Financial-grade API (FAPI) and CIBA

DeveloperWeek NYC 2019 @ Brooklyn Expo Center on June 20, 2019

OAuth 2.0
OpenID Connect
Authorization Focused • Reliable and Scalable • Developer Friendly
Faster Time to Market • Choice of Hosting Options • Broad Usage
Integrates with any Authentication methods
API Security
OpenID
AUTHLETE

Co-founder, Authlete, Inc.
Takahiko Kawasaki <taka@authlete.com>

AUTHLETE

Company Profile

Name	Authlete, Inc.
Establishment	September 18, 2015
Capital	444,710,000 JPY (including the capital reserve)
Website	https://www.authlete.com/

Offices

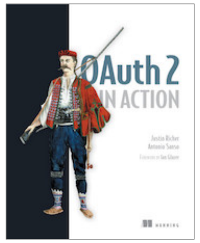
Tokyo	FINOLAB, Otemachi Bldg 4F, Otemachi 1-6-1, Chiyoda-ku, Tokyo, 100-0004, Japan
London	Level39, One Canada Square, Canary Wharf, London E14 5AB, UK

Product

Authlete, SaaS providing Web APIs whereby developers implement servers that support OAuth 2.0 and OpenID Connect.

Team

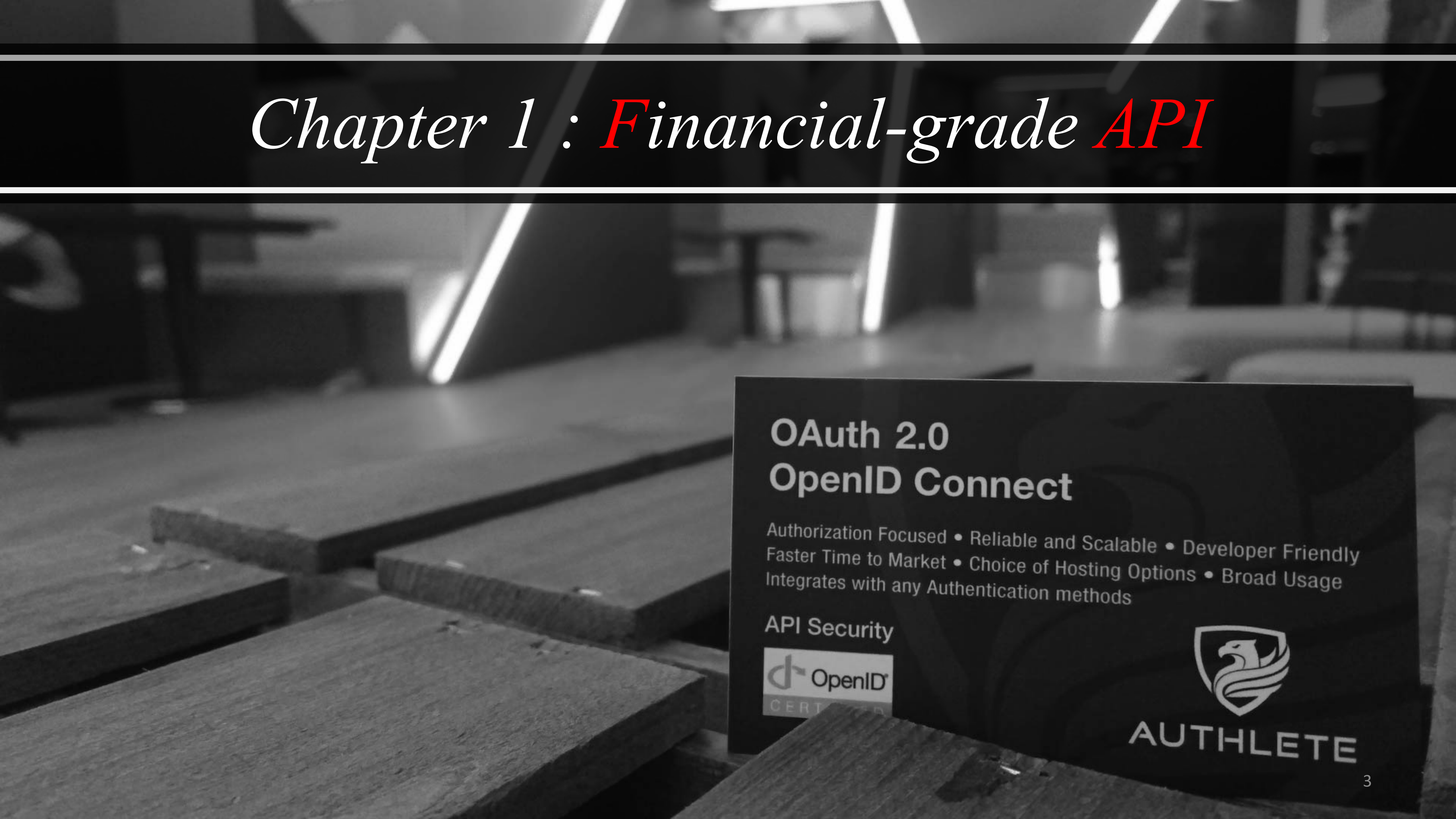
Takahiko Kawasaki – Co-founder, software engineer
Ali Adnan – Co-founder, multilingual serial entrepreneur
Joseph Heenan – Lead of the official OpenID test suite
Justin Richer – Author of "OAuth 2 in Action"
and other wonderful members



