Fayda Secured Biometric Interface Specification - SBI For Relaying Party Integrators

BETA VERSION

Introduction

Relying parties those integrate with Fayda Platform using biometrics data for Authentication and eKYC services needs to collect data from biometrics devices in a secured channel using defined standards. SBI is an interface that handles communication between biometric devices and relaying parties systems. Usually is a service that runs in the background. SBI avail its interface using Rest API protocol relying parties system need to call to those API as defined in this specification document.

Biometric device vendors must comply to this specification and develop their own SBI.

API specification version: **1.0**

Glossary of Terms

- **Fayda Devices**

  A hardware capable of capturing biometric information. All devices that collect biometric data for Fayda should operate within the specification of this document.

- **Device Provider**

  An entity that manufactures or imports the devices in their name. This entity should have legal rights to obtain an organization level digital certificate from the respective authority in the country.

- **FTM Provider**

  An entity that manufactures or guarantees the trustworthiness of the foundational trust module. This can be the device provider as well.

- **SBI 2.0 Certified Device / SBI 2.0 Device**

  A device certified as capable of performing all biometric functionalities (capture, processing), signing and encryption in line with this spec in its hardware trusted zone/FTM.

- **SBI 1.0 Certified Device / SBI 1.0 Device**

  A device certified as one where the biometric functionalities (capture, processing), signing and encryption is done on the host machine software zone as a separate service (protected from users or other OS-level applications) or at the device driver level.

- **FTM Provider Certificate**

  A digital certificate issued to the "Foundational Trust Provider". This certificate proves that the provider has successfully gone through the required Foundational Trust Provider evaluation. The entity is expected to keep this certificate in secure possession in an HSM. All the individual FTM trust certificates are issued using this certificate as the root. This certificate would be issued by the countries in conjunction with Fayda.

- **Device Provider Certificate**

  A digital certificate issued to the "Device Provider". This certificate proves that the provider has been certified for SBI 1.0/SBI 2.0 respective compliance. The entity is expected to keep this certificate in secure possession in an HSM. All the individual device trust certificates are issued using this certificate as the root. This certificate is issued by the countries in conjunction with Fayda.

- **Management Server**

  A server run by the device provider to manage the life cycle of the biometric devices.

- **FPS**

  Frames Per Second

- **Signature**

  All signature should be as per RFC 7515. Header - The attribute with "alg" set to RS256 and x5c set to base64urlencoded certificate. Payload - Byte array of the actual data, always represented as base64urlencoded. Signature - Base64urlencoded signature bytes

- **ISO format timestamp**

  ISO 8601 with format yyyy-mm-ddTHH:MM:ssZ (Example: 2020-12-08T09:39:37Z). This value should be in UTC (Coordinated Universal Time).

- **Registration**

  The process of applying for a Foundational Id.

- **Auth**

  The process of verifying one's identity.

- **KYC**

  Know Your Customer is the process of providing consent to perform profile verification and update.

Technical API Specification

The section explains the necessary details of the biometric device connectivity, accessibility, discover-ability and protocols used to build and communicate with the device.

The device should implement only the following set of APIs. All the API's are independent of the physical layer and the operating system, with the invocation being different across operating systems. While the operating system names are defined in this spec a similar technology can be used for unspecified operating systems. It is expected that the device service ensures that the device is connected locally to the host.

API

All the device API will be based on the HTTP specification. The device always binds to any of the available ports ranging from 4501 - 4600. The IP address used for binding has to be 127.0.0.1 and not localhost.

The applications that require access to Fayda devices could discover them by sending the HTTP request to the supported port range. We will call this port the device_service_port in the rest of the document.

Device Discovery Request

Device discovery would be used to identify Fayda compliant devices in a system by the applications. The protocol is designed as a simple plug and play with all the necessary abstraction to the specifics.

