

For keycloak FAPI-SIG  
Jan 2021

# CIBA Implementation Practical Guide

This document describes the internals of Client Initiated Backchannel Authentication (CIBA<sup>\*1</sup>) support. This support has been developed based on its design document<sup>\*2</sup>.

The aim of this document is as follows.

- making it ease for the programmer to comprehend codes for CIBA support.

The target reader of this document is as follows.

- The reviewer who reviews codes for CIBA support.
- The programmer who want to make some CIBA support related contribution.

\*1 : [https://openid.net/specs/openid-client-initiated-backchannel-authentication-core-1\\_0.html](https://openid.net/specs/openid-client-initiated-backchannel-authentication-core-1_0.html)

\*2 : <https://github.com/keycloak/keycloak-community/blob/master/design/client-initiated-backchannel-authentication-flow.md>

# Preface

# Table of Contents

Overview

Scope

Functional Specification

Prerequisite

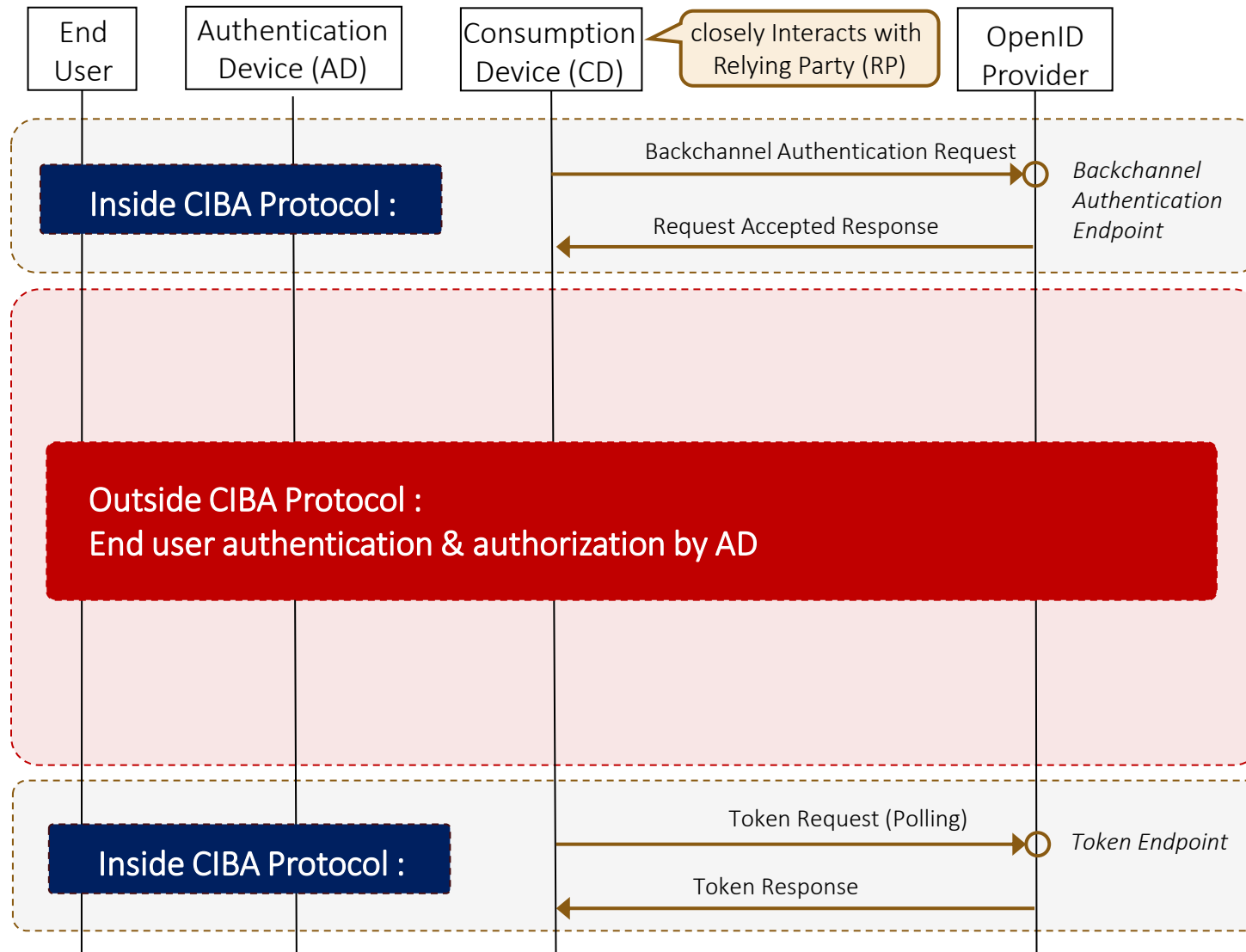
Interface Specification

Internals

Trial Run

# 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.

# CIBA Flow : Interfaces on non-protocol specified part

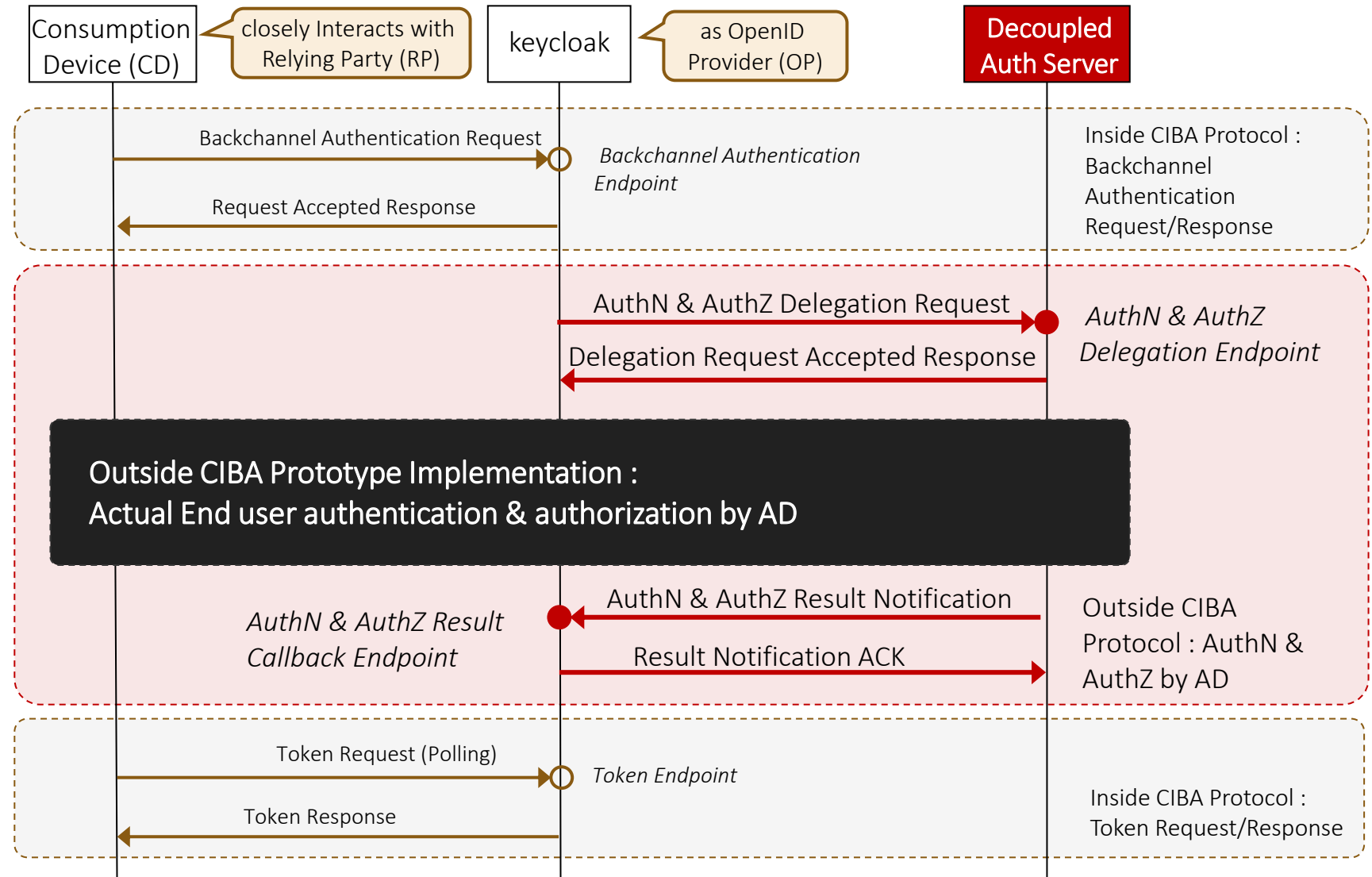