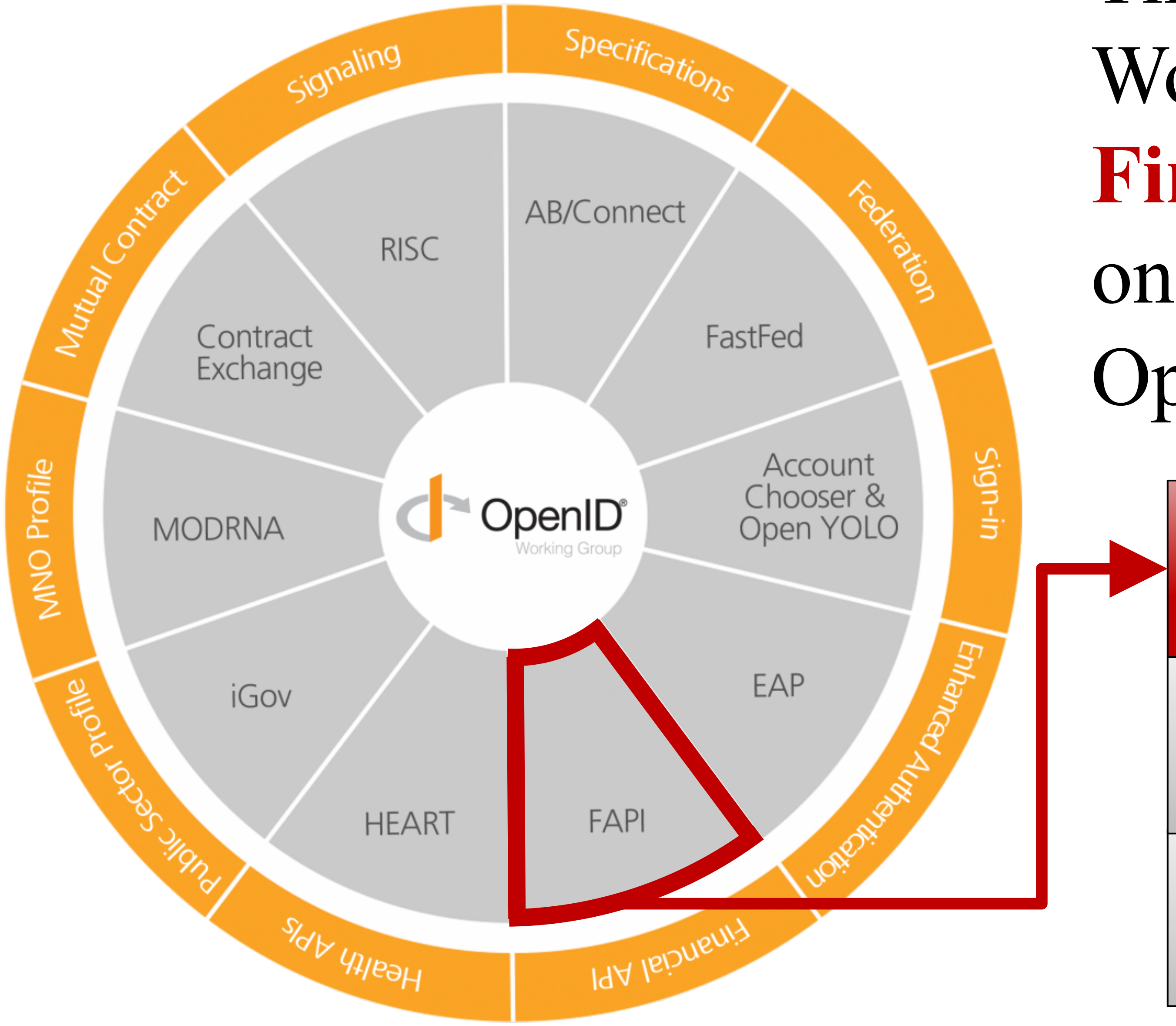
History

Jan. 2014	Starts to implement Authlete
Sep. 2015	Establishes Authlete, Inc.
Sep. 2016	Establishes Authlete UK, Ltd.
Nov. 2016	Joins FINOLAB
Feb. 2017	Joins OpenID Foundation
Mar. 2017	Wins FIBC 2017 Grand Prize
May 2017	Joins Level39
May 2017	Fund Raising (seed round)
Jul. 2017	Gets OpenID Certification
Aug. 2017	Cyber39 Founding Member
Sep. 2017	Tech in Asia Tokyo 2017 Finalist
Feb. 2018	Fund Raising (pre-series A)
Apr. 2018	Wins IBM Prize at Draper Nexus B2B Summit 2018
Jul. 2018	Joins Fintech Association of Japan
Jul. 2018	Organizes Japan/UK Open Banking and APIs Summit 2018
Jul. 2018	Supports Financial-grade API (Authlete 2.0)
Aug. 2018	Passes Open Banking Security Profile Test
Jan. 2019	Supervises "OAuth 徹底入門" (book)
Feb. 2019	Supports CIBA
Apr. 2019	Gets Certified Financial-grade API (FAPI) OpenID Provider

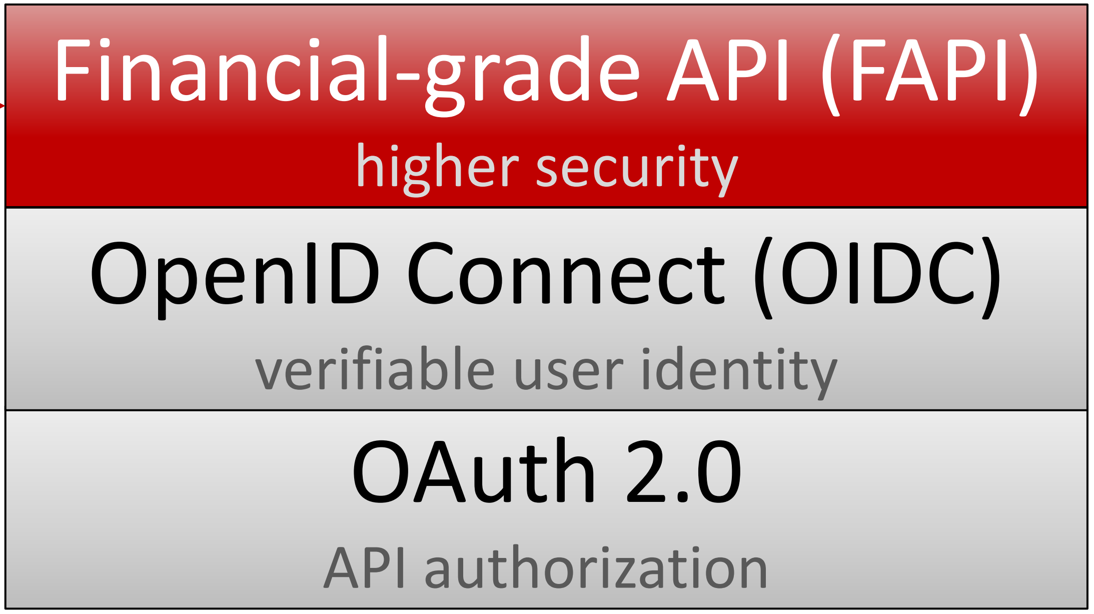
Chapter 1 : Financial-grade API


OAuth 2.0
OpenID Connect
Authorization Focused • Reliable and Scalable • Developer Friendly
Faster Time to Market • Choice of Hosting Options • Broad Usage
Integrates with any Authentication methods
API Security
OpenID
AUTHLETE

OpenID Foundation



The Financial-grade API (FAPI) Working Group has developed **Financial-grade API (FAPI)** on top of OAuth 2.0 and OpenID Connect.



Financial-grade API (FAPI) higher security
OpenID Connect (OIDC) verifiable user identity
OAuth 2.0 API authorization

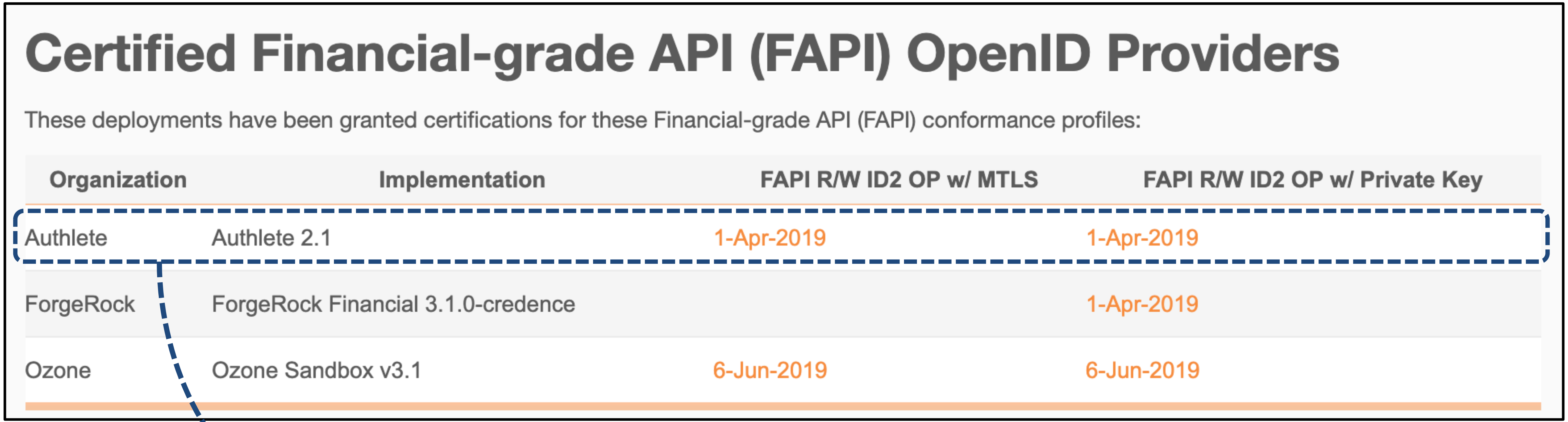
History of FAPI

2017	2	Part 1 of Financial API Implementer's Draft 1
2017	7	Part 2 of Financial API Implementer's Draft 1
2018	10	Financial-grade API Implementer's Draft 2

The specification was renamed from Financial API to Financial-grade API because the specification can apply to not only the financial industry but also other industries that need high security.