Discovery API

Windows/Linux

HTTP Request:

```
SBIDISC http://127.0.0.1:<device_service_port>/device
HOST: 127.0.0.1: <device_service_port>
EXT: <app name>
```

HTTP Response:

```
HTTP/1.1 200 OK
CACHE-CONTROL:no-store
LOCATION:http://127.0.0.1:<device_service_port>
Content-Length: length in bytes of the body
Content-Type: application/json
Connection: Closed
```

- The payloads are JSON in both cases and are part of the body.
- CallbackId would be set to the `http://127.0.0.1:<device_service_port>/`. So, the caller will use the respective HTTP verb/method and the URL to call the service.

Message Body

Device Discovery Request

```
{
  "type": "type of the device",
  "specVersion": "SBI specification version"
}
```

Allowed Values

Parameters	Description
type	This represents the type of device. Allowed values here are "Biometric Device", "Finger", "Face" or "Iris". "Biometric Device" - is a special type and used in case you are looking for "any" biometric device.
specVersion	This represents the spec version of the SBI. This is a mandatory parameter in SBI 1.0, but when requested the response the SBI supporting this spec version should respond to the discovery call.

Device Discovery Response

```
[
  {
    "serialNo": "Printed Serial Number of the device",
    "deviceStatus": "Device status",
    "certification": "Certification level",
    "serviceVersion": "Device service version",
    "deviceSubId": ["Array of supported device sub Ids"],
    "callbackId": "Base URL to reach to the device",
    "digitalId": "Unsigned Digital ID of the device",
    "specVersion": ["Array of supported SBI specification version"],
    "purpose": "Auth  or Registration",
    "error": {
      "errorCode": "101",
      "errorInfo": "Invalid JSON Value Type For Discovery.."
    }
  },
  ...
]
```

Allowed Values

Parameters	Description
deviceStatus	Allowed values are "Ready", "Busy", "Not Ready", and "Not Registered". "Not Registered" denotes that the device does not have a valid certificate issued by the device provider for the device.
certification	Allowed values are "SBI 1.0" or "SBI 2.0" based on the level of certification.
serviceVersion	Device service version.
serialNo	This represents the serial number of the device. This value should be the same as printed on the device

deviceSubId	Allowed values are 0, 1, 2 or 3. The device sub id could be used to enable a specific module in the scanner appropriate for a biometric capture requirement. Device sub id is a simple index that always starts with 1 and increases sequentially for each sub-device present. In the case of Finger/Iris it's 1 for left slap/iris, 2 for right slap/iris and 3 for two thumbs/irises. The device sub id should be set to 0 if we don't know any specific device sub id (0 is not applicable for fingerprint slap).
callbackId	This differs as per the OS. In the case of Linux and Windows operating systems, it is an HTTP URL. In the case of android, it is the intent name. In IOS, it is the URL scheme. The callback URL takes precedence over future request as a base URL.
digitalId	Digital ID as per the Digital ID definition but it will not be signed.
specVersion	Array of supported SBI specification version. The array element Zero will always contain the spec version using which the response is created.
purpose	Purpose of the device in the Fayda ecosystem. Allowed values are "Auth" or "Registration".
error	Relevant errors as defined under the Error Codes of this document.
error.errorCode	Standardized error code.
error.errorInfo	Description of the error that can be displayed to the end-user. It should have multi-lingual support.

- The response is an array that we could have a single device enumerating with multiple biometric options.
- The service should ensure to respond only if the type parameter matches the type of device or the type parameter is a "Biometric Device".

Device Info Request

The device information API would be used to identify the Fayda compliant devices and their status by the applications.

Windows/Linux

HTTP Request:

```
SBIINFO http://127.0.0.1:<device_service_port>/info
HOST: 127.0.0.1:<device_service_port>
EXT: <app name>
```

HTTP Response:

```
HTTP/1.1 200 OK
CACHE-CONTROL:no-store
LOCATION:http://127.0.0.1:<device_service_port>
Content-Length: length in bytes of the body
Content-Type: application/json
Connection: Closed
```

The payloads are JSON in both cases and are part of the body.

