Technical Documentation

BCA API OAuth & Signature

Version 0.1.4

Release Date: January 25, 2018

Document Version

Date	Doc Version	Description	PIC
25 January 2018	0.1.4	- Update HMAC-SHA256 Result for Scenario 4 on Page 10	BIL
28 September 2017	0.1.3	- Move port 443 to description column	BIL
19 September 2017	0.1.2	- Update port for endpoint UAT to 443	BIL
09 August 2017	0.1.1	- Update port for endpoint production to 443	ABE
20 April 2017	0.1.0	- Create this document as separate file from API Service Tech. Doc. - Add notes for HexEncode usage (page 6) - Add notes about how to handle comma characters when creating signature (page 7) - Add section “Signature How To” (page 10)	ABE

Contents

Introduction

Overview of BCA Corporate Banking API:

Method	Endpoint	Usage
POST	/api/oauth/token	Get access token

URL that you can access:

URL	Usage
https://devapi.klikbca.com	BCA API UAT environment With default port of HTTPS (443)
https://api.klikbca.com	BCA API production environment With default port of HTTPS (443)

Authorization

The BCA Corporate Banking API is using OAuth 2 as the authorization framework. To access all the services you'll need the access token with `grant_type=client_credentials`. To get the access token, you need to be authorized by `client_id` and `client_secret`. To learn more about the OAuth 2 authorization framework you can read the rfc6749 documentation (https://tools.ietf.org/html/rfc6749).

Access Token Request

```
POST /api/oauth/token HTTP/1.1
Host: server.example.com
Authorization: Basic Base64(client_id:client_secret)
Content-Type: application/x-www-form-urlencoded

grant_type=client_credentials
```

Sample Request:

```
curl https://api.klikbca.com/api/oauth/token \
```

```
   -H "Content-Type: application/x-www-form-urlencoded" \
   -H "Authorization: Basic
      jk5ZTkyYzgtYzAzNC00YmNhLWE0OTAtYWM4NGI0YTZiMjQxOjNmYWIwNGI1LWM4ODctNGZmM
      i05OGNkLTE1YjJmYTcyNzA1NA==" \
   -d "grant_type=client_credentials"
```

Sample Response:

```
{
   "access_token":"2YotnFZFEjr1zCsicMWpAA",
   "token_type":"bearer",
   "expires_in":3600,
   "scope":"resource.WRITE resource.READ"
}
```

Headers

To successfully communicate with BCA Banking API, you **must provide** the following headers **in every API request** :

Name	Type	Description
Authorization	AN	OAuth 2 token Format value : `Bearer {access_token}`
Content-Type	AN	Content of you request body e.g. `application/json`
Origin	url	Origin domain e.g. yourdomain.com
X-BCA-Key	AN	Your API key generated by BCA
X-BCA-Timestamp	yyyy-MM-ddThh:mi:ss.sssTZD (ISO 8601)	Timestamp generated by merchant in ISO 8601 e.g. “2015-04-29T09:54:00.234Z”
X-BCA-Signature	AN	Signature, please refer to Signature section

Signature

Signature is used by BCA to verify that your request is not altered by attackers.

The outline of the HMAC validation process is as follows:

1. Retrieve Timestamp from HTTP Header (`X-BCA-Timestamp`)
2. Retrieve the API Key form HTTP Header (X-BCA-Key)
3. Lookup the API Secret corresponding to the received key in internal store
4. Retrieve client HMAC from HTTP Header lowercase hexadecimal format (`X-BCA-Signature`)
5. Calculate HMAC using the API Secret as the HMAC secret key
6. Compare client HMAC with calculated HMAC

If HMAC hash comparison is invalid API Gateway will return a HTTP 400 error code together with the following error message on JSON format:

```
{
	"ErrorCode" : "...",
	"ErrorMessage" : {
		"Indonesian": "HMAC tidak cocok",
		"English": "HMAC mismatch"
	}
}
```

If the HMAC calculation is successful and the calculated value matches the value received from the client, the signature is considered valid.

Generate Signature

SHA-256 HMAC is used to generate the signature with your API secret as the key.

```
Signature = HMAC-SHA256(apiSecret, StringToSign)
```

The `StringToSign` will be a colon-separated list derived from some request data as below :

```
StringToSign = HTTPMethod+":"+RelativeUrl+":"+AccessToken+":"+
  Lowercase(HexEncode(SHA-256(RequestBody)))+":"+Timestamp
```

HexEncode are optional to use, use it if the `SHA-256` returns a binary stream.

Details about the data used to derived The `StringToSign` is explained in the next sections.

HTTP Method

- `HTTP Method` is HTTP Method such as `GET`, `POST`, `PUT`, `PATCH`, `DELETE`.
- HTTP method must be given in upper case.

