MobileID InApp

Mobile integration guide

2020-02-19

Version 0.4

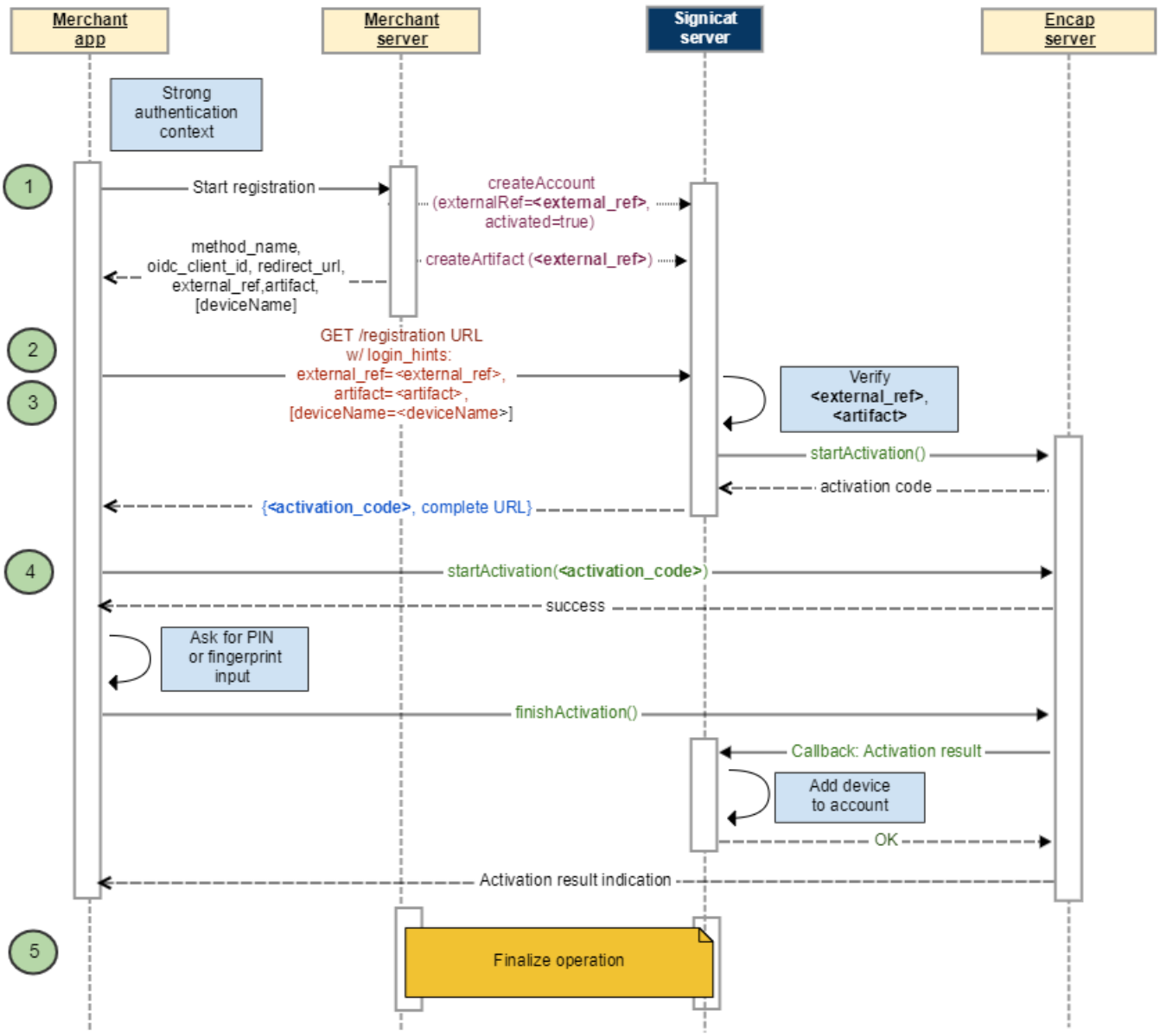
Contents

This guide illustrates how to integrate with MobileID InApp by both registering the user and providing authentication solely within the merchant's app.