Keycloak itself does not do an end user AuthN & AuthZ. It is delegated to the other entity called “Decoupled Auth Server”.

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

This CIBA prototype only defines 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>



# Entities

Entities in CIBA flow are the followings :

[Inside CIBA protocol]

- End User
- Authentication Device (AD)
- Consumption Device (CD)
- keycloak (OpenID Provider)

[Outside CIBA protocol]

- Decoupled Auth Server

# Scope



# Scope

This document covers the followings :

- Specification on Inside CIBA Protocol
- Specification on Outside CIBA Protocol
- Endpoint specification of CIBA Flow
- HTTP Request/Response specification of CIBA Flow

Especially, this document does not cover the followings :

- Performance
- 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 it 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 provide the software running as Client.

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

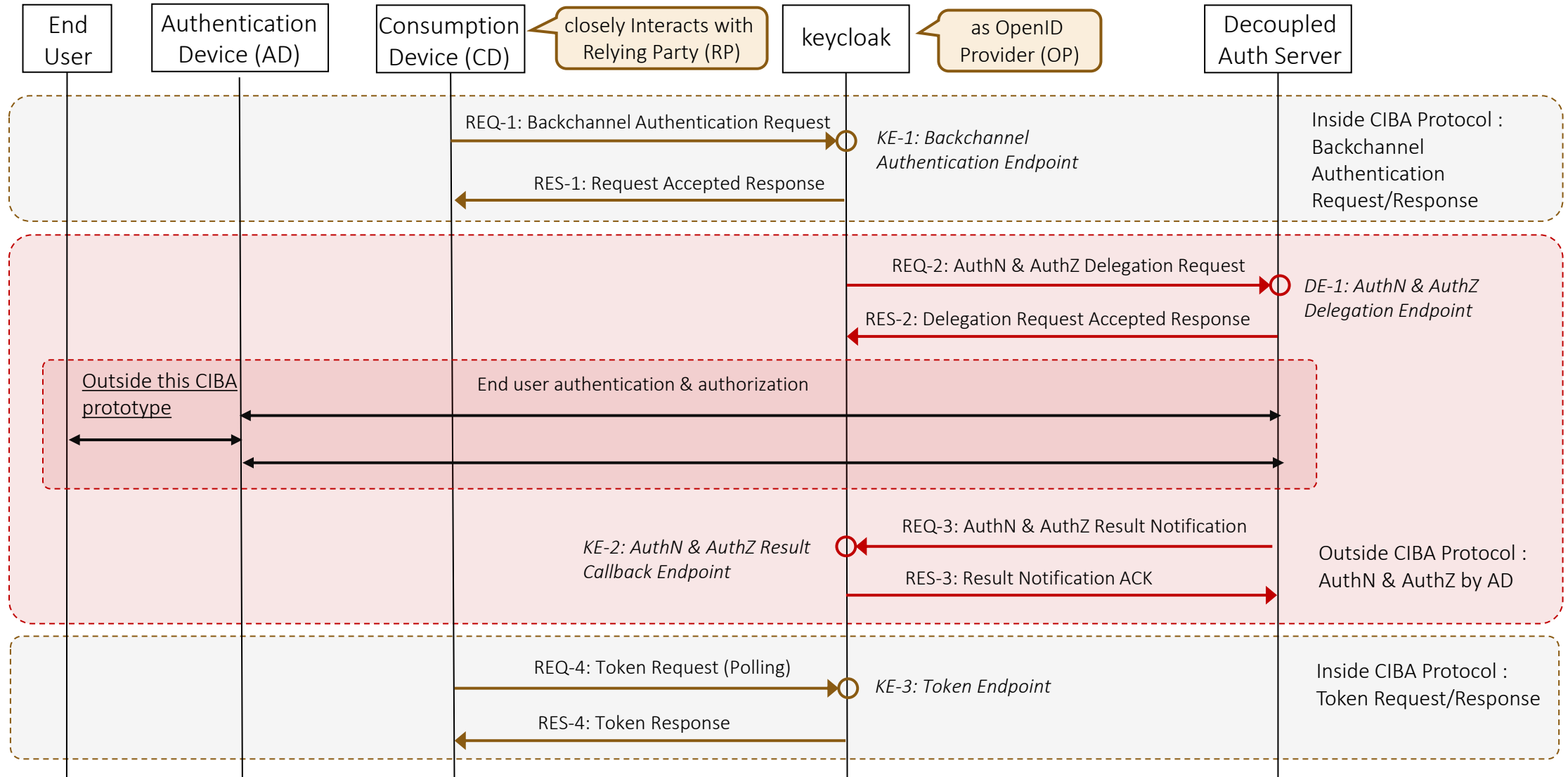
## [AD]