Relative URL

- Relative URL is the URL after the hostname & port number.
- Relative URL also includes the query string and must begin with a slash character.
  Example :

Full URL	Relative URL
https://example.com/api/v2/sample?param1=value1&param2=value2	/api/v2/sample?param1=value1&param2=value2
https://example.com or https://example.com/	/

- The Relative URL must be URI-encoded according to the following rules:
  1. Do not URI-encode forward slash (/) if it was used as path component.
  2. Do not URI-encode question mark (?), equals sign (=), and ampersand (&) if they were used as query string component: as separator between the path and query string, between query parameter and its value, and between each query parameter and value pairs.
  3. Do not URI-encode these characters: A-Z, a-z, 0-9, hyphen (-), underscore (_), period (.), and tilde (~) which are defined as unreserved characters in RFC 3986.
     As for RFC 3986, means that comma that appear in the value of query params or path params should be encoded too when generating Signature.
  4. Percent-encode all other characters not meeting the above conditions using the format: %XY, where X and Y are hexadecimal characters (0-9 and uppercase A-F). For example, the space character must be encoded as %20 (not using '+', as some encoding schemes do) and extended UTF-8 characters must be in the form %XY%ZA%BC.
- The query string parameters must be re-ordered according to the following rules:
  1. Sorted by parameter name lexicographically
  2. If there are two or more parameters with the same name, sort them by parameter values.

     Example :

Relative URL	Sorted Relative URL
/api/v2/sample?A-param=value1&Z-param=value2&B-param=value3	/api/v2/sample?A-param=value1&B-param=value3&Z-param=value2

AccessToken

- AccessToken is an OAuth 2 access token retrieved from the HTTP "Authorization" header.

RequestBody

- RequestBody need to be hashed with SHA-256.
- If the RequestBody is empty, set it to empty string.
- RequestBody should be canonicalized before computing the SHA-256 hash.
- The canonicalization of the request body is performed according to the following rules:
  1. All carriage return characters, "\r", are stripped
  2. All line feed characters, "\n", are stripped
  3. All tab characters, "\t", are stripped
  4. All whitespace characters, " ", are stripped

  An example request JSON body like below:

```
{
	"Test1": "strVal",
	"Test2": 1
}
```

  Will look like below after canonicalization has been performed:

```
{"Test1":"strVal","Test2":1}
```

Timestamp

The timestamp must be presented in ISO8601 format (YYYY-MM-DDThh:mm:ss.sssTZD)

YYYY = four-digit year
MM = two-digit month (01=January, etc.)
DD = two-digit day of month (01 through 31)
T = literal 'T' as date and time separator
hh = two digits of hour (00 through 23) (am/pm NOT allowed)

mm = two digits of minute (00 through 59)
ss = two digits of second (00 through 59)
sss = three digits representing millisecond (000 through 999)
TZD = time zone designator (Z or +hh:mm or -hh:mm)