Message Body

Device Info Request

```
{
  "type": "type of the device",
  "specVersion": "SBI specification version"
}
```

Allowed Values

Parameters	Description
type	This represents the type of device. This is a mandatory parameter in SBI 1.0. Allowed values here are "Biometric Device", "Finger", "Face" or "Iris". "Biometric Device" - is a special type and used in case you are looking for "any" biometric device.
specVersion	This represents the spec version of the SBI. This is a mandatory parameter in SBI 1.0, but when requested the response the SBI supporting this spec version should respond to the discovery call.

Device Info Response

```
[
  {
    "deviceInfo": {
      "deviceStatus": "Current status",
      "serialNo": "Printed Serial Number of the device",
      "firmware": "Firmware version",
      "certification": "Certification level",
      "serviceVersion": "Device service version",
      "deviceSubId": ["Array of supported device sub Ids"],
      "callbackId": "BaseURL to reach to the device",
      "digitalId": "Signed digital id as described in the digital id
section of this document.",
      "env": "Target environment",
      "purpose": "Auth  or Registration",
      "specVersion": ["Array of supported SBI specification version"],
    },
    "error": {
      "errorCode": "106",
      "errorInfo": "Device not found"
    }
  }
  ...
]
```

The final JSON is signed with the JSON Web Signature using the device key.

So the API would respond in the following format,

```
[
  {
    "deviceInfo": "base64urlencode(header).base64urlencode(payload).
base64urlencode(signature)"
    "error": {
      "errorCode": "106",
      "errorInfo": "Device not found"
    }
  }
]
```

Allowed Values

Parameters	Description
deviceInfo	The deviceInfo object is sent as JSON Web Token (JWT). For devices that do not have a valid certificate issued by the device provider, the deviceInfo will be unsigned. For devices that are registered, the deviceInfo will be signed using the device key.
deviceInfo.deviceStatus	This is the current status of the device. Allowed values are "Ready", "Busy", "Not Ready", and "Not Registered". "Not Registered" denotes that the device does not have a valid certificate issued by the device provider against the device.
deviceInfo.firmware	Exact version of the firmware (SBI 2.0). In the case of SBI 1.0 this is the same as serviceVersion.
deviceInfo.certification	Allowed values are "SBI 1.0" or "SBI 2.0" based on the level of certification.
deviceInfo.serviceVersion	Version of the SBI specification that is supported.
deviceInfo.serialNo	This represents the serial number of the device. This value should be the same as printed on the device.
deviceInfo.deviceSubId	Allowed values are 0, 1, 2 or 3. The device sub id could be used to enable a specific module in the scanner appropriate for a biometric capture requirement. Device sub id is a simple index that always starts with 1 and increases sequentially for each sub-device present. In the case of Finger/Iris, it's 1 for left slap/iris, 2 for right slap/iris and 3 for two thumbs/irises. The device sub id should be set to 0 if we don't know any specific device sub id (0 is not applicable for fingerprint slap).
deviceInfo.callbackId	This differs as per the OS. In the case of Linux and Windows operating systems, it is an HTTP URL. In the case of android, it is the intent name. In IOS, it is the URL scheme. The callback URL takes precedence over future requests as a base URL.
deviceInfo.digitalId	The digital id as per the digital id definition. For SBI 1.0 devices that are yet to obtain a certificate from the device provider, the digitalId will be unsigned. For SBI 1.0 devices with valid certificates issued by the provider, the digital id will be signed using the device key. For SBI 2.0 devices, the digital id will be always signed using the FTM key.
deviceInfo.env	This represents the target environment. For devices that are not registered, the environment is "None". For devices that are registered, send the environment in which it is registered. Allowed values are "Staging", "Developer", "Pre-Production" or "Production".
deviceInfo.purpose	Purpose of the device in the Fayda ecosystem. Allowed values are "Auth" or "Registration".

deviceInfo.specVersion	Array of supported SBI specification version. The array element Zero will always contain the spec version using which the response is created.
error	Relevant errors as defined under the Error Section of this document.
error.errorCode	Standardized error code.
error.errorInfo	Description of the error that can be displayed to the end-user. It should have multi-lingual support.

- The response is an array that we could have a single device enumerating with multiple biometric options.
- The service should ensure to respond only if the type parameter matches the type of device or the type parameter is a "Biometric Device".