If using Decoupled Auth Server Reference Implementation<sup>\*1</sup>, AD is not required. You can control the result of AuthN & AuthZ by AD on this reference implementation.

\*1 : <https://github.com/tnorimat/ciba-decoupled-authn-server>

# Interface Specification

# CIBA Flow : Endpoints and Messages





# CIBA Flow Overview

CIBA flow 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 flow 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” or “email” 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

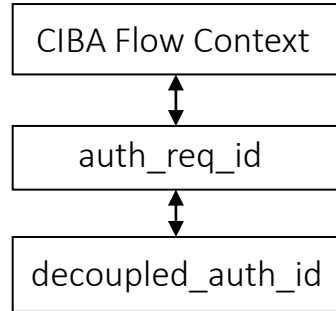
<Entity>

error : "invalid\_grant"

error\_description : "unauthorized client."

# Internals

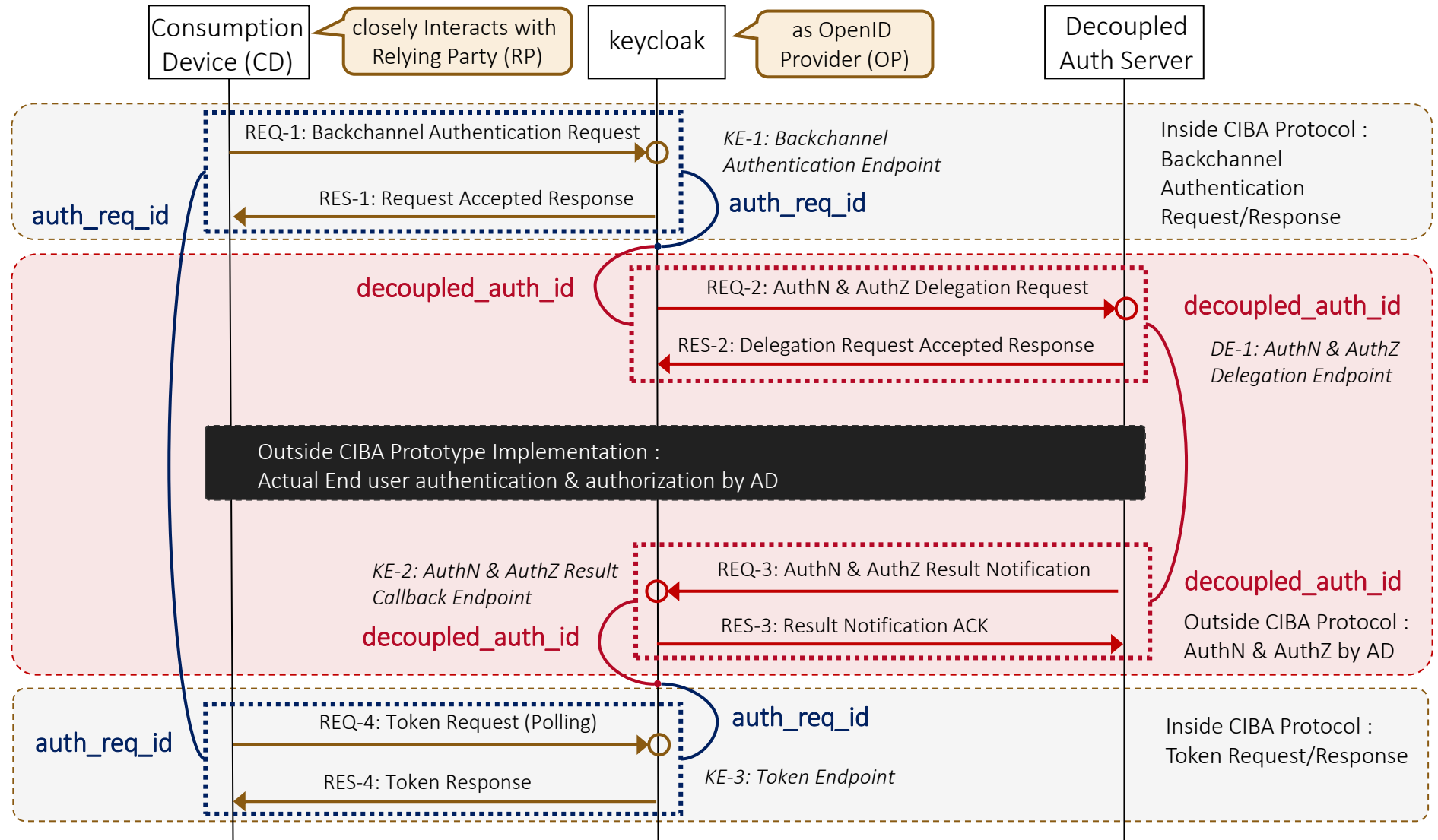
# CIBA Flow : Session Binding



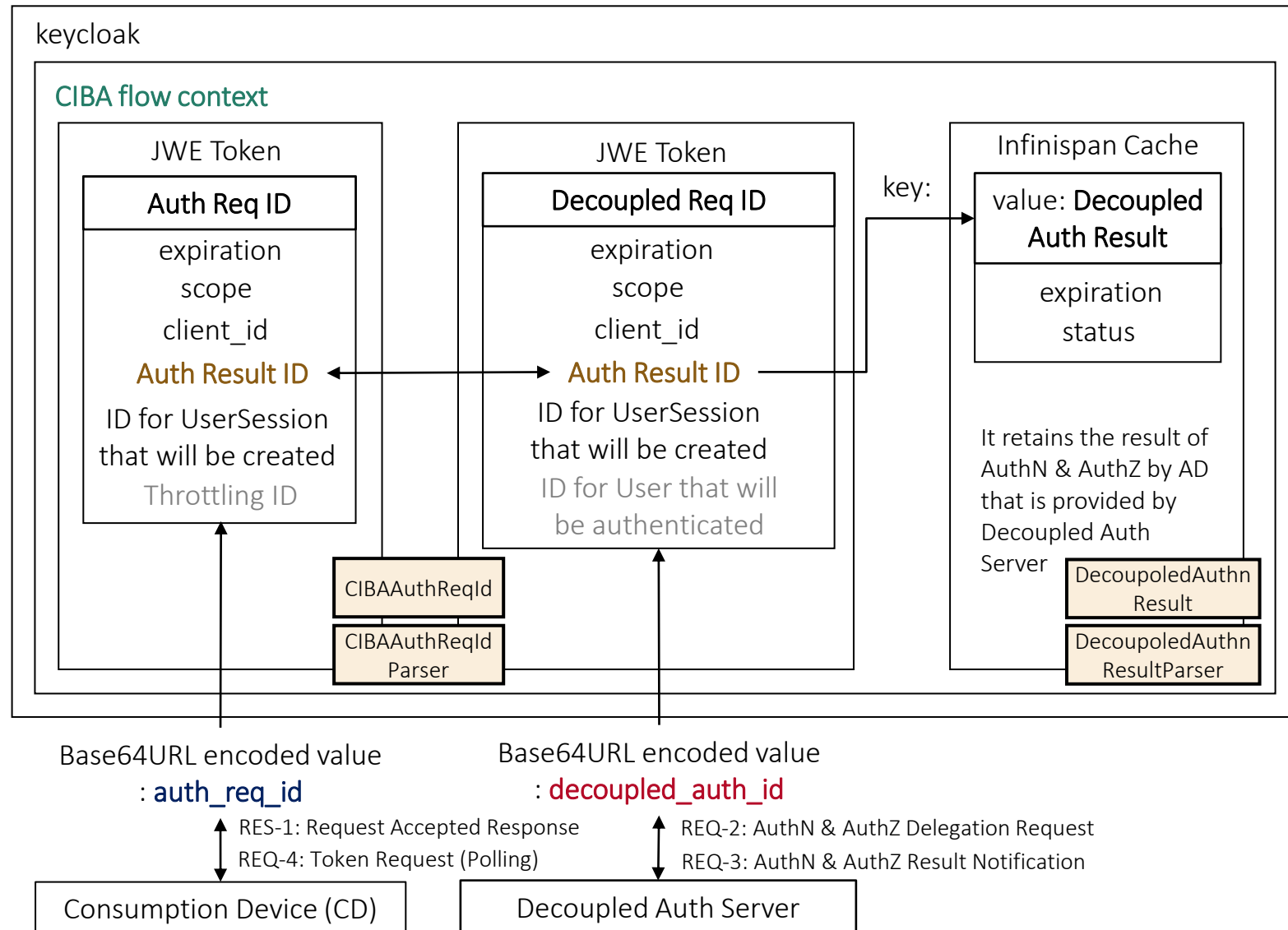
To identify the CIBA flow between keycloak and CD, **auth\_req\_id** is used.

To identify 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**.



# CIBA Flow : Context



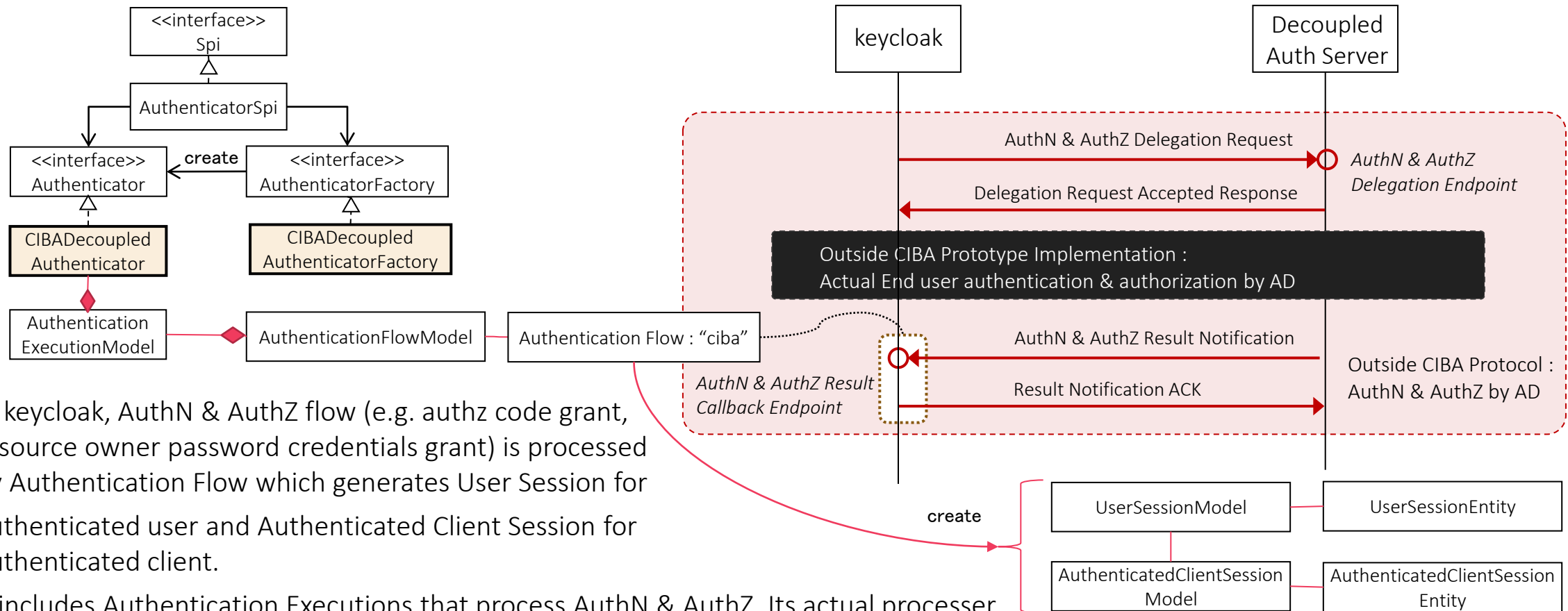
In keycloak, **CIBA flow context** consists of Auth Req ID, Decoupled Req ID and Decoupled Auth Result. These can be bound with **Auth Result ID**.