FAPI Certification

OpenID Foundation started FAPI Certification Program on April 1, 2019.



Certified Financial-grade API (FAPI) OpenID Providers

These deployments have been granted certifications for these Financial-grade API (FAPI) conformance profiles:

Organization	Implementation	FAPI R/W ID2 OP w/ MTLS	FAPI R/W ID2 OP w/ Private Key
Authlete	Authlete 2.1	1-Apr-2019	1-Apr-2019
ForgeRock	ForgeRock Financial 3.1.0-credence		1-Apr-2019
Ozone	Ozone Sandbox v3.1	6-Jun-2019	6-Jun-2019

(Certified FAPI OPs, as of June 12, 2019)

FAPI Parts

From the foreword of FAPI specification:

Financial-grade API consists of the following parts:

- *Part 1: Read-Only API Security Profile*
- *Part 2: Read and Write API Security Profile*
- *Part 3: Client Initiated Backchannel Authentication Profile*

CIBA specification adds new authorization flows.

2019	2	CIBA Core 1.0

NEW

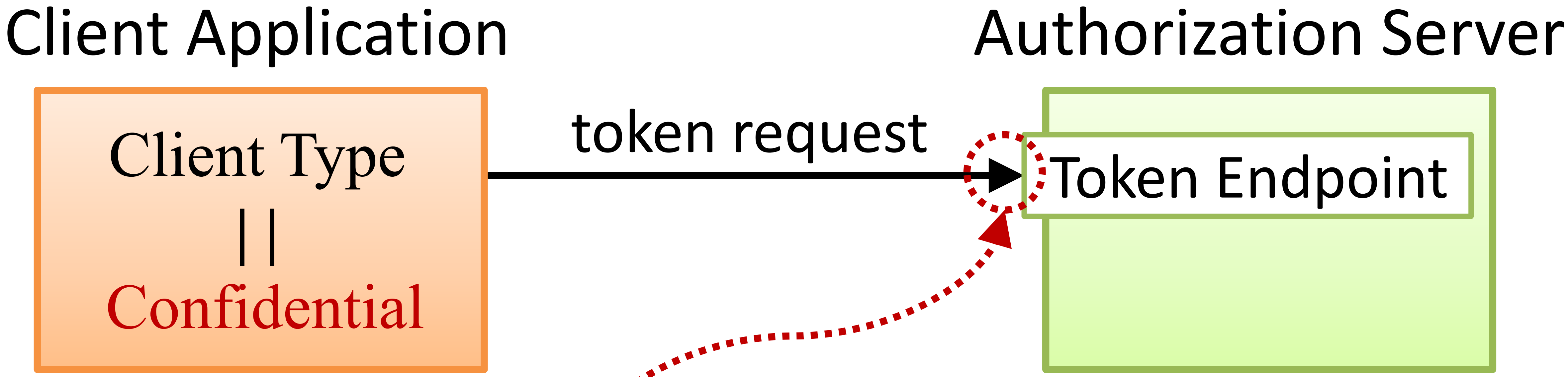
AUTHLETE

Enhanced Security

Topics covered in this talk

- ✓ Entropy Requirement for Client Secret
- ✓ JWT-Based Client Authentication
- ✓ Certificate-Based Client Authentication
- ✓ Key Size Requirement for Client Authentication
- ✓ Proof Key for Code Exchange
- ✓ Redirect URI Pre-registration
- ✓ Redirect URI Mandatory Request Parameter
- ✓ Redirect URI Exact Match
- ✓ Level of Assurance for End-User Authentication
- ✓ Explicit Consent for Requested Scopes
- ✓ Prohibition of Authorization Code Reuse
- ✓ Scope Mandatory Response Parameter
- ✓ Entropy Requirement for Access Token
- ✓ Access Token Revocation
- ✓ Claimed HTTP Scheme URI Redirection
- ✓ Prohibition of Access Token in Query Part
- ✓ Detached Signature
- ✓ State Hash
- ✓ Certificate-Bound Access Token
- ✓ Token Binding
- ✓ Request Object Mandatory Request Parameter
- ✓ Request Object including All Request Parameters
- ✓ Request Object EXP Claim
- ✓ Request Object Mandatory Signing
- ✓ Essential ACR Claim
- ✓ JWT Secured Authorization Response Mode
- ✓ TLS Cipher Suite Restriction
- ✓ JWS Signature Algorithm Restriction

Client Authentication



Client Authentication is required when a confidential client accesses the token endpoint.

The traditional ways described in RFC 6749 use Client ID and Client Secret for client authentication.

1. Basic Authentication (`client_secret_basic`)

```
"{Client ID}:{Client Secret}"
```

Encode by BASE64

```
POST {Token Endpoint} HTTP/1.1
Host: {Authorization Server}
Authorization: Basic {BASE64-encoded Credentials}
Content-Type: application/x-www-form-urlencoded

 (abbrev)
```

2. Form Parameters (`client_secret_post`)

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

client_id={Client ID}&
client_secret={Client Secret}&
 (abbrev)
```

These traditional ways (`client_secret_basic` and `client_secret_post`) are not allowed in FAPI.

Client Authentication Method		Part 1	Part 2
`client_secret_basic`	traditional	×	×
`client_secret_post`		×	×
`client_secret_jwt`	JWT-based	○	×
`private_key_jwt`		○	○
`tls_client_auth`	certificate-based	○	○
`self_signed_tls_client_auth`		○	○

JWT-based Client Authentication (RFC 7523)

- ✓ Generate JWT and pass it to the token endpoint instead of passing a pair of client ID & client secret directly.
- ✓ The JWT is passed as the value of `client_assertion`.
- ✓ The JWT is signed using either
  (a) the client's client secret (`client_secret_jwt`), or
  (b) the client's private key (`private_key_jwt`).

AUTHLETE

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

client_assertion_type=
   urn:ietf:params:oauth:client-assertion-type:jwt-bearer&
client_assertion={JWT}&
(abbrev)
```

payload

```
{
  "iss": "{Client ID}",
  "sub": "{Client ID}",
  "aud": "{Token Endpoint}",
  "jti": "{JWT ID}",
  "exp": {Expiration Time},
  "iat": {Issue Time}
}
```

The `iss` claim and the `sub` claim in the JWT hold the client ID.

