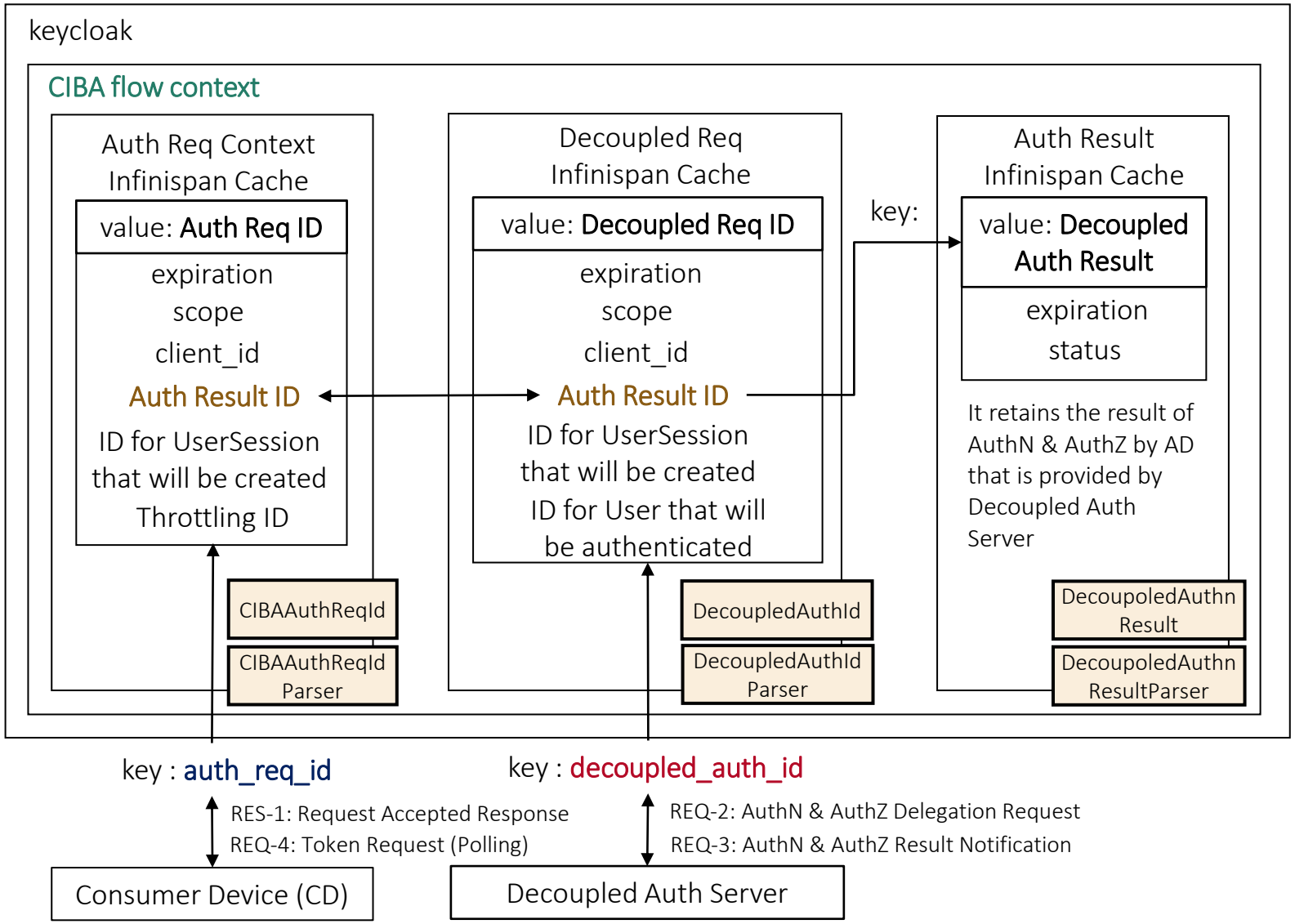


To identify the CIBA flow between keycloak and CD, auth_req_id is used.
To identity the CIBA flow between keycloak and Decoupled Auth Server, decoupled_auth_id is introduced.
Keycloak retains the relationship between auth_req_id and decoupled_auth_id.



PROPOSED DRAFT

CIBA Flow : Context



In keycloak, CIBA flow context consists of Auth Req ID, Decoupled Req ID and Decoupled Auth Result. These can be bound with Auth Result ID.