Capture

The capture request would be used to capture a biometric from Fayda compliant devices by the applications for authentication. The capture call will respond with success to only one call at a time. So, in case of a parallel call, the device info details are sent with status as "Busy".

Capture Request

The applications that want to capture biometric data from a Fayda device could do so by sending the HTTP request to the supported port range.

HTTP Request:

```
CAPTURE http://127.0.0.1:<device_service_port>/capture
HOST: 127.0.0.1: <apps port>
EXT: <app name>
```

HTTP Response:

```
HTTP/1.1 200 OK
CACHE-CONTROL:no-store
LOCATION:http://127.0.0.1:<device_service_port>
Content-Length: length in bytes of the body
Content-Type: application/json
Connection: Closed
```

The payloads are JSON in both cases and are part of the body.

Message Detail

Capture Request Message

```
{
  "env": "Target environment",
  "purpose": "Auth",
  "specVersion": "Expected version of the SBI spec",
  "timeout" : "Timeout for capture",
  "captureTime": "Time of capture request in ISO format",
  "domainUri": "URI of the auth server",
  "transactionId": "Transaction Id for the current capture",
  "bio": [
    {
      "type": "Type of the biometric data",
      "count":  "Finger/Iris count, in case of face max is set to 1",
      "bioSubType": ["Array of subtypes"],
      "requestedScore": "Expected quality score that should match to
complete a successful capture. This value will be scaled from 0 - 100
for NFIQ v1.0. The logic for scaling is mentioned below.",
      "serialNo": "Physical Serial Number of the device",
      "deviceSubId": "Specific Device Sub Id",
      "previousHash": "Hash of the previous block"
    }
  ],
  "customOpts": {
    //max of 50 key-value pair.
    //This is so that vendor-specific parameters can be sent if
necessary.
    //The values cannot be hardcoded and have to be configured by the
apps server and should be modifiable upon need by the applications.
    //Vendors are free to include additional parameters and fine-tuning
parameters.
    //None of these values should go undocumented by the vendor.
    //No sensitive data should be available in the customOpts.
  }
}
```

Count value should be driven by the count of the bioSubType for Iris and Finger. For Face, there will be no bioSubType but the count should be "1"

Allowed Values

Parameters	Description
env	This represents the target environment. Allowed values are "Staging", "Developer", "Pre-Production" or "Production".
purpose	The purpose of the device in the Fayda ecosystem. For devices that are not registered the purpose is empty. The allowed value is "Registration".
specVersion	Expected version of SBI specification.

timeout	Max time the app will wait for the capture. It's expected that the API will respond before timeout if the requested score is met, or with the best frame at the timeout. All timeouts are in milliseconds.
captureTime	Time of capture in ISO format. The time is as per the requesting application.
domainUri	URI of the authentication server. This can be used to federate across multiple providers or countries or unions.
transactionId	Unique ID for the transaction. This is an internal Id to the application that's providing the service. The different id should be used for every transaction. So, even if the transaction fails after auth we expect this number to be unique.
bio.type	Allowed values are "Finger", "Iris" or "Face".
bio.count	Number of biometric data that is collected for a given type. The device should validate and ensure that this number is in line with the type of biometric that's captured.
bio.bioSubType	For Finger: ["Left IndexFinger", "Left MiddleFinger", "Left RingFinger", "Left LittleFinger", "Left Thumb", "Right IndexFinger", "Right MiddleFinger", "Right RingFinger", "Right LittleFinger", "Right Thumb", "UNKNOWN"] For Iris: ["Left", "Right", "UNKNOWN"] For Face: No bioSubType
bio.requestedScore	Upon reaching the quality score the biometric device is expected to auto-capture the image. If the requested score is not met, until the timeout, the best frame during the capture sequence must be captured /returned. This value will be scaled from 0 - 100 for NFIQ v1.0. The logic for scaling is mentioned below.
bio.serialNo	This represents the serial number of the device. This value should be the same as printed on the device.
bio.deviceSubId	Allowed values are 0, 1, 2 or 3. The device sub id could be used to enable a specific module in the scanner appropriate for a biometric capture requirement. Device sub id is a simple index that always starts with 1 and increases sequentially for each sub-device present. In the case of Finger/Iris, it's 1 for left slap/iris, 2 for right slap/iris and 3 for two thumbs/irises. The device sub id should be set to 0 if we don't know any specific device sub id (0 is not applicable for fingerprint slap). Wherever possible SBI must detect if the placement of biometrics is not in sync with the deviceSubId. For example, if the deviceSubId is selected as 1 and if a right slap is presented instead of left, SBI must provide appropriate messages.
bio.previousHash	For the first capture the previousHash is SHA256 hash of an empty UTF-8 string. From the second capture the previous capture's "hash" is used as input. This is used to chain all the captures across modalities so all captures have happened for the same transaction and during the same time.
customOpts	In case, the device vendor wants to send additional parameters they can use this to send key-value pairs if necessary. The values cannot be hardcoded and have to be configured by the apps server and should be modifiable upon need by the applications. Vendors are free to include additional parameters and fine-tuning the process. None of these values should go undocumented by the vendor. No sensitive data should be available in the customOpts.