Certificate-based Client Authentication

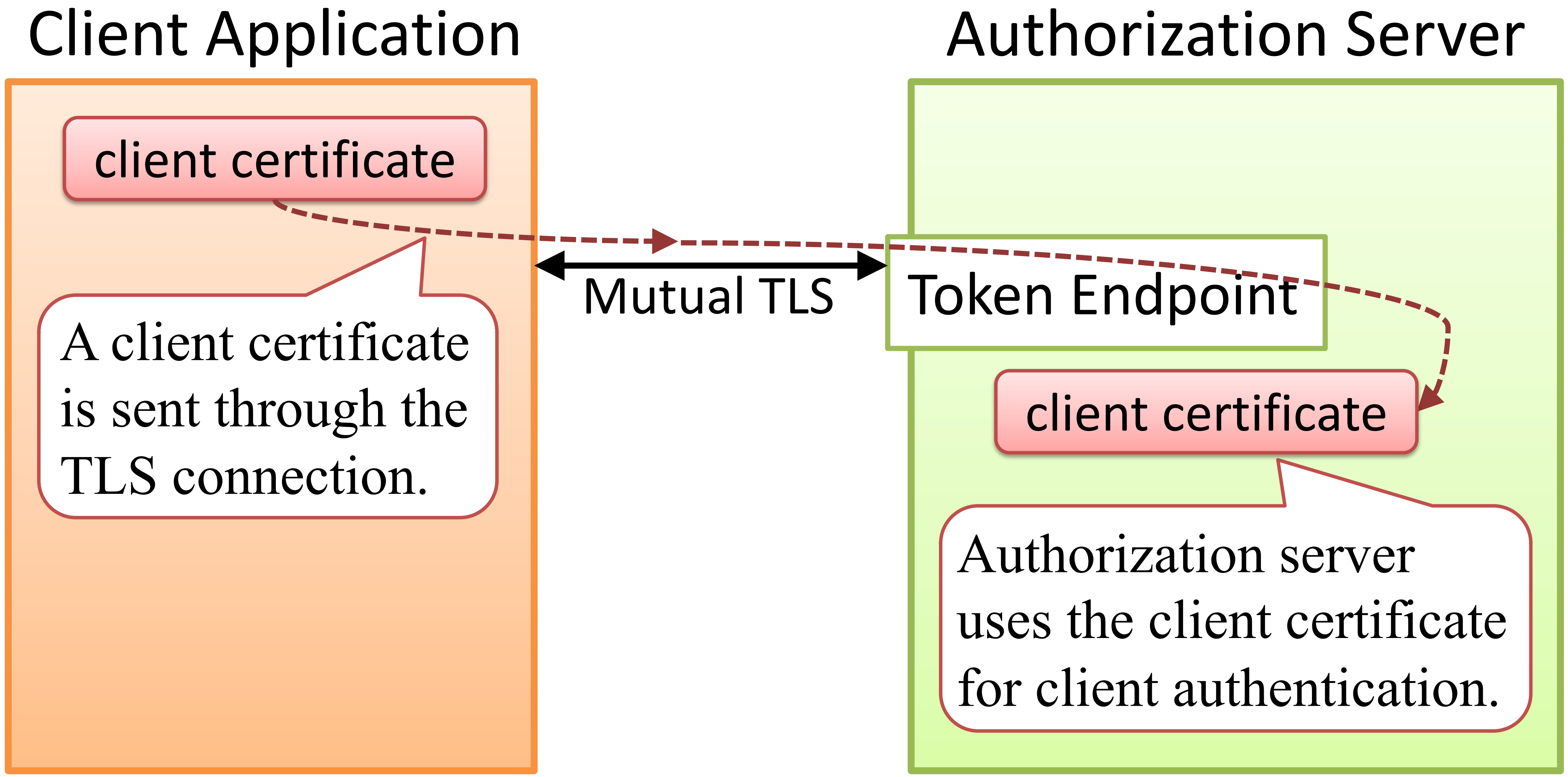
✓ Establish mutual TLS connection to the token endpoint.

✓ The client certificate presented in the connection is used for client authentication.