The former two are represented as JWE token while the latter is stored on Infinispan Cache for existing Action Tokens. All parameters needed to be retained in this CIBA flow are enclosed and encrypted onto JWT token which makes keycloak stateless about this CIBA flow.

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**.

# CIBA Flow : Authentication Flow/Execution

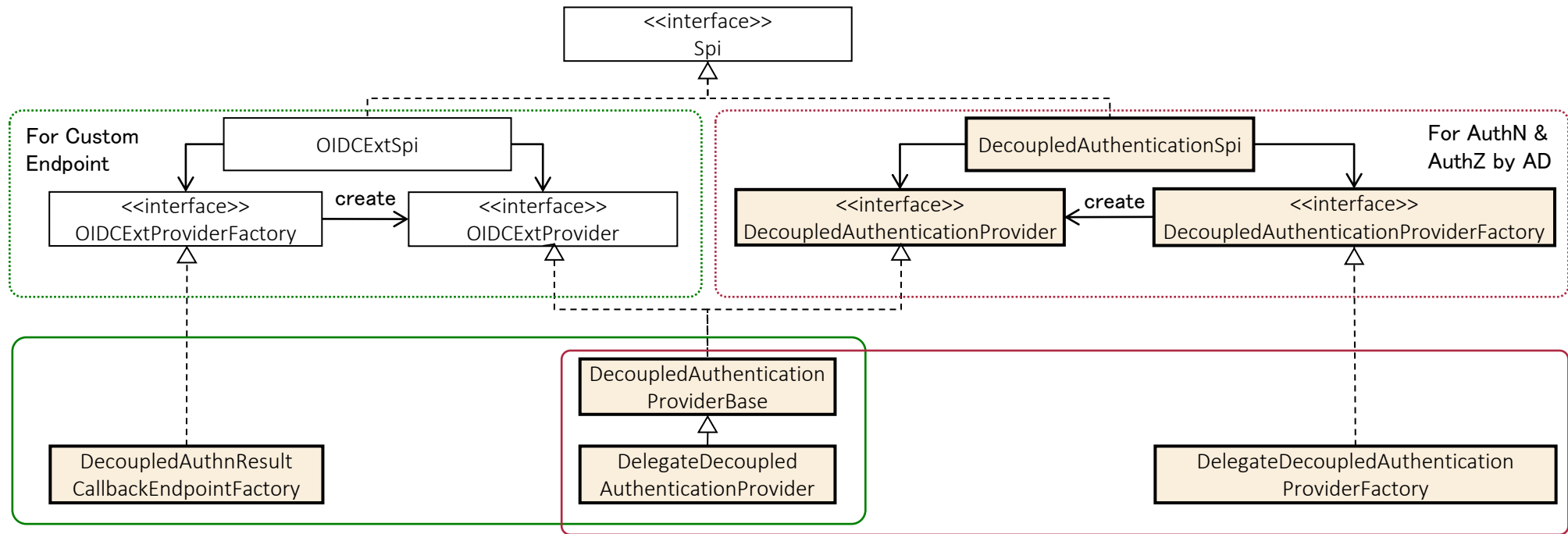


In keycloak, AuthN & AuthZ flow (e.g. authz code grant, resource owner password credentials grant) is processed by Authentication Flow which generates User Session for authenticated user and Authenticated Client Session for authenticated client.

It includes Authentication Executions that process AuthN & AuthZ. Its actual processor 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.

# CIBA Flow : Providers



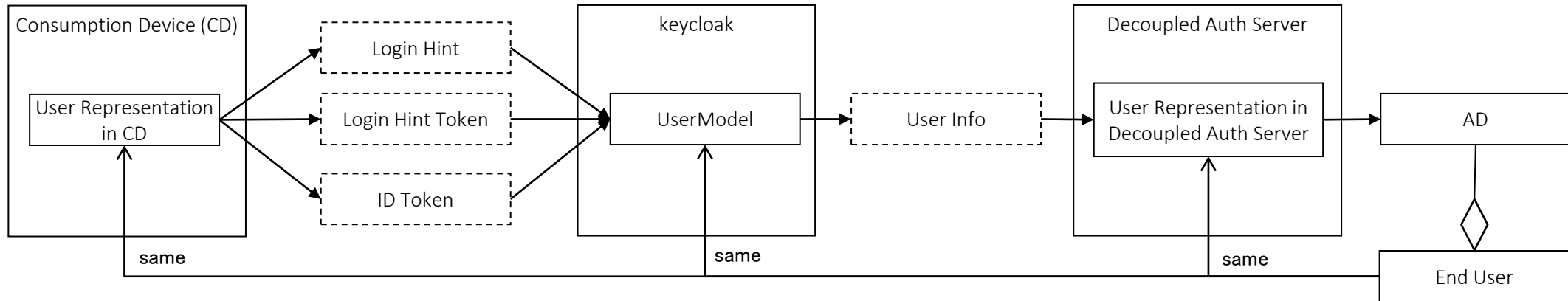
CIBA specification does not specify how to realize AuthN & AuthZ by AD. Therefore, this CIBA prototype implementation defines interfaces of this part and provides its reference implementation.

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

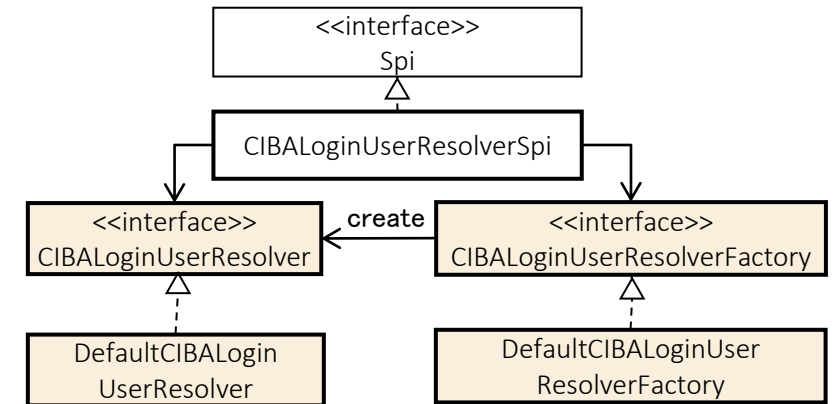


It is possible that each entities included in CIBA flow identifies the same user by the different way.