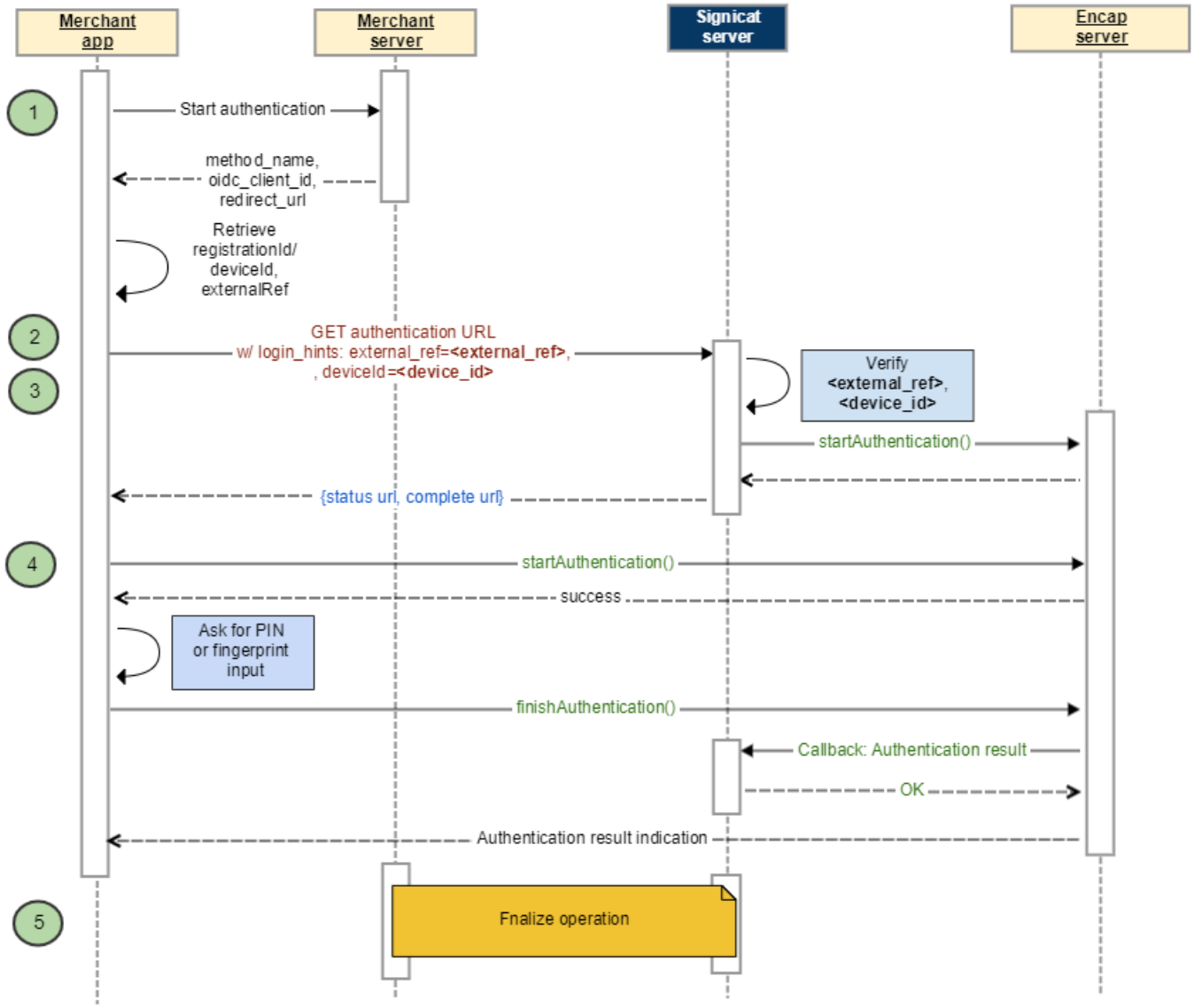
1 Integration process

1.1 Sequence diagrams for registration and authentication

1.1.1 *Registration*



1.1.2 *Authentication*



1.2 Detailed description of registration and authentication

The following table provides step-by-step instructions on the integration process. The numbering in the table corresponds to the numbered steps in the sequence diagrams above.