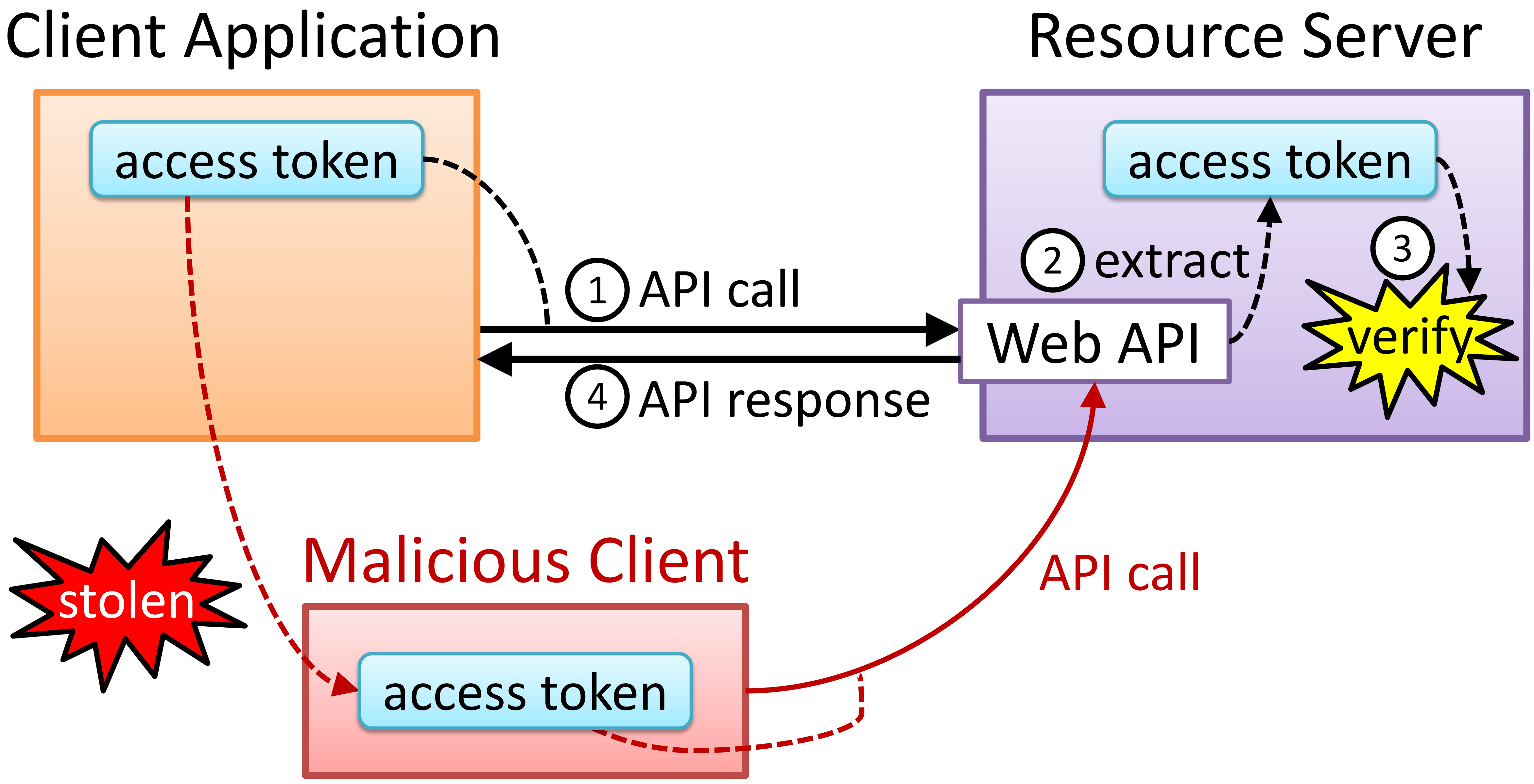
✓ The client certificate is either

(a) PKI certificate (`tls_client_auth`), or

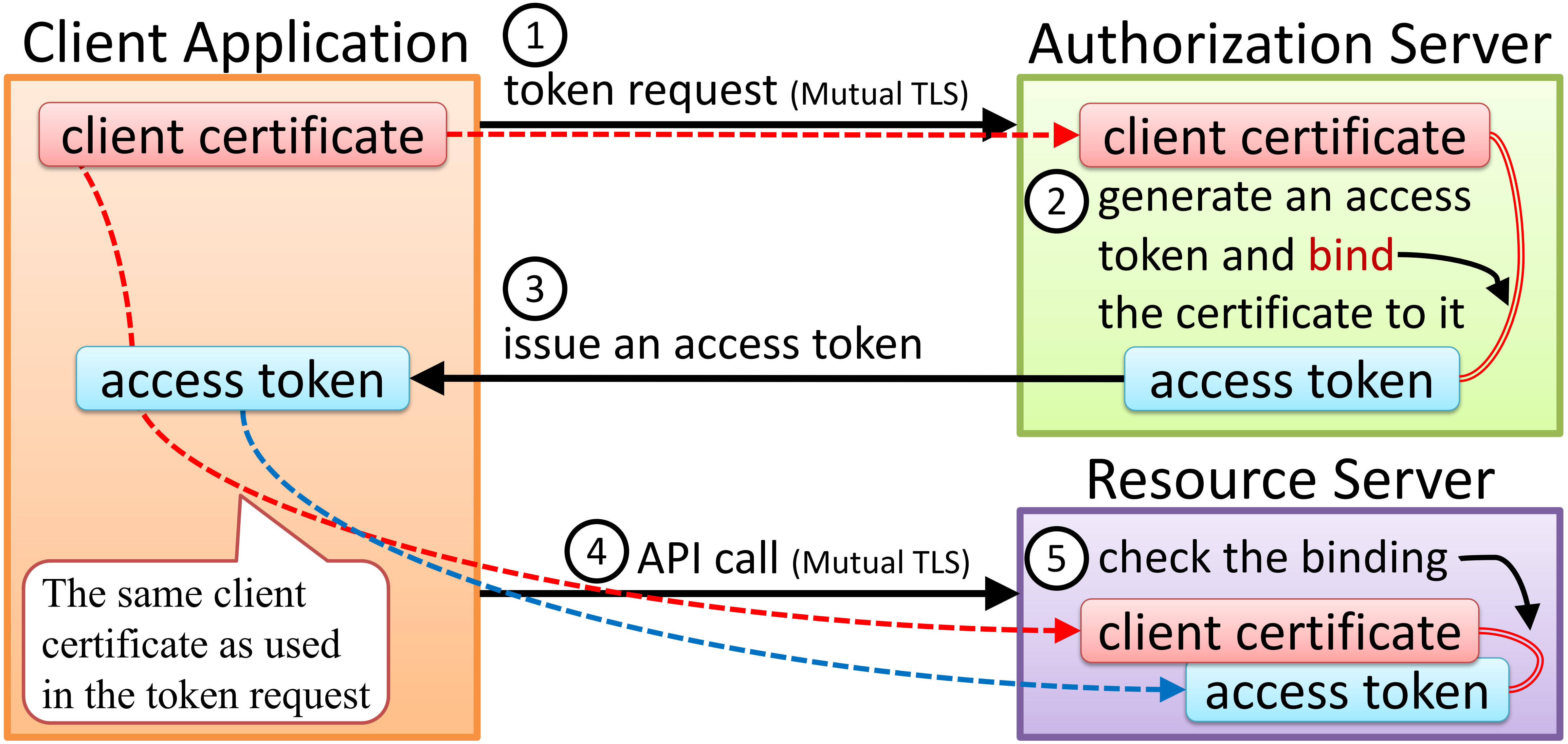
(b) self-signed certificate (`self_signed_tls_client_auth`).


Client Application
Authorization Server
client certificate
A client certificate is sent through the TLS connection.
Mutual TLS
Token Endpoint
client certificate
Authorization server uses the client certificate for client authentication.

Certificate-Bound Access Token


Client Application
Resource Server
access token
access token
① API call
② extract
③
verify
Web API
④ API response
stolen
Malicious Client
access token
API call

Certificate-Bound Access Token



JWT Secured Authorization Response Mode (JARM)

JARM is a specification to pack response parameters from the authorization endpoint into a JWT.

In normal cases

```
HTTP/1.1 302 Found
Location: https://client.example.com/callback?
  code={Authorization Code}&state={State}
```

In JARM

```
HTTP/1.1 302 Found
Location: https://client.example.com/callback?
  response={JWT}
```

AUTHLETE

Example of an authorization response in JARM

```
HTTP/1.1 302 Found
Location: https://client.example.com/cb?response=eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ
9.eyJpc3MiOiJodHRwczovL2FjY291bnRzLmV4YW1wbGUuY29tIiwiYXVkIjoiczZCaGRSa3F0MyIsImV4cC
I6MTMxMTI4MTk3MCwiY29kZSI6IlB5eUZhdXgybzdRMFlmWEJVMzJqaHcuNUZYU1FwdnI4YWt2OUNlUkRTZD
BRQSIsInN0YXRlIjoiUzhOSjd1cWs1Zlk0RWpOdlBfR19GdHlKdTZwVXN2SDlqc1luaTlkTUFKdyJ9.HkdJ_
TYgwBBj10C-aWuNUiA062Amq2b0_oyuc5P0aMTQphAqC2o9WbGSkpfuHVBowlb-zJ15tBvXDIABL_t83q6aj
vjtq_pqsByiRK2dLVdUwKhW3P_9wjvI0K20gdoTNbNlP9Z41mhart4BqraIoI8e-L_EfAHfhCG_DDDv7Yg
```

Decoded payload

```
{
  "iss": "https://accounts.example.com",
  "aud": "s6BhdRkqt3",
  "exp": 1311281970,
  "code": "PyyFaux2o7Q0YfXBU32jhw.5FXSQpvr8akv9CeRDSd0QA",
  "state": "S8NJ7uqk5fY4EjNvP_G_FtyJu6pUsvH9jsYni9dMAJw"
}
```

To use JARM, include the `response_mode` parameter with `*.jwt`.

```
response_mode=query.jwt
             |fragment.jwt
             |form_post.jwt
             |jwt
```

```
GET {Authorization Endpoint}
  ?response_type={Response Type}
  &client_id={Client ID}
  &response_mode=jwt
  HTTP/1.1
Host: {Authorization Server}
```

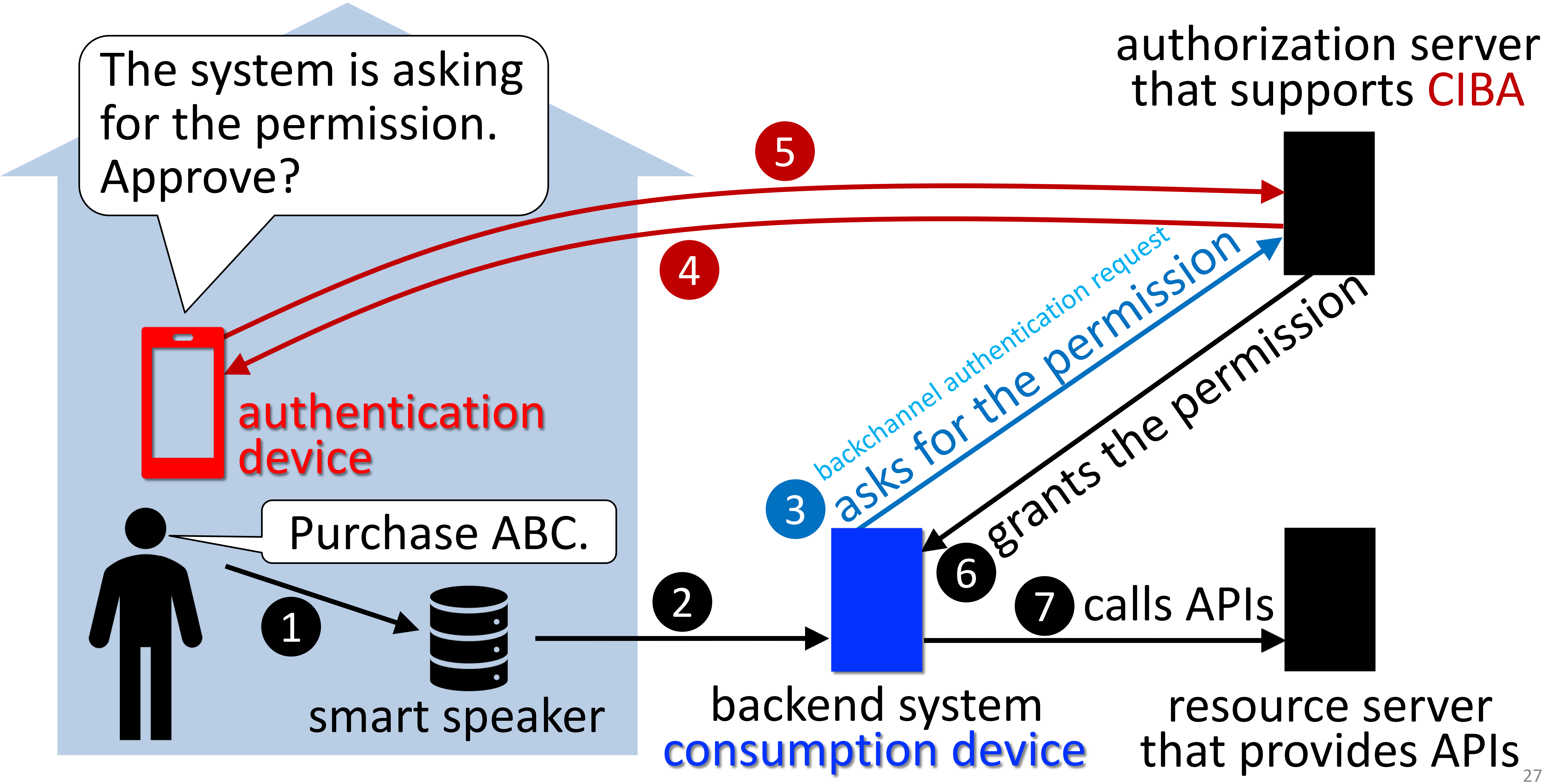
Chapter 2 : Client Initiated Backchannel Authentication