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.



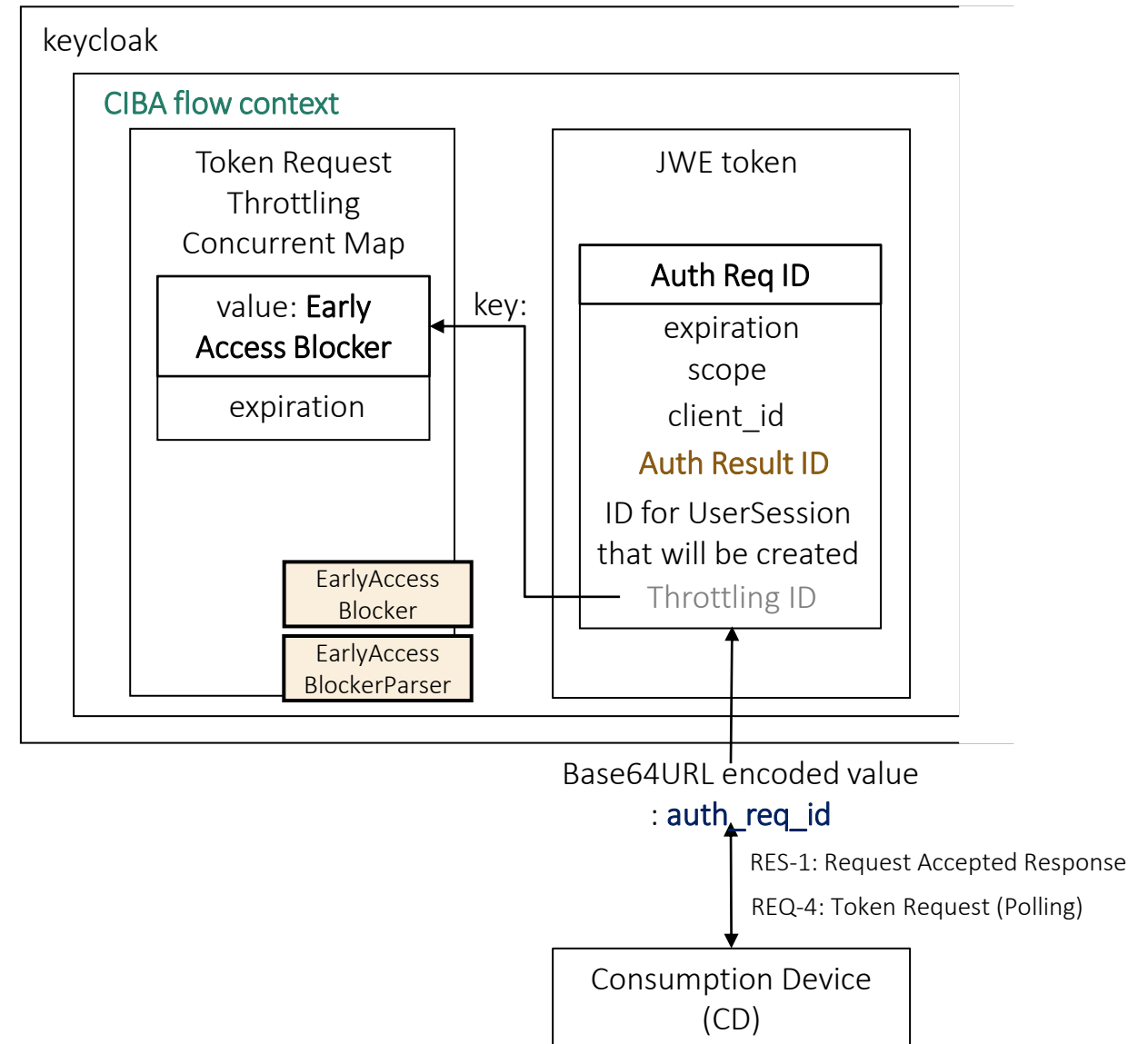
# Token Request Throttling

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 ConcurrentMap 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