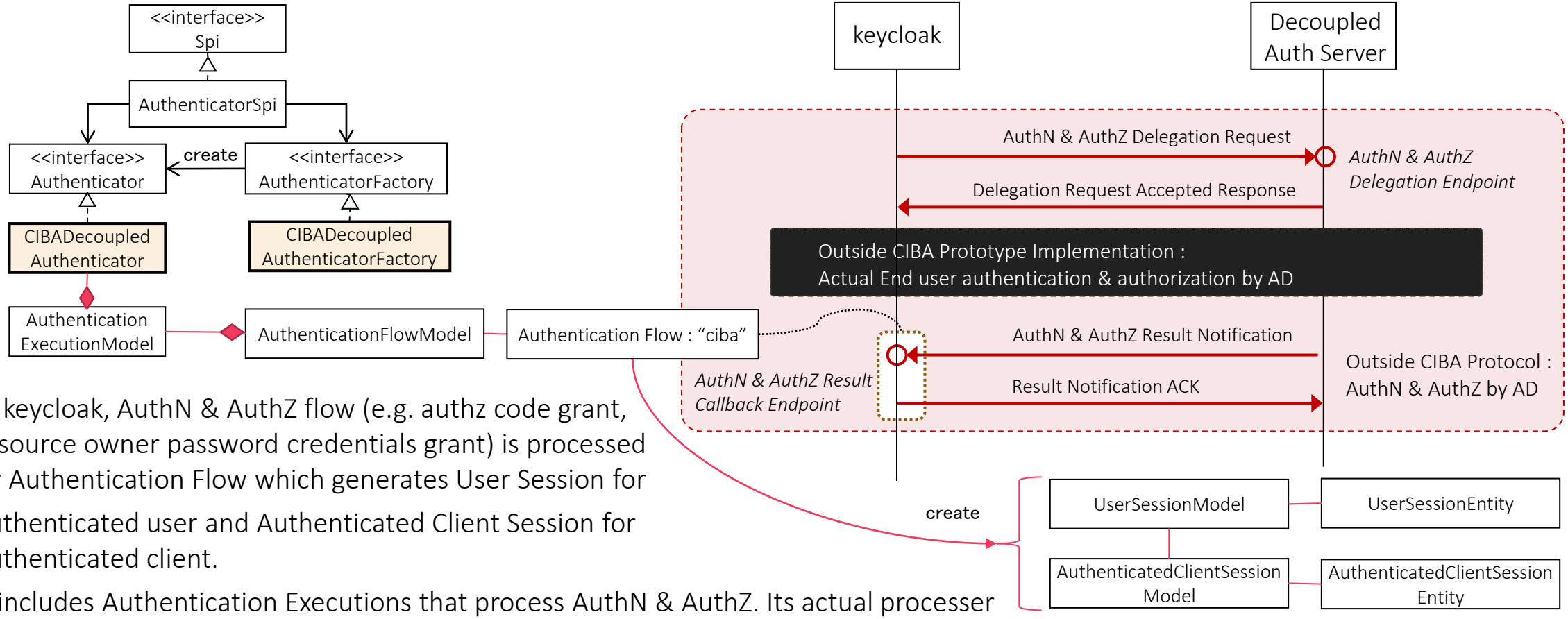
These three items are stored on Infinispan Cache for existing Action Tokens.

Between keycloak and CD, CIBA flow context can be bound with auth_req_id. It is defined by CIBA standard specification.

Between keycloak and Decoupled Auth Server, CIBA flow context can be bound with decoupled_auth_id.