NFIQ v1.0 on a scale of 0-100 (quality score).

Scale	**NFIQ v1.0**
81 - 100	1
61 - 80	2

41 - 60	3
21 - 40	4
0 - 20	5

"bio.bioSubType" is a mandatory parameter for Capture request. For cases where “any” biometrics are expected, an array of UNKNOWN can be passed equally to the count specified in bio.count.

The SBI must make sure of the following,

- Must not allow the capture of the same biometrics segment wherever possible.
- In case of fingerprint, if a multi-finger scanner is used, and if bioSubType is passed as UNKNOWN, capture must return fingers in the order starting from IndexFinger to LittleFinger.

Capture Response Message

```
{
  "biometrics": [
    {
      "specVersion": "SBI spec version",
      "data": { //data block in JWT format signed using device key
        "digitalId": "Digital Id as described in this document signed
using FTM key (SBI 2.0)",
        "deviceServiceVersion": "SBI version",
        "bioType": "Finger",
        "bioSubType": "UNKNOWN",
        "purpose": "Auth",
        "env": "Target environment",
        "domainUri": "URI of the auth server",
        "bioValue": "Encrypted with session key and base64urlencoded
biometric data",
        "transactionId": "Unique transaction id",
        "timestamp": "Current datetime in ISO format",
        "requestedScore": "Floating point number to represent the
minimum required score for the capture",
        "qualityScore": "Floating point number representing the score
for the current capture"
      },
      "hash": "sha256 in hex format in upper case (previous "hash" +
sha256 hash of the current biometric ISO data before encryption)",
      "sessionKey": "Session key used for encrypting bioValue,
encrypted with Fayda public key (dynamically selected based on the URI)
and base64urlencoded",
      "thumbprint": "SHA256 representation of the certificate (HEX
encoded) that was used for encryption of session key. All texts to be
treated as uppercase without any spaces or hyphens",
      "error": {
        "errorCode": "101",
        "errorInfo": "Invalid JSON Value"
      },
      "additionalInfo": {
        //Additional information can be sent by the SBI in key value
```

```
pair.
        //max of 50 key value pair.
        //Vendors are free to include any number of additional
parameters.
        //None of these values should go undocumented by the vendor.
        //No sensitive data should be available here.
      }
    },
    {
      "specVersion" : "SBI spec version",
      "data": { //data block in JWT format signed using device key
        "digitalId": "Digital Id as described in this document signed
using FTM key (SBI 2.0)",
        "deviceServiceVersion": "SBI version",
        "bioType": "Finger",
        "bioSubType": "Left IndexFinger",
        "purpose": "Auth",
        "env": "Target environment",
        "domainUri": "URI of the auth server",
        "bioValue": "Encrypted with session key and base64urlencoded
biometric data",
        "transactionId": "Unique transaction id",
        "timestamp": "Current datetime in ISO format",
        "requestedScore": "Floating point number to represent the
minimum required score for the capture",
        "qualityScore": "Floating point number representing the score
for the current capture"
      },
      "hash": "sha256 in hex format in upper case (previous "hash" +
sha256 hash of the current biometric ISO data before encryption)",
      "sessionKey": "Session key used for encrypting bioValue,
encrypted with Fayda public key (dynamically selected based on the URI)
and base64urlencoded",
      "thumbprint": "SHA256 representation of the certificate (HEX
encoded) that was used for encryption of session key. All texts to be
treated as uppercase without any spaces or hyphens",
      "error": {
        "errorCode": "101",
        "errorInfo": "Invalid JSON Value"
      },
      "additionalInfo": {
        //Additional information can be sent by the SBI in key value
pair.
        //max of 50 key value pair.
        //Vendors are free to include any number of additional
parameters.
        //None of these values should go undocumented by the vendor.
        //No sensitive data should be available here.
      }
    }
```

```
    ]
}
```