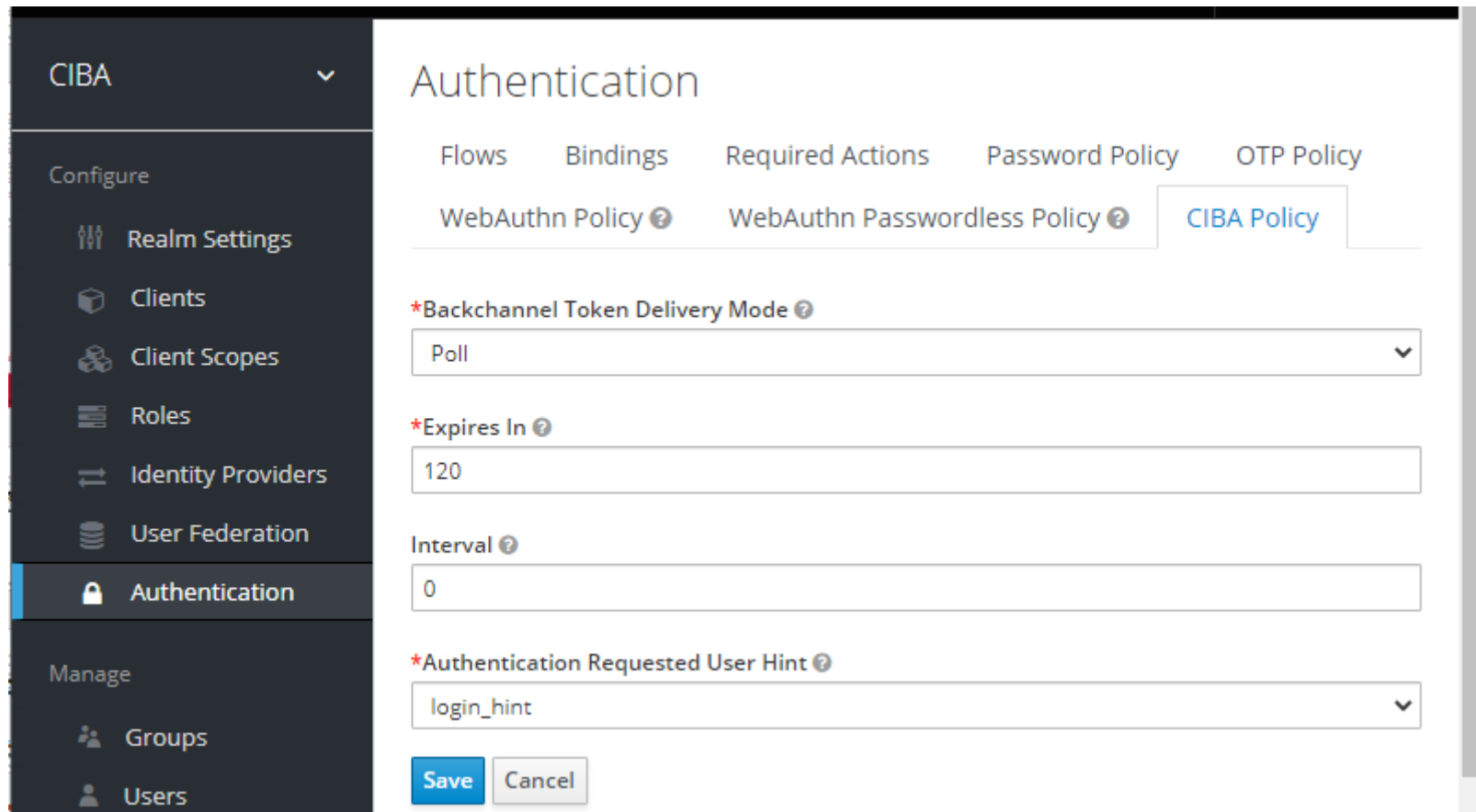
standalone.xml :

```
<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



The screenshot shows the 'CIBA' configuration page in the Azure AD portal. The left sidebar is expanded to 'Authentication'. The main content area is titled 'Authentication' and contains tabs for 'Flows', 'Bindings', 'Required Actions', 'Password Policy', 'OTP Policy', 'WebAuthn Policy', 'WebAuthn Passwordless Policy', and 'CIBA Policy'. The 'CIBA Policy' tab is selected. Below the tabs, there are four configuration fields: '\*Backchannel Token Delivery Mode' (set to 'Poll'), '\*Expires In' (set to '120'), 'Interval' (set to '0'), and '\*Authentication Requested User Hint' (set to 'login\_hint'). At the bottom, there are 'Save' and 'Cancel' buttons.

CIBA

Authentication

Flows Bindings Required Actions Password Policy OTP Policy

WebAuthn Policy WebAuthn Passwordless Policy CIBA Policy

\*Backchannel Token Delivery Mode

Poll

\*Expires In

120

Interval

0

\*Authentication Requested User Hint

login\_hint

Save Cancel

No need to modify this default settings.

# Run with Decoupled Auth Server Reference Implementation

FYI : CIBA Settings - CIBA Flow

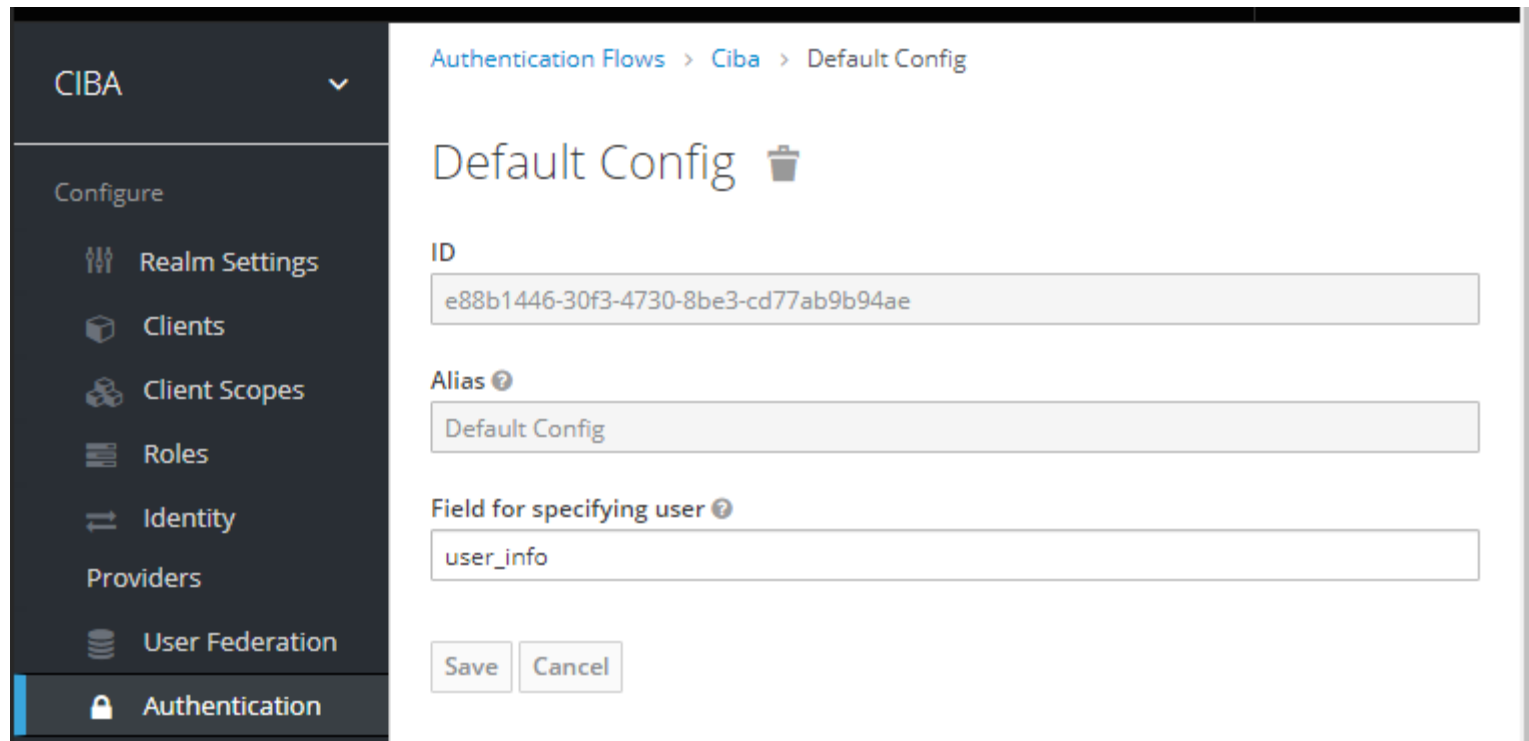
The screenshot shows the 'Authentication' configuration page for the 'Ciba' flow. The left sidebar contains a 'Configure' section with various options, and 'Authentication' is selected. The main panel has tabs for 'Flows', 'Bindings', 'Required Actions', 'Password Policy', 'OTP Policy', 'WebAuthn Policy', 'WebAuthn Passwordless Policy', and 'CIBA Policy'. The 'Flows' tab is active, displaying a table with the following data:

Auth Type	Requirement	
CIBA Decoupled Authenticator (Default Config)	<input checked="" type="radio"/> REQUIRED	Actions ▾

No need to modify this default settings.

# Run with Decoupled Auth Server Reference Implementation

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



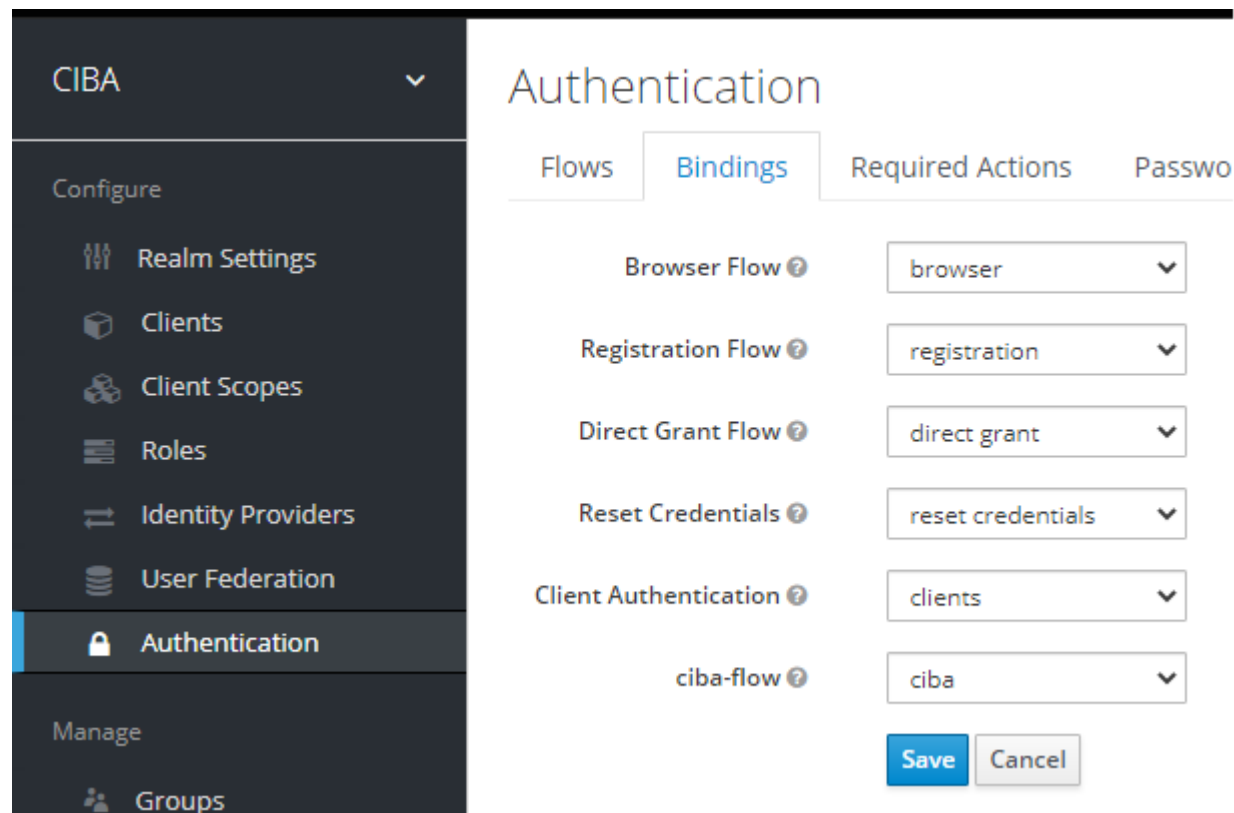