PROPOSED DRAFT

CIBA Flow : Authentication Flow/Execution



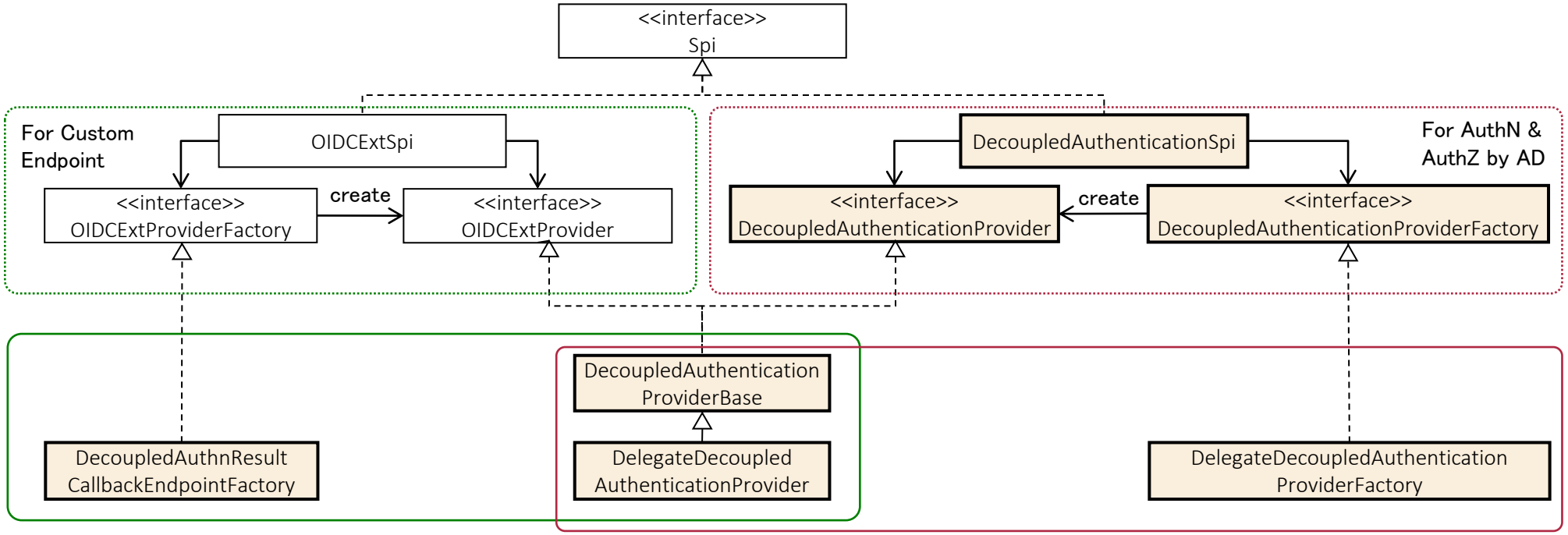
In keycloak, AuthN & AuthZ flow (e.g. authz code grant, resource owner password credentials grant) is processed by Authentication Flow which generates User Session for authenticated user and Authenticated Client Session for authenticated client.

It includes Authentication Executions that process AuthN & AuthZ. Its actual processer is Authenticator (e.g. password, OTP, webauthn) that this Authentication Execution holds.

For CIBA flow, the corresponding Authentication Flow is newly introduced. Also the corresponding Authenticator is also provided. This Authentication Flow for CIBA flow is invoked when keycloak receives the result of AuthN & AuthZ by AD. The AuthN & Auth Z has already been completed so that corresponding Authenticator (CIBADecoupledAuthenticator) only relies on this result.

PROPOSED DRAFT

CIBA Flow : Providers

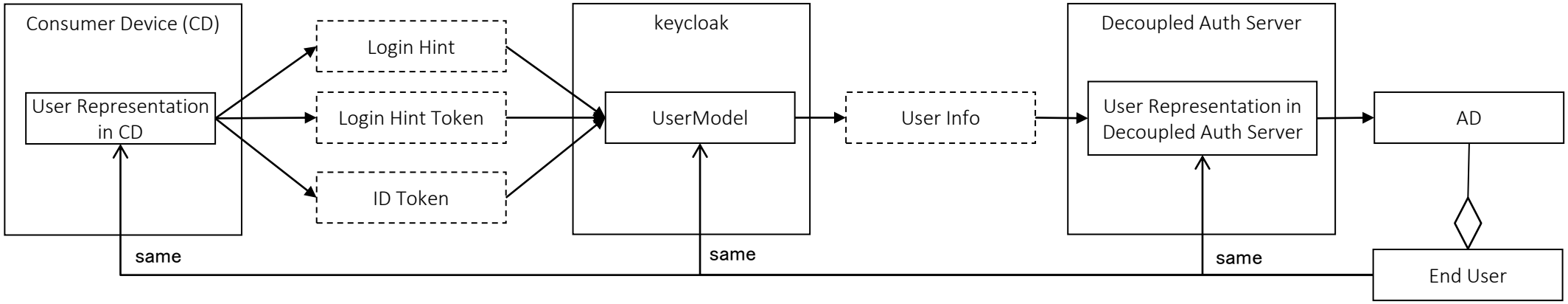


CIBA specification does not specify how to realize AuthN & AuthZ by AD. Therefore, this CIBA prototype implementation defines interfaces of this part and provides its reference implementation.

To realize this interfaces, existing one provider (OIDCExtProvider) is used for “AuthN & AuthZ Result Callback Endpoint”, and one provider (DecoupledAuthenticationProvider) is newly introduced for interacting with “AuthN & AuthZ Delegation Endpoint” of Decoupled Auth Server.