Allowed Values

Parameters	Description
specVersion	Version of the SBI specification using which the response was generated.
data	The data object is sent as JSON Web Token (JWT). The data block will be signed using the device key.
data.digitalId	The digital id as per the digital id definition in JWT format. For SBI 1.0 devices, the digital id will be signed using the device key. For SBI 2.0 devices, the digital id will be signed using the FTM key.
data.deviceServiceVersion	SBI version
data.bioType	Allowed values are "Finger", "Iris" or "Face".
data.bioSubType	For Finger: ["Left IndexFinger", "Left MiddleFinger", "Left RingFinger", "Left LittleFinger", "Left Thumb", "Right IndexFinger", "Right MiddleFinger", "Right RingFinger", "Right LittleFinger", "Right Thumb", "UNKNOWN"] For Iris: ["Left", "Right", "UNKNOWN"] For Face: No bioSubType
data.purpose	The purpose of the device in the Fayda ecosystem. Allowed value is "Auth".
data.env	The target environment. Allowed values are "Staging", "Developer", "Pre-Production" or "Production".
data.domainUri	URI of the authentication server. This can be used to federate across multiple providers or countries or unions.
data.bioValue	Biometric data is encrypted with random symmetric (AES GCM) key and base-64-URL encoded. For symmetric key encryption of bioValue, (biometrics.data.timestamp XOR transactoinId) is computed and the last 16 bytes and the last 12 bytes of the results are set as the aad and the IV(salt) respectively. Look at the Authentication document to understand more about encryption.
data.transactionId	Unique transaction id sent in request
data.timestamp	Time as per the biometric device. Note: The biometric device is expected to sync its time from the management server at regular intervals so accurate time could be maintained on the device.
data.requestedScore	Floating point number to represent the minimum required score for the capture.
data.qualityScore	Floating point number representing the score for the current capture.
hash	sha256 in hex format in upper case (previous "hash" + sha256 hash of the current biometric ISO data before encryption)
sessionKey	The session key (used for the encrypting of the bioValue) is encrypted using the Fayda public certificate with RSA/ECB/OAEPWITHSHA-256ANDMGF1PADDING algorithm and then base64-URL-encoded.
thumbprint	SHA256 representation of the certificate (HEX encoded) that was used for encryption of session key. All texts to be treated as uppercase without any spaces or hyphens.
error	Relevant errors as defined under the Error Section of this document.
error.errorInfo	Description of the error that can be displayed to the end-user. It should have multi-lingual support.

The entire data object is sent in a JWT format. So, the data object will look like,

```
"data" : "base64urlencode(header).base64urlencode(payload).
base64urlencode(signature)
```

The payload is defined as the entire byte array of the data block.

Error Codes

Code	Message
0	Success
101	Unable to detect a biometric object
102	Technical error during extraction.
103	Device tamper detected
104	Unable to connect to the management server
105	Image orientation error
106	Device not found
107	Device public key expired
108	Domain public key missing
109	Requested number of biometric (Finger/IRIS) not supported
5xx	Custom errors. The device provider is free to choose his error code and error messages.