OAuth 2.0
OpenID Connect
Authorization Focused • Reliable and Scalable • Developer Friendly
Faster Time to Market • Choice of Hosting Options • Broad Usage
Integrates with any Authentication methods
API Security
OpenID
AUTHLETE

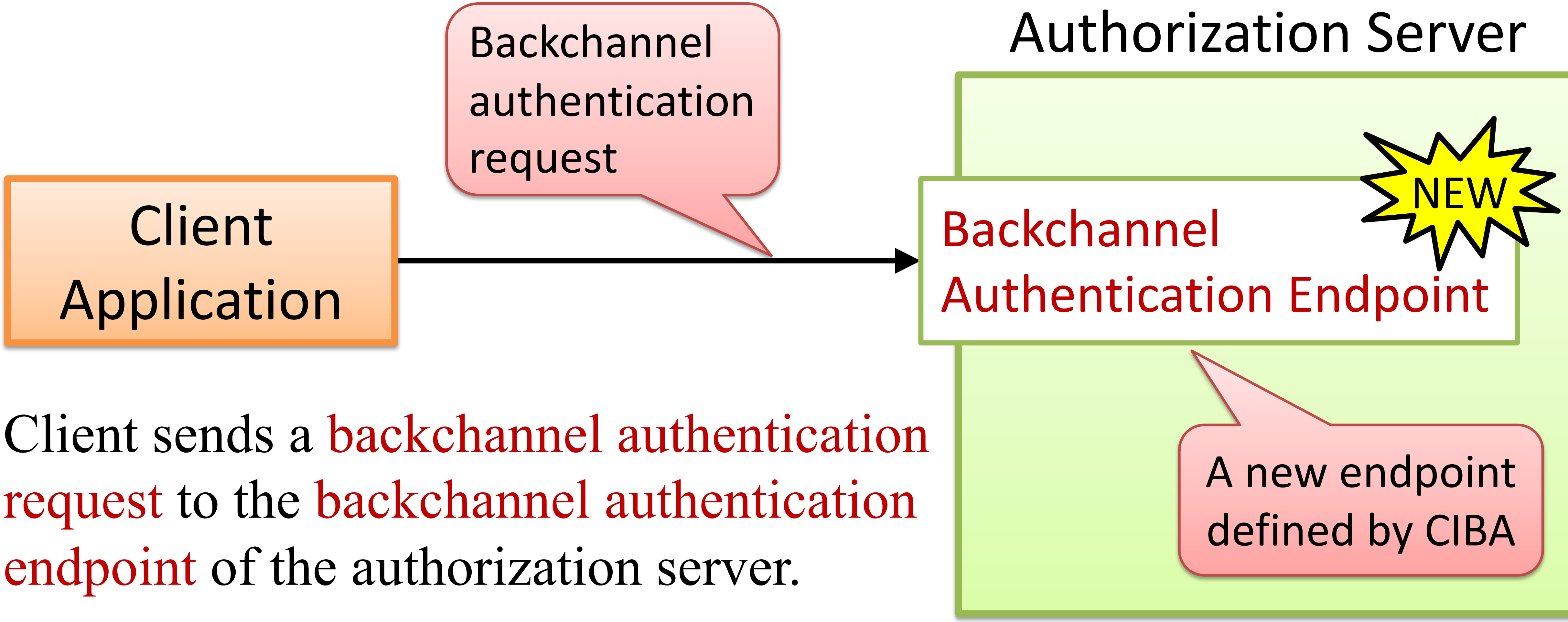
CIBA (Client Initiated Backchannel Authentication) defines new authorization flows.

1	CIBA POLL Mode
2	CIBA PING Mode
3	CIBA PUSH Mode

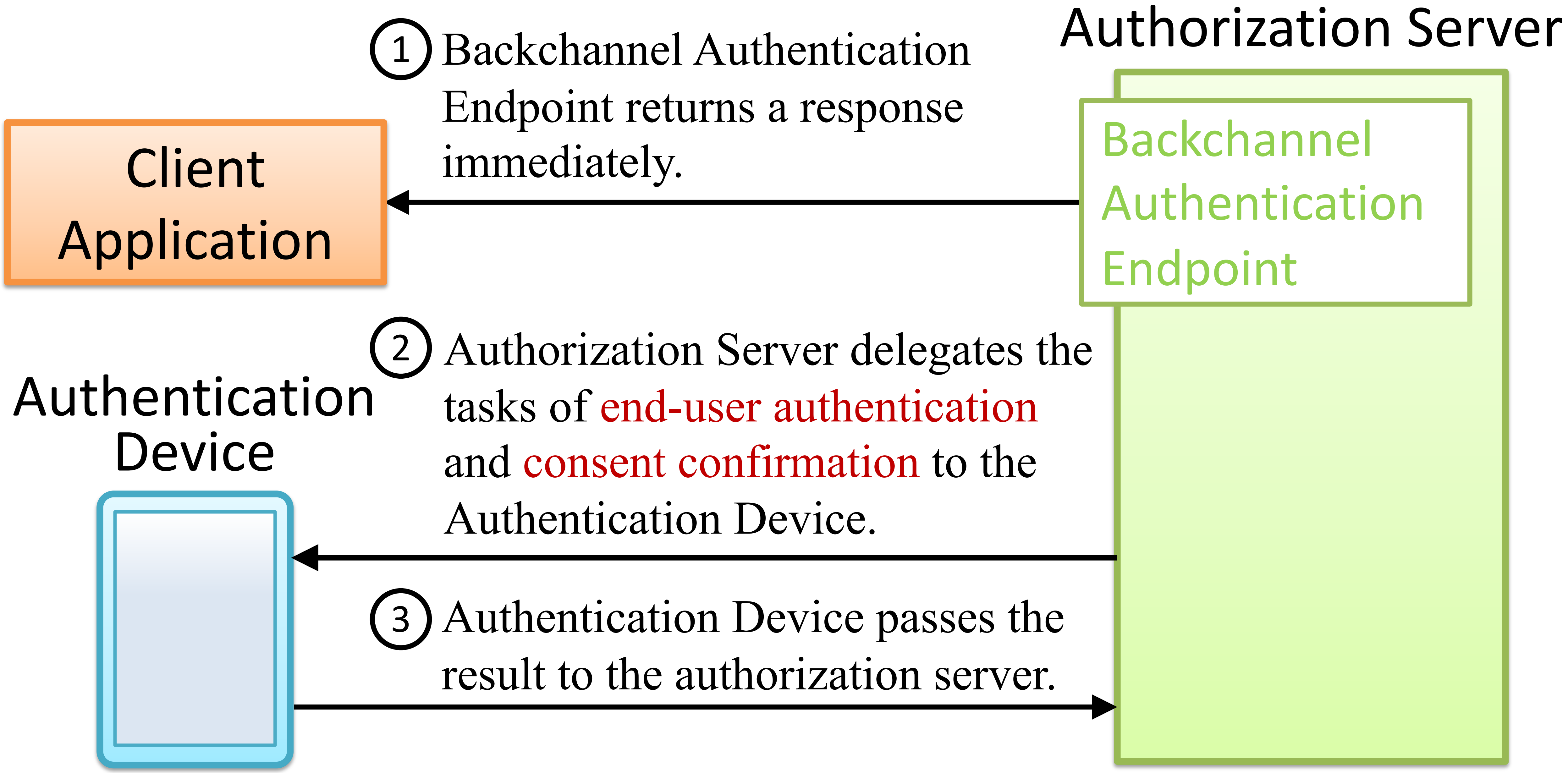
The flows enable to separate the authentication device on which a user is authenticated and API authorization is granted from the consumption device on which a client application that calls APIs runs.