If you want to do your own AuthN & AuthZ by AD, you can implement it by using these two providers.

User Resolver

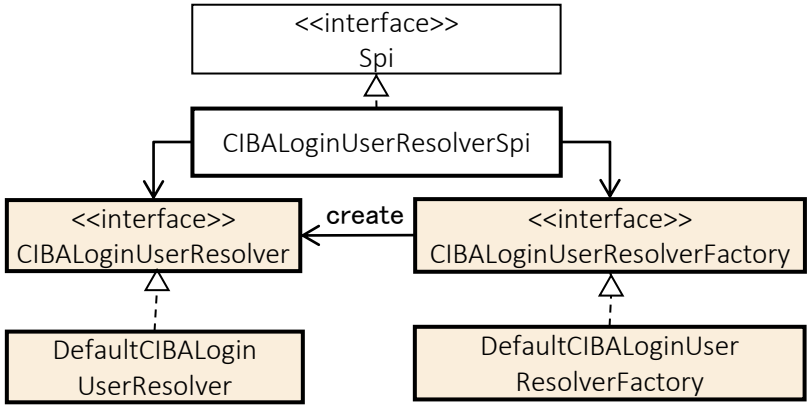


It is possible that each entities included in CIBA flow identifies the same user by the different way.

Also, it is possible that the conveyance of the information that is used for identifying the user the takes the different forms.

Considering these points, this CIBA prototype implementation provides “User Resolver”. It can convert each user representation and format used between CD and keycloak, keycloak and Decoupled Auth Server.

Developer can implement its own User Resolver.