	Registration	**Authentication**

1. Initiate operation on merchant server	The goal of this call is to obtain other necessary information to build a proper registration URL.
	Note: The nature of the call is out of scope of this document (but this is most commonly HTTP GET).
	
	Account and Artifact Creation
	
	In order to carry out registration, an account with Signicat needs to be created. The merchant uses two SOAP WS calls:
	
	• One call for account creation
	  o It is important that the flag `activated` is true
	  o It is possible to add additional attributes to the account at the time of creation.
	• One call for creation of `artifact` that has to be passed to Signicat's registration service when operation is initiated
	
	The following information is passed back in response:
	
	• The name of the `oidc_client_id`
	• The name of the Signicat registration method
	• `redirect_url` (on merchant server) where final results should end
	• `external_ref` (account name with Signicat)
	• `artifact`, just created
		The goal of this call is to obtain necessary information to build a proper authentication URL.
		Note: The nature of the call is out of scope of this document (but this is most commonly HTTP GET).
		
		The following information is passed back in response:
		
		• The name of the `oidc_client_id`
		• The name of the Signicat authentication method
		• `redirect_url` (on merchant server) where final results should end

2. Generate URL	Construct a registration URL as shown in the MobileID InApp integration guide - URL Construction Guide, based on information received from the merchant server in the previous step.	Construct an authentication URL as shown in the MobileID InApp integration guide - URL Construction Guide, based on information received from the merchant server in the previous step as well as the information already available through the merchant app: • `externalRef` • `deviceId` (to be extracted from `EncapController:getRegistrationId())`

<table>
<tr><th></th><th>Registration</th><th>Authentication</th></tr>
<tr><td>3. Initiate operation on Signicat's server</td><td colspan="2">The merchant app executes an HTTP GET request with the URL constructed previously. See the normal response in the URL construction guides linked to above.

Note: To be able to perform the subsequent requests, you must keep the cookies you receive and make these available for subsequent requests.

Response error example
<pre>{
    "completeUrl": "https://id.signicat.com/...",
    "status": "ERROR",
    "error": {
        "code": "urn:signicat:error:idp:ACCESS_DENIED",
        "message": "Access denied. Wrong credentials."
    },
    ...
}</pre>• If an error occurs during initialization, you will receive a status indicating this, and an error object will be present. Upon error, if you choose to make a GET request towards the <code>completeUrl</code>, you will get

<pre>error=access_denied&
error_description=The Resource Owner did not complete the login.
urn:signicat:error:idp:ACCESS_DENIED; Access denied. Wrong credentials.</pre></td></tr>
<tr><td>4. Execute operation toward Encap</td><td>If the status was "OK", you can start the Encap activation.
This involves the regular <code>startActivation()</code> / <code>finishActivation()</code> calls towards the Encap Client API.</td><td>If the status was "OK", you can start the Encap authentication.
This involves the regular <code>startAuthentication()</code> / <code>finishAuthentication()</code> calls towards the Encap Client API.</td></tr>
<tr><td>5. Get result of the process — Finalize operation</td><td colspan="2">Using the <code>completeUrl</code> received in step 1, execute a GET request for the <code>authorization_code</code>.

Carry out the regular OIDC authorization_code sequence of steps to obtain the device information. See the MobileID InApp integration guide - Finalize operation guide for details.</td></tr>
</table>

2 URL construction guide

2.1 Requests and responses

2.1.1 Registration

Registration OIDC request	Registration response

Without PKCE
`GET <SIGNICAT_AUTHORIZATION_ENDPOINT>?`
`response_type=code&`
`scope=openid+profile+mobileid&`
`client_id=<CUSTOMER_CLIENT_ID>&`
`redirect_uri=<CUSTOMER_REDIRECT_URI>&`
`state=<CUSTOMER_REG_METHOD_NAME:STATE_IDENTIFIER>&`
`acr_values= urn:signicat:oidc:method:<CUSTOMER_REG_METHOD_NAME>&`
`login_hint=deviceName-<DEVICE_NAME>&`
`login_hint=artifact-<ARTIFACT>&`
`login_hint=externalRef-<ACCOUNT_NAME>`
 	