The system is asking for the permission. Approve?
authorization server that supports CIBA
5
4
authentication device
backchannel authentication request
3 asks for the permission
grants the permission 6
Purchase ABC.
1
2
7 calls APIs
smart speaker
backend system
consumption device
resource server that provides APIs

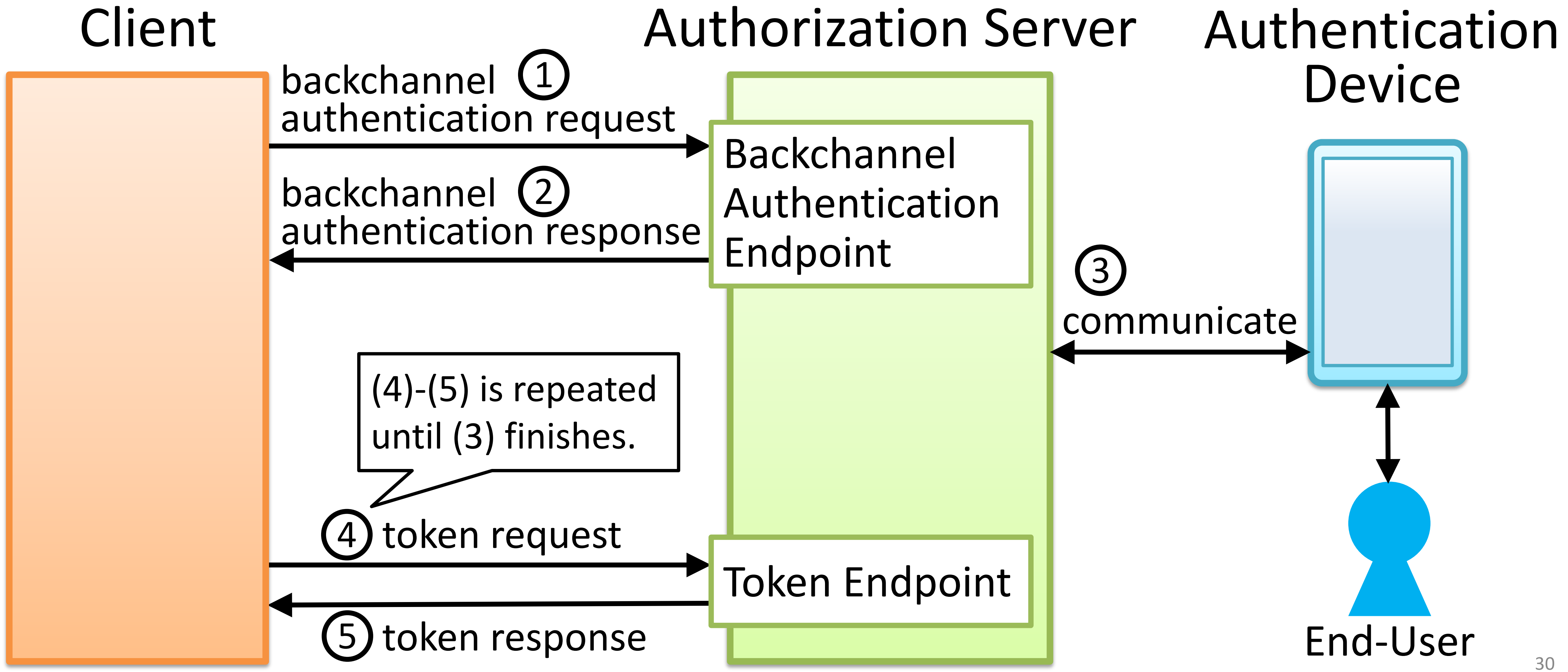
Every CIBA flow starts from a backchannel authentication request.


Backchannel authentication request
Authorization Server
Client Application
Backchannel Authentication Endpoint
NEW
A new endpoint defined by CIBA

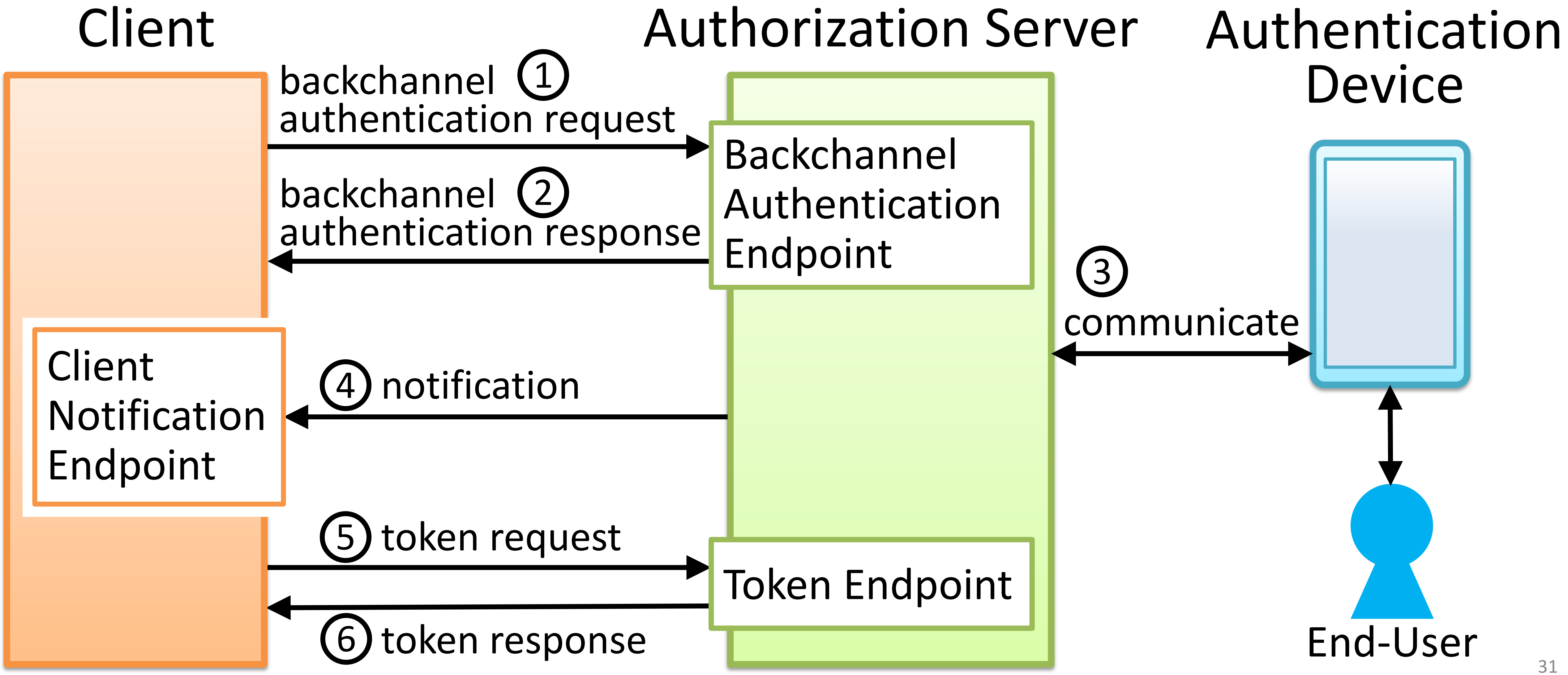
Client sends a backchannel authentication request to the backchannel authentication endpoint of the authorization server.