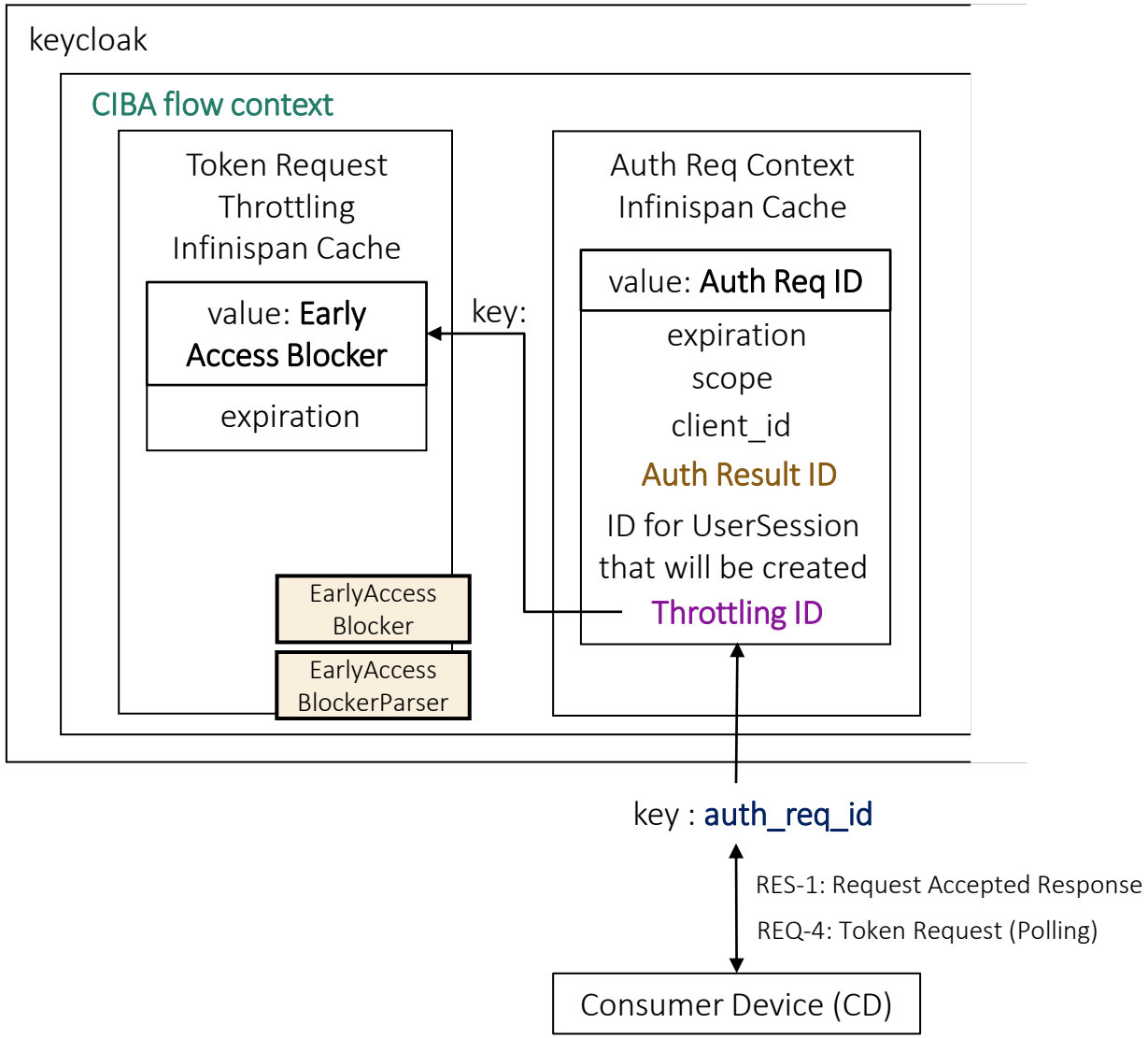
Token Request Throttling

According to CIBA specification, keycloak must not respond the token request from CD until the specified time passes to prevent CD from sending token request too much frequently.

To realize it, this CIBA prototype implementation provides Token Request Throttling cache to store the status showing whether the keycloak is allowed to respond the token request or not.

Its entry called Early Access Blocker has its expiration whose value is “interval” defined by CIBA specification. When receiving a token request from CD, the corresponding Early Access Blocker not expired means that this token request from CD is too much early.

In this case, this Early Access Blocker is re-created with its expiration being “interval” + penalty time.



Trial Run

Run by Arquillian Integration Test