With PKCE
`GET <SIGNICAT_AUTHORIZATION_ENDPOINT>?`
`response_type=code&`
`scope=openid+profile+mobileid&`
`client_id=<CUSTOMER_CLIENT_ID>&`
`redirect_uri=<CUSTOMER_REDIRECT_URI>&`
`code_challenge=<CODE_CHALLENGE>&`
`code_challenge_method=S256&`
`state=<CUSTOMER_REG_METHOD_NAME:STATE_IDENTIFIER>&`
`acr_values= urn:signicat:oidc:method:<CUSTOMER_REG_METHOD_NAME>&`
`login_hint=deviceName-<DEVICE_NAME>&`
`login_hint=artifact-<ARTIFACT>&`
`login_hint=externalRef-<ACCOUNT_NAME>`
	`{`
	`"status":"<STATUS>",`
	`"activationCode":"<ACTIVATION_CODE>",`
	`"statusUrl":"<STATUS_URL>",`
	`"completeUrl":"<COMPLETE_URL>"`
	`}`

2.1.2 *Authentication*

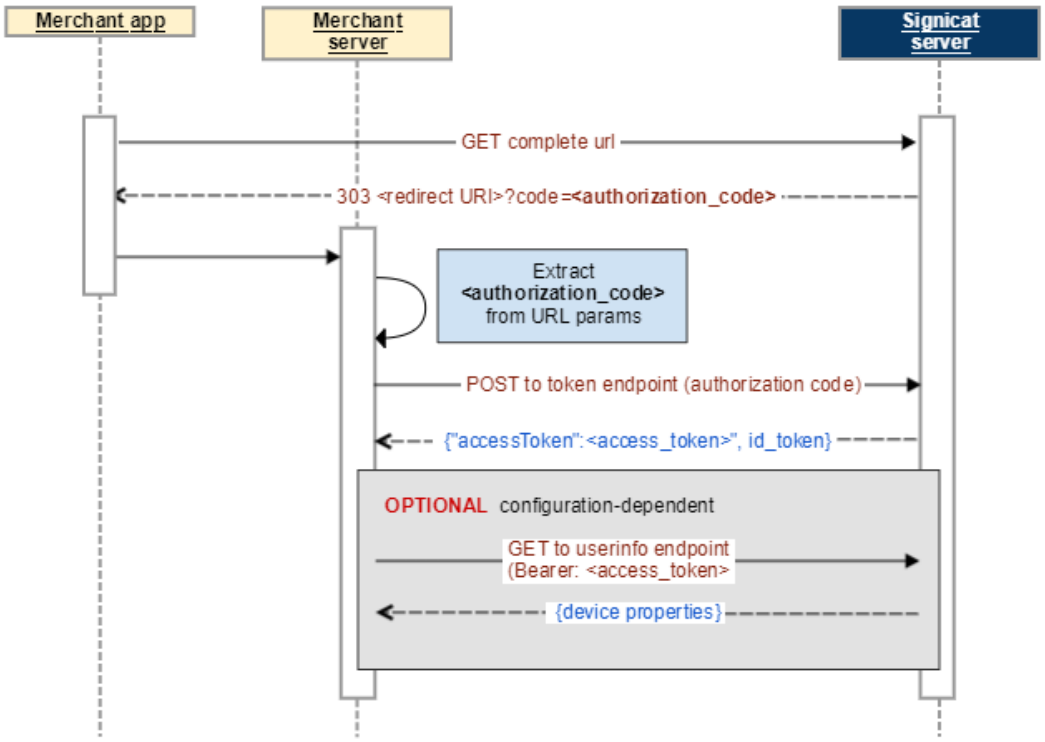
Authentication OIDC request	Authentication response