The screenshot displays the configuration page for the CIBA flow. On the left is a dark sidebar with a 'CIBA' header and a 'Configure' section containing links to 'Realm Settings', 'Clients', 'Client Scopes', 'Roles', 'Identity', 'Providers', 'User Federation', and 'Authentication' (which is highlighted). The main content area has a breadcrumb trail 'Authentication Flows > Ciba > Default Config'. Below this, the title 'Default Config' is followed by a trash icon. Three text input fields are present: 'ID' with the value 'e88b1446-30f3-4730-8be3-cd77ab9b94ae', 'Alias' with the value 'Default Config', and 'Field for specifying user' with the value 'user\_info'. At the bottom are 'Save' and 'Cancel' buttons.

No need to modify this default settings.



# Run with Decoupled Auth Server Reference Implementation

FYI : CIBA Settings - CIBA Flow - Flow Binding



The screenshot displays the 'CIBA' configuration page in an authentication management console. On the left is a dark sidebar with a 'CIBA' header and a dropdown arrow. Below it, the 'Configure' section lists: Realm Settings, Clients, Client Scopes, Roles, Identity Providers, User Federation, and 'Authentication' (which is highlighted with a blue bar). The 'Manage' section at the bottom lists 'Groups'. The main content area is titled 'Authentication' and has four tabs: 'Flows', 'Bindings' (selected), 'Required Actions', and 'Passwo'. Under the 'Bindings' tab, there are six rows, each with a flow name, a help icon, and a dropdown menu. The flows and their selected values are: Browser Flow (browser), Registration Flow (registration), Direct Grant Flow (direct grant), Reset Credentials (reset credentials), Client Authentication (clients), and ciba-flow (ciba). At the bottom right of the configuration area are 'Save' and 'Cancel' buttons.

Flow	Binding
Browser Flow ?	browser
Registration Flow ?	registration
Direct Grant Flow ?	direct grant
Reset Credentials ?	reset credentials
Client Authentication ?	clients
ciba-flow ?	ciba

No need to modify this default settings.

End