To confirm that this CIBA prototype implementation works, run the corresponding Arquillian Integration Test (org.keycloak.testsuite.client.CIBATest)

```
> mvn -f testsuite/integration-arquillian/tests/base/pom.xml test
  -Dtest=org.keycloak.testsuite.client.CIBATest
```

Run with Decoupled Auth Server Reference Implementation

Add config for Decoupled Auth Server on the keycloak config file.

- AuthN & AuthZ Delegation Endpoint

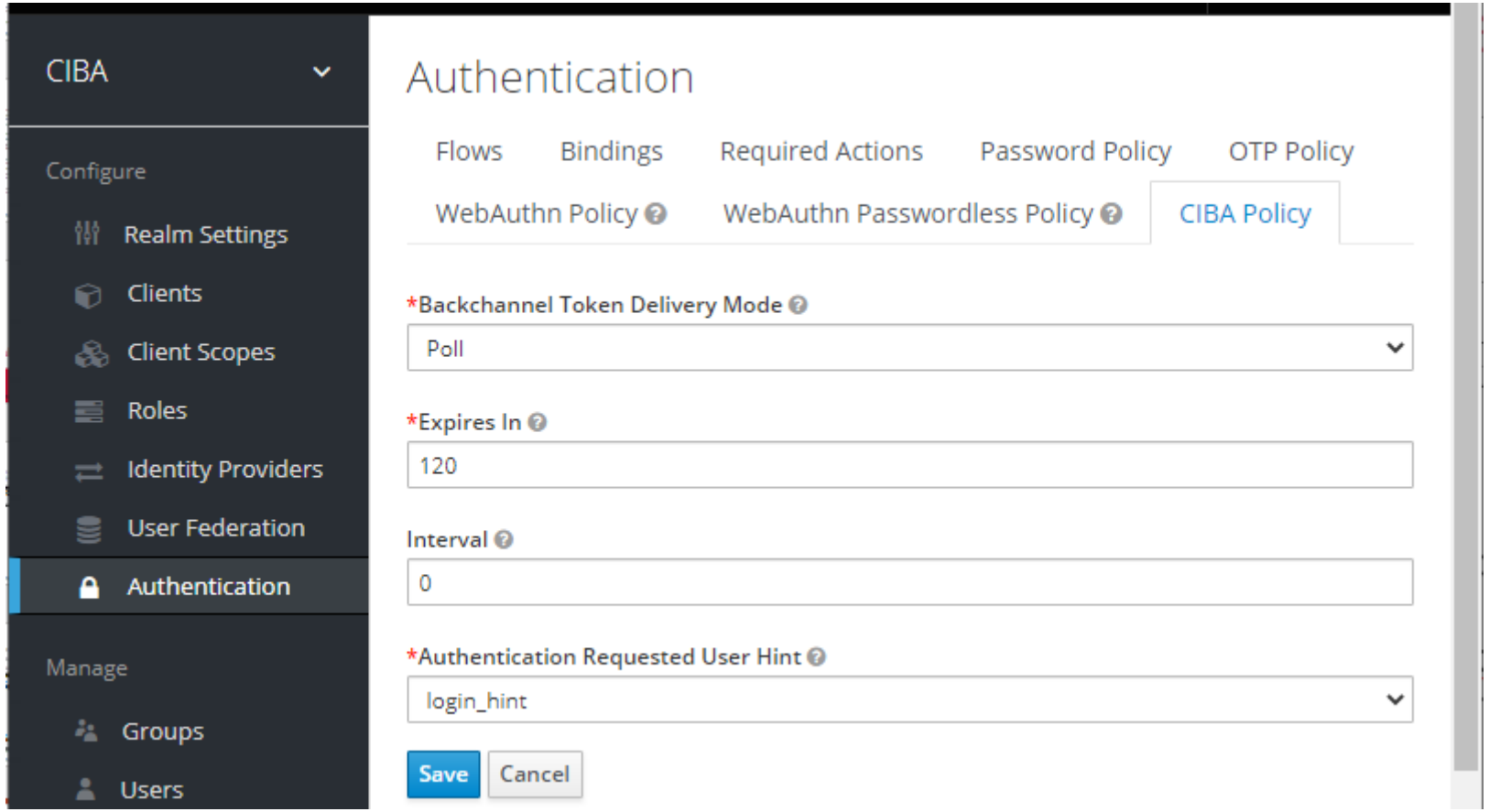
standalone.xml :

```
<subsystem xmlns="urn:jboss:domain:keycloak-server:1.1">
…
 <spi name="decoupled-authn">
            <default-provider>delegate-decoupled-authn</default-provider>
            <provider name="delegate-decoupled-authn" enabled="true">
               <properties>
                  <property name="decoupledAuthnRequestUri"
                              value="http://localhost:8888/request-decoupled-authentication"/>
               </properties>
            </provider>
         </spi>
…
</subsystem>
```