Signature How to

No.	Parameter	Scenario 1 (Simple request)	Scenario 2 (Path param value contains comma ",")	Scenario 3 (JSON body request with white space)	Scenario 4 (Multiple query param)
Data for Client					
A	HTTP Method	get	get	post	get
B	URL	/banking/v2/corporates/h2hauto009/accounts/0611104625	/banking/v2/corporates/h2hauto009/accounts/0611104625,0613106704	/banking/corporates/transfers	/banking/v2/corporates/h2hauto009/accounts/0611104625/statements?StartDate=2017-03-01&EndDate=2017-03-017
C	API Credential	b66925de-d8ec-476e-a170-6cf06c863b78	b66925de-d8ec-476e-a170-6cf06c863b78	b66925de-d8ec-476e-a170-6cf06c863b78	b66925de-d8ec-476e-a170-6cf06c863b78
D	API Credential Secret	efc71ced-b0e7-4b47-8270-3c24829764aa	efc71ced-b0e7-4b47-8270-3c24829764aa	efc71ced-b0e7-4b47-8270-3c24829764aa	efc71ced-b0e7-4b47-8270-3c24829764aa
E	API Key	34bec438-9911-494c-9e29-d0041f941eec	34bec438-9911-494c-9e29-d0041f941eec	34bec438-9911-494c-9e29-d0041f941eec	34bec438-9911-494c-9e29-d0041f941eec
F	API Key Secret	f6068d37-0fd8-456a-bced-61ac35af53da	f6068d37-0fd8-456a-bced-61ac35af53da	f6068d37-0fd8-456a-bced-61ac35af53da	f6068d37-0fd8-456a-bced-61ac35af53da
G	Access Token	gp9HjjEj813Y9JGoqwOeOPWbnt4CUpvIJbU1mMU4a11MNDZ7Sg5u9a	gp9HjjEj813Y9JGoqwOeOPWbnt4CUpvIJbU1mMU4a11MNDZ7Sg5u9a	gp9HjjEj813Y9JGoqwOeOPWbnt4CUpvIJbU1mMU4a11MNDZ7Sg5u9a	gp9HjjEj813Y9JGoqwOeOPWbnt4CUpvIJbU1mMU4a11MNDZ7Sg5u9a
H	Timestamp	2017-03-17T09:44:18.000+07:00	2017-03-17T09:44:18.000+07:00	2017-03-17T09:44:18.000+07:00	2017-03-17T09:44:18.000+07:00
I	Request Body	*empty*	*empty*	{ "CorporateID" : "H2HAUTO009", "SourceAccountNumber" : "0611104625", "TransactionID" : "00177914", "TransactionDate" : "2017-03-17", "ReferenceID" : "1234567890098765", "CurrencyCode" : "IDR", "Amount" : "175000000", "BeneficiaryAccountNumber" : "0613106704", "Remark1" : "Pencairan Kredit", "Remark2" : "1234567890098765" }	*empty*
Step Generate String to Sign	**Action**	**Result**	**Result**	**Result**	**Result**
J	Examine HTTP Method	**upper(A)**	GET	POST	GET
K	Examine URI	**uri_encode(B)**	/banking/v2/corporates/h2hauto009/accounts/0611104625	/banking/v2/corporates/h2hauto009/accounts/0611104625%2C0613106704	/banking/corporates/transfers	/banking/v2/corporates/h2hauto009/accounts/0611104625/statements?StartDate=2017-03-01&EndDate=2017-03-017
		sort_lexicography(B)	/banking/v2/corporates/h2hauto009/accounts/0611104625	/banking/v2/corporates/h2hauto009/accounts/0611104625%2C0613106704	/banking/corporates/transfers	/banking/v2/corporates/h2hauto009/accounts/0611104625/statements?EndDate=2017-03-017&StartDate=2017-03-01
L	Examine Requst Body	**canonicalized(I)**	""	""	{"CorporateID":"H2HAUTO009","SourceAccountNumber":"0611104625","TransactionID":"00177914","TransactionDate":"2017-03-17","ReferenceID":"1234567890098765","CurrencyCode":"IDR","Amount":"175000000","BeneficiaryAccountNumber":"0613106704","Remark1":"PencairanKredit","Remark2":"1234567890098765"}	""
		sha256(canonicalized(I))	e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855	e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855	50552692103b705cf3d0d0bda7b943df86ecc19ada6ae1bda44192e158f5cb0a	e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
		lower(sha256(canonicalized(I)))	e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855	e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855	50552692103b705cf3d0d0bda7b943df86ecc19ada6ae1bda44192e158f5cb0a	e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
M	Examine Timestamp	**to_char(H, 'YYYY-MM-DDThh:mm:ss.sssTZD')**	2017-03-17T09:44:18.000+07:00	2017-03-17T09:44:18.000+07:00	2017-03-17T09:44:18.000+07:00	2017-03-17T09:44:18.000+07:00
N	Construct String to Sign	J + ":" + K + ":" + G + ":" + L + ":" + M	GET:/banking/v2/corporates/h2hauto009/accounts/0611104625:gp9HjjEj813Y9JGoqwOeOPWbnt4CUpvIJbU1mMU4a11MNDZ7Sg5u9a:e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855:2017-03-17T09:44:18.000+07:00	GET:/banking/v2/corporates/h2hauto009/accounts/0611104625%2C0613106704:gp9HjjEj813Y9JGoqwOeOPWbnt4CUpvIJbU1mMU4a11MNDZ7Sg5u9a:e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855:2017-03-17T09:44:18.000+07:00	POST:/banking/corporates/transfers:gp9HjjEj813Y9JGoqwOeOPWbnt4CUpvIJbU1mMU4a11MNDZ7Sg5u9a:50552692103b705cf3d0d0bda7b943df86ecc19ada6ae1bda44192e158f5cb0a:2017-03-17T09:44:18.000+07:00	GET:/banking/v2/corporates/h2hauto009/accounts/0611104625/statements?EndDate=2017-03-017&StartDate=2017-03-01:gp9HjjEj813Y9JGoqwOeOPWbnt4CUpvIJbU1mMU4a11MNDZ7Sg5u9a:e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855:2017-03-17T09:44:18.000+07:00
Step Construct Signature	**Action**	**Result**	**Result**	**Result**	**Result**
O	Examine Secured Hashing	**HMAC-SHA256(F, N)**	85be817c55b2c135157c7e89f52499bf0c25ad6eeebe04a986e8c862561b19a5	6175d27fd8d03ddb806abfd2c3fd6e8271e862883ac0cb6383f823546d776c67	6dffdb3952eb45e4012a88594040ffde3bbdedfc97fe94c1a97749c4a7d2e5f5	22a901d2654178c797235357b39792a189e5dface71e7cea3c4dafccf1509401
Step Construct Request to API	**Parameter**	**Value**	**Value**	**Value**	**Value**
P	Examine URI	**URI to call**	B	B	B	B

Q	Examine Header	**Authorization**	Bearer G	Bearer G	Bearer G	Bearer G
		X-BCA-Key	E	E	E	E
		X-BCA-Signature	O	O	O	O
		X-BCA-Timestamp	H	H	H	H
R	Examine Body	**Body**	*empty*	*empty*	{ "CorporateID" : "H2HAUTO009", "SourceAccountNumber" : "0611104625", "TransactionID" : "00177914", "TransactionDate" : "2017-03-17", "ReferenceID" : "1234567890098765", "CurrencyCode" : "IDR", "Amount" : "175000000", "BeneficiaryAccountNumber" : "0613106704", "Remark1" : "**Pencairan Kredit**", "Remark2" : "1234567890098765" }	*empty*