Authorization Server
Backchannel Authentication Endpoint
Client Application
Authentication Device
① Backchannel Authentication Endpoint returns a response immediately.
② Authorization Server delegates the tasks of end-user authentication and consent confirmation to the Authentication Device.
③ Authentication Device passes the result to the authorization server.

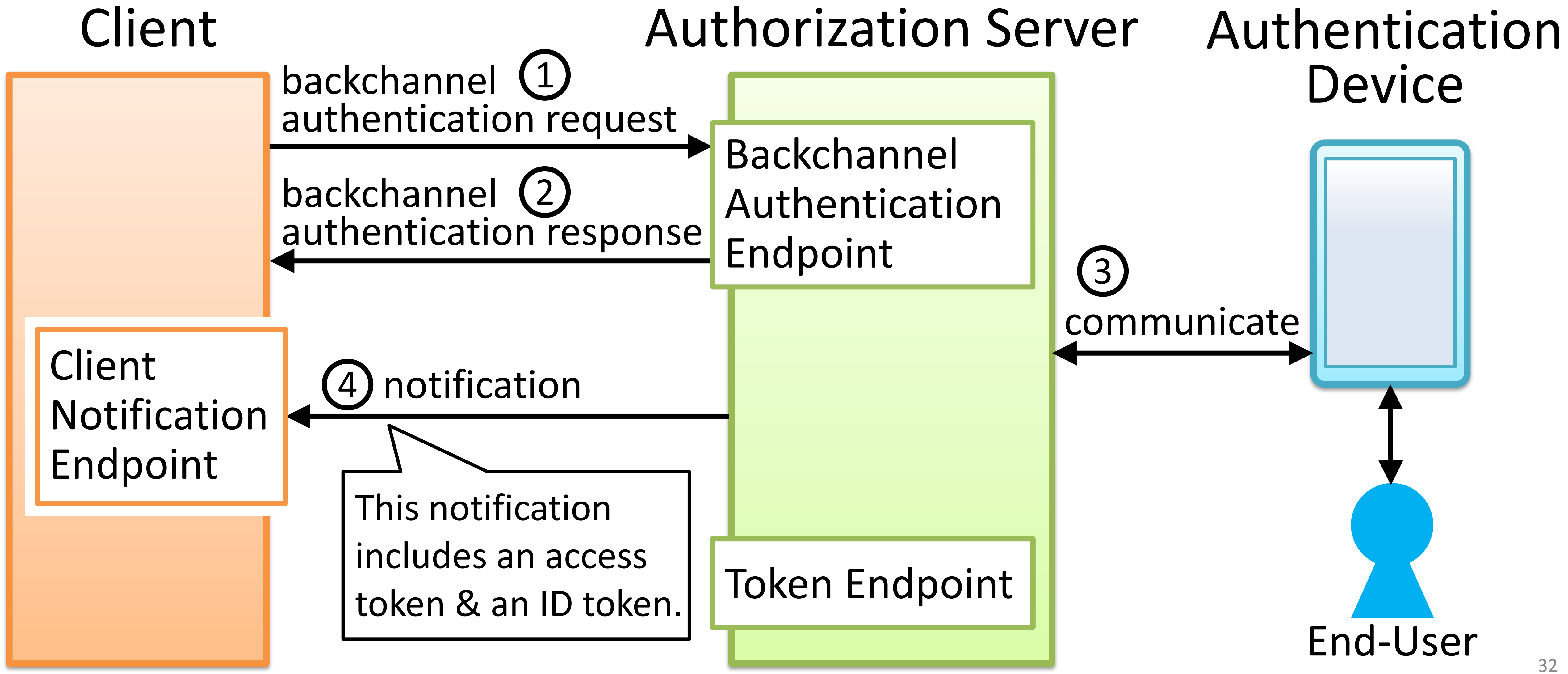
CIBA POLL mode


Client
Authorization Server
Authentication Device
backchannel ① authentication request
backchannel ② authentication response
Backchannel Authentication Endpoint
③ communicate
(4)-(5) is repeated until (3) finishes.
④ token request
Token Endpoint
⑤ token response
End-User

CIBA PING mode


Client
Authorization Server
Authentication Device
backchannel ① authentication request
Backchannel Authentication Endpoint
backchannel ② authentication response
③ communicate
Client Notification Endpoint
④ notification
⑤ token request
Token Endpoint
⑥ token response
End-User

CIBA PUSH mode


Client
Authorization Server
Authentication Device
① backchannel authentication request
② backchannel authentication response
Backchannel Authentication Endpoint
③ communicate
Client Notification Endpoint
④ notification
This notification includes an access token & an ID token.
Token Endpoint
End-User

Thank You

Contact

https://www.authlete.com/contact/

General	info@authlete.com
Sales	sales@authlete.com
PR	pr@authlete.com
Technical	support@authlete.com

@authlete

OAuth 2.0
OpenID Connect
Authorization Focused • Reliable and Scalable • Developer Friendly
Faster Time to Market • Choice of Hosting Options • Broad Usage
Integrates with any Authentication methods
API Security
OpenID
AUTHLETE

References

AUTHLETE

Specifications

- ✓ **Financial-grade API, Part 1: Read-Only Security Profile**
  https://openid.net/specs/openid-financial-api-part-1-ID2.html
- ✓ **Financial-grade API, Part 2: Read and Write API Security Profile**
  https://openid.net/specs/openid-financial-api-part-2-ID2.html
- ✓ **Financial-grade API: JWT Secured Authorization Response Mode for OAuth 2.0 (JARM)**
  https://openid.net/specs/openid-financial-api-jarm-ID1.html
- ✓ **OpenID Connect Client Initiated Backchannel Authentication Flow – Core 1.0**
  https://openid.net/specs/openid-client-initiated-backchannel-authentication-core-1_0.html
- ✓ **OAuth 2.0 Mutual TLS Client Authentication and Certificate Bound Access Tokens**
  https://datatracker.ietf.org/doc/draft-ietf-oauth-mtls/
- ✓ **RFC 7523 – JSON Web Token (JWT) Profile for OAuth 2.0 Client Authentication and Authorization Grants**
  https://tools.ietf.org/html/rfc7523

Articles

- ✓ **Financial-grade API (API), explained by an implementer**
  https://medium.com/@darutk/financial-grade-api-fapi-explained-by-an-implementer-d09fcf2ff932
- ✓ **"CIBA", a new authentication/authorization technology in 2019, explained by an implementer**
  https://medium.com/@darutk/ciba-a-new-authentication-authorization-technology-in-2019-explained-by-an-implementer-d1e0ac1311b4

Others

- ✓ **Financial-grade API (FAPI) Working Group**
  https://openid.net/wg/fapi/
- ✓ **Official Conformance Suite**
  https://gitlab.com/openid/conformance-suite