Here assumed that Decoupled Auth Server Reference Implementation run on localhost:8888

Run with Decoupled Auth Server Reference Implementation

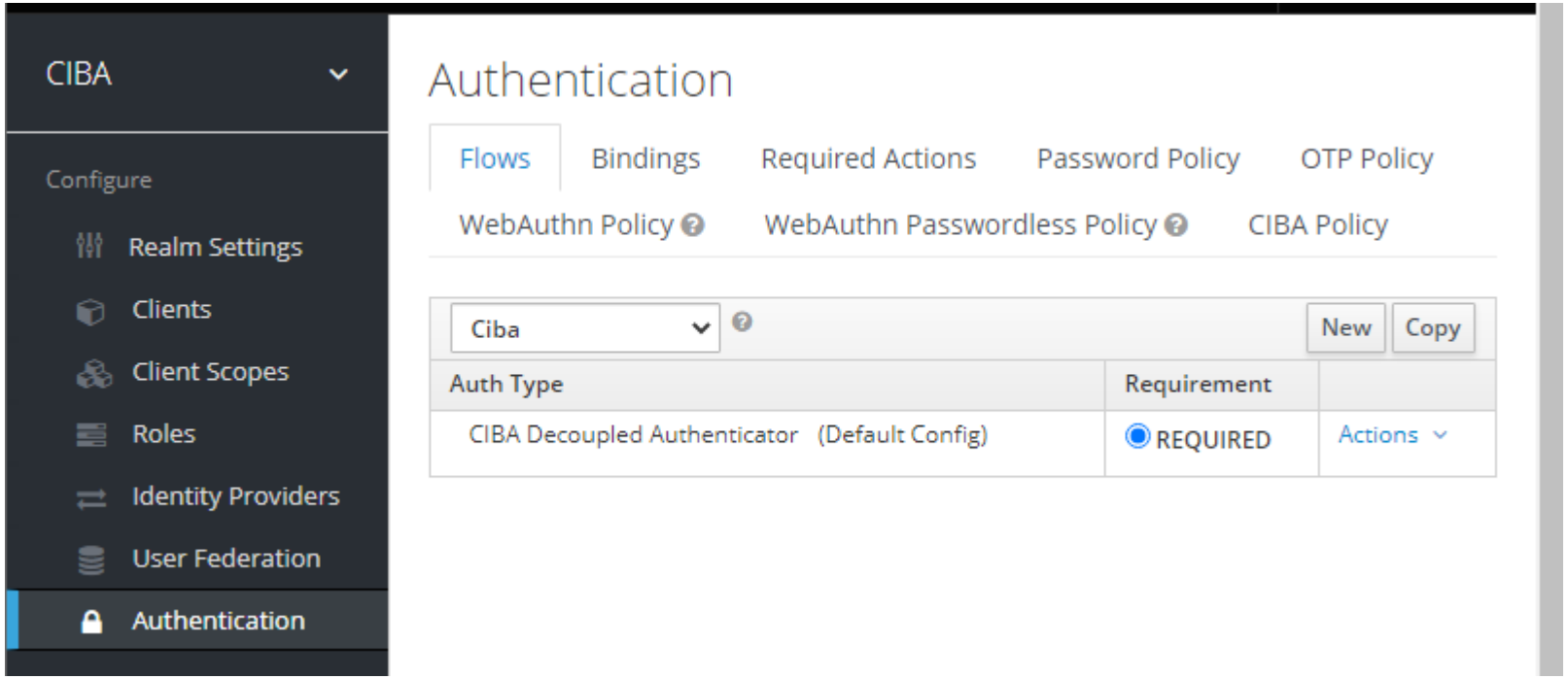
FYI : CIBA Settings - CIBA Policy



No need to modify this default settings.