Without PKCE
`GET <SIGNICAT_AUTHORIZATION_ENDPOINT>?`
`response_type=code&`
`scope=openid+profile+mobileid&`
`client_id=<CUSTOMER_CLIENT_ID>&`
`redirect_uri=<CUSTOMER_REDIRECT_URI>&`
`state=<CUSTOMER_AUTH_METHOD_NAME:STATE_IDENTIFIER>&`
`acr_values= urn:signicat:oidc:method:<CUSTOMER_AUTH_METHOD_NAME>&`
`login_hint=deviceId-<DEVICE_ID>&`
`login_hint=externalRef-<ACCOUNT_NAME>`
With PKCE
`GET <SIGNICAT_AUTHORIZATION_ENDPOINT>?`
`response_type=code&`
`scope=openid+profile+mobileid&`
`client_id=<CUSTOMER_CLIENT_ID>&`
`redirect_uri=<CUSTOMER_REDIRECT_URI>&`
`code_challenge=<CODE_CHALLENGE>&`
`code_challenge_method=S256&`
`state=<CUSTOMER_AUTH_METHOD_NAME:STATE_IDENTIFIER>&`
`acr_values= urn:signicat:oidc:method:<CUSTOMER_AUTH_METHOD_NAME>&`
`login_hint=deviceId-<DEVICE_ID>&`
`login_hint=externalRef-<ACCOUNT_NAME>`
	`{`
	`"status":"<STATUS>",`
	`"statusUrl":"<STATUS_URL>",`
	`"completeUrl":"<COMPLETE_URL>"`
	`}`

2.2 Parameters

Parameter	Description
`STATE_IDENTIFIER`	Random text used together with `CUSTOMER_REG_METHOD_NAME` to uniquely identify the ongoing registration session in the merchant's backend. The session state can be compared when callback/ redirect data is received from Signicat.
`ACTIVATION_CODE`	Code to be used with Encap.
`STATUS_URL`	URL (towards Signicat's server) that is used to get the status of the ongoing operation.
`COMPLETE_URL`	URL (towards Signicat's server) that is used to signal the completion of the transaction. This will need to be used when the merchant's app gets notification from the MobileID App that the registration is done.
`DEVICE_ID`	Device ID
`CODE_CHALLENGE`	PKCE Code Challenge. Base64UrlEncoded SHA256 of the the value for `CODE_VERIFIER` (to be used later when the authentication code is exchanged for `access_token`)
`CODE_CHALLENGE_METHOD`	PKCE Code Challenge Method. Recommended: S256

3 Finalize operation



Note: This operation is carried out in the same way regardless of whether the operation in question is **registration** or **authentication**.

3.1 Complete operation

Request	Response	Comment
`GET <COMPLETE_URL>`	`AUTHORIZATION_CODE`	Signicat's server sends an authorization code to the `CUSTOMER_REDIRECT_URL`.

`POST <SIGNICAT_TOKEN_ENDPOINT>`
`HTTP/1.1`
`Content-Type: application/json`
`Authorization: Basic <CUSTOMER_BASIC_AUTH_HEADER>`
`#body`
`client_id=<CUSTOMER_CLIENT_ID>&`
`redirect_uri=<CUSTOMER_REDIRECT_URI>&`
`grant_type=authorization_code&`
`code=<AUTHORIZATION_CODE>`
	`{`
	`"access_token": "<ACCESS_TOKEN>",`
	`"token_type": "Bearer",`
	`...`
	`}`
		The authorization code is exchanged for an access token.

[OPTIONAL]
`GET <SIGNICAT_USERINFO_ENDPOINT>`
`HTTP/1.1`
`Content-Type: application/json`
`Authorization: Bearer <ACCESS_TOKEN>`
	For registration:
	`{`
	`"sub": "<SUBJECT>",`
	`"name": "<EXTERNAL_REF>"`
	`...`
	`}`
	
	For Authentication:
	`{`
	`"sub": "<SUBJECT>",`
	`"externalRef": "<EXTERNAL_REF>",`
	`"deviceName": "<DEVICE_NAME>",`
	`...`
	`}`
		Additional information (such as data on the authenticated user) can be retrieved from Signicat's OIDC server using the /userinfo endpoint.