Run with Decoupled Auth Server Reference Implementation

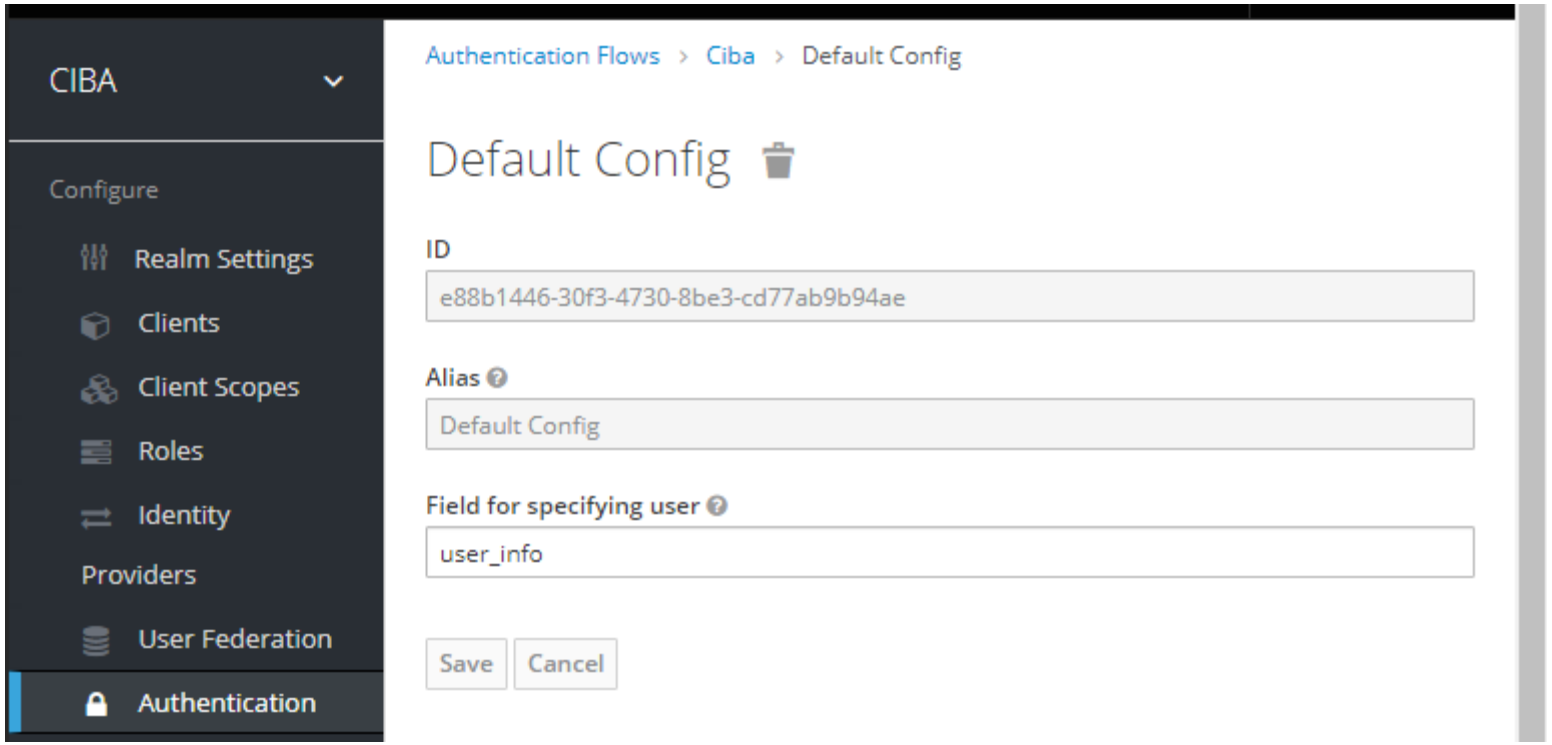
FYI : CIBA Settings - CIBA Flow



No need to modify this default settings.

Run with Decoupled Auth Server Reference Implementation

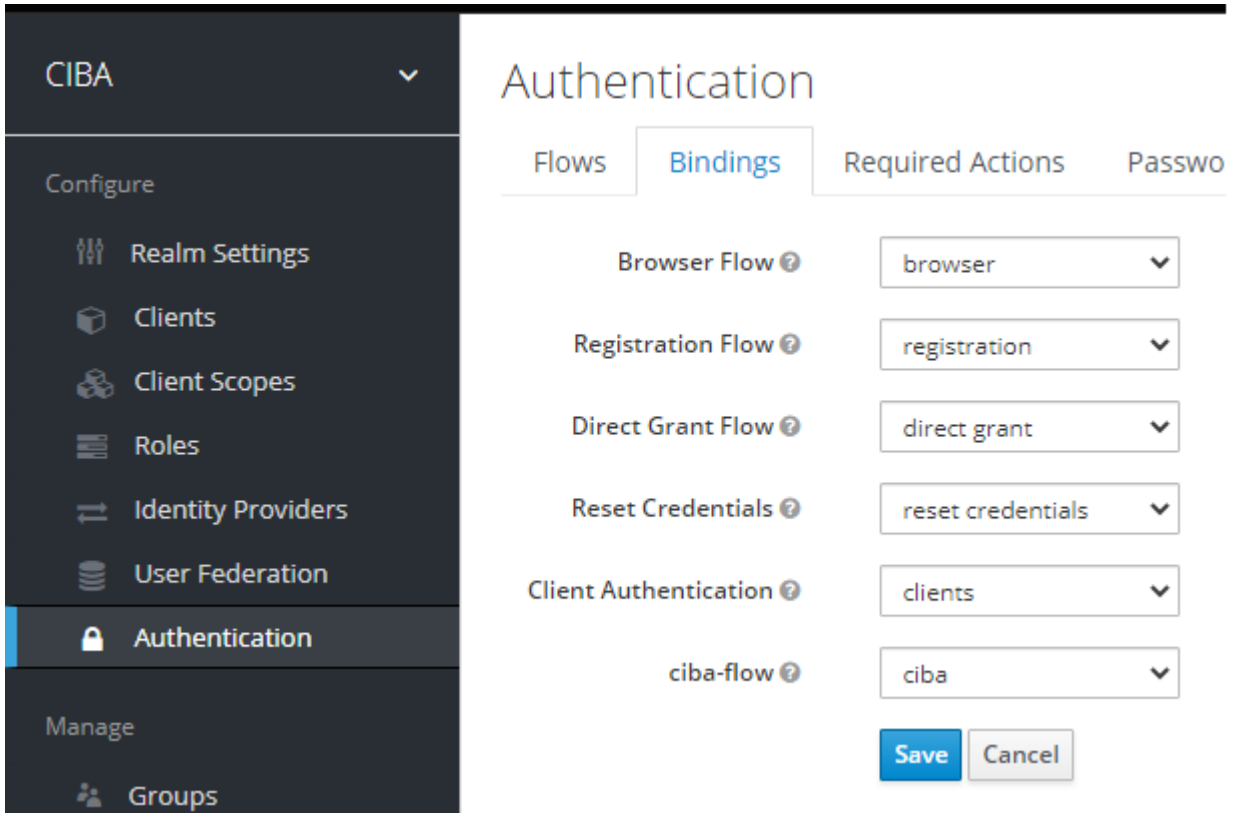
FYI : CIBA Settings - CIBA Flow - Authenticator’s Config



No need to modify this default settings.

Run with Decoupled Auth Server Reference Implementation

FYI : CIBA Settings - CIBA Flow - Flow Binding



No need to modify this default settings.

End