Unique Identification Authority of India (UIDAI),

Government of India (GoI), Bangla Sahib Road, Behind Kali Mandir, Gole Market

New Delhi - 110001

AADHAAR TOKENIZE API

TECHNICAL SPECIFICATION - VERSION 1.0

January 2019

Contents

1 Introduction

The Unique Identification Authority of India (UIDAI) has been established with the mandate of providing a Unique Identification Number (Aadhaar) to all residents of India. The UIDAI also provides the service of online authentication of identity on the basis of demographic and biometric data.

1.1 UID Tokenization

UIDAI has introduced tokenization within Aadhaar authentication system. UID Token is returned as part of every authentication which is the unique token for that Aadhaar number holder within that agency. Agencies should use UID token to seed their database and map to their customer/beneficiary data. This Token will be unique for each Aadhaar number for a particular entity (AUA/Sub-AUA). This Token will remain same for an Aadhaar number for all authentication requests by that particular entity. However, for a particular Aadhaar Number, different AUAs/Sub-AUAs will have different UID Tokens. The UID Token is an alphanumeric string meant only for system usage. More about VID and UID Token are available on UIDAI website.

This document describes the API that can be used to migrate Aadhaar numbers within various systems to corresponding tokens. AUAs and KUAs can invoke this API as part of migration.

1.2 Target Audience and Pre-Requisites

This is a technical document that is targeted at software professionals who are incorporating the Aadhaar Tokenize API into their applications. Readers should also read the following related documents for complete understanding.

1. Aadhaar Authentication API - http://uidai.gov.in/images/resource/aadhaar_authentication_api_2_5.pdf

2 Tokenize API

This chapter describes the specification in detail for Tokenize API which can be used to migrate Aadhaar numbers to corresponding tokens within databases of various agencies.

2.1 API Details

Tokenize API allows various agencies to tokenize Aadhaar number within their databases and convert existing eKYC XMLs to API 2.5 format.

SECURITY NOTE: This API MUST ONLY be implemented for backend server to server usages and agencies MUST NOT expose this API to Internet or to user interfaces (internal or external).

2.2 API Protocol

Aadhaar Tokenize API is exposed as stateless service over HTTPS. Usage of open data format in XML and widely used protocol such as HTTP allows easy adoption and deployment of Aadhaar Tokenize API.

Following is the URL format for Aadhaar Tokenize service:

https:// *<host>*/uidauthtokenizationapi/<ver>/ *<ac>*/<asalk>

API input data should be sent to this URL as XML document using Content-Type “application/xml” or “text/xml”.

2.2.1 Element Details

host – Aadhaar Tokenize server address. Actual production server address will be provided to ASAs. Note that production servers can only be accessed through private secure connection. ASA server should ensure that actual URL is configurable.

ver – Tokenize API version. For this specification, version value is “**1.0**”.

ac – Unique agency code assigned by UIDAI.

asalk – A valid ASA license key. It is important that license keys are maintained safely. **When adding license key to the URL, ensure it is “URL encoded” to handle special characters**.

For all valid responses, HTTP response code 200 is used. All application error codes are encapsulated in response XML element. In the case of connection and other server errors, standard HTTP error response codes are used (4xx codes such as 403, 404, etc.). HTTP automatic redirects also should be handled by ASA server.

2.3 Tokenize API: Input Data Format

Aadhaar Tokenize API uses XML as the data format for input and output. To avoid sending unnecessary data, do not pass any optional attribute or element unless its value is different from default value. Any bad data or extra data will be rejected.

Following is the XML data format:

```
<Tokenize ac="" sa=”” ver="" txn="" lk="" ts="" eKycFlag="" de=””>
   <Skey ci="">encrypted and encoded session key</Skey>
   <Hmac>SHA-256 Hash of data block, encrypted and then encoded</Hmac>
   <Data>encrypted TokenizeData element</Data>
   <Signature>Digital signature of AUA</Signature>
</Tokenize>
```

Data element contains encrypted TokenizeData element which is given below:

```
<TokenizeData id="">
      Base-64 encoded eKYC XML if eKycFlag is set to true
</TokenizeData>
```

2.3.1 Element Details

Element: **Tokenize** (mandatory)

- Root element of the input XML for Tokenize API.

Attributes:

- **ac** – (mandatory) A unique code for the AUA which is assigned by UIDAI during AUA registration process. This is an alpha-numeric string having maximum length 10.
- **sa** – (mandatory) A unique “Sub-AUA” code assigned by UIDAI. AUAs should get their Sub-AUAs registered within UIDAI system.

  - This allows auditing and business intelligence to be provided at SA level. If AUA and SA are same agency, use value of “ac” for this attribute.
  - This is an alpha-numeric string having maximum length 10.
  - AUA to intimate UIDAI of their valid sub-AUAs and the sub-AUA code will be assigned by UIDAI.
- **ver** – (mandatory) version of the API. Currently only valid value is “1.0”.
- **txn** – (mandatory) Agency specific transaction identifier. Calling agency can choose to pass this as part of input. This is returned as part of response as is. This is very useful for linking transactions full round trip across systems.
  - This is an alpha-numeric string of maximum length 50. Only supported characters are A-Z, a-z, 0-9, period, comma, hyphen, backward & forward slash, left & right parenthesis, and colon. No other characters are supported.
  - This **MUST NOT** start with **“U*:”** where “*” can be one or more alpha-numeric characters. All namespaces starting with “U” is reserved for various APIs offered by UIDAI.
- **lk** – (mandatory) A valid “License Key” assigned to the agency.
  - These license keys have expiry built into them and agencies need to generate new license keys before current ones expires through self-service portal.
  - This is an alpha-numeric string of length up to 64 characters.
- **ts –** (mandatory) Timestamp at the time of capture of Tokenize input. This is in format “YYYY-MM-DDThh:mm:ss” (derived from ISO 8601).

- **eKycFlag –** (mandatory) This field contains either “Y” or “N”. Pass this ONLY if eKYC XML is also need to be tokenized using this API.
  - If flag is “Y” then eKYC XML MUST BE passed in base-64 encoded form as the value of TokenizeData element.
  - NOTE: if eKYC XML is passed, then Aadhaar number within the eKYC XML must be same as what is passed in “id” field.

- **de** – (optional) To be set to “Y” only if eKycFlag is set to “Y”. Indicates if KUA is delegating decryption to ASA. If this flag is set to “Y”, then ASA public key will be used to encrypt Tokenise API response XML instead of KUA key provided ASA is allowed to do so. This facility is subject to UIDAI approval.
  - **This is OPTIONAL attribute and hence should be used ONLY when KUA requires to change the default option based on ASA setup. This option works only if ASA is approved to do decryption**.
  - By default, KUA public key is always used to encrypt e-KYC response.
  - If KUA key is NOT available in CIDR, ASA key will be used to encrypt provided ASA is authorized to do so.
  - A dynamic option of setting “de” attribute to “Y” allows KUA to make this choice at transaction level based on the ASA they use for Tokenise API service.

Element: **Skey** (mandatory)

- Value of this element is base-64 encoded value of encrypted 256-bit AES session key. **Session key must be dynamically generated for every transaction (session key must not be reused) and must not be stored anywhere except in memory**.

Attributes:

- **ci –** (mandatory) Public key certificate identifier using which "skey" was encrypted. UIDAI may have multiple public keys in field at the same time. Value of this attribute is the certificate expiration date in the format "YYYYMMDD". Expiry date of the certificate can be obtained from the certificate itself.

Element: **Data** (mandatory)

- Contains the encrypted "TokenizeData" element in base-64 format.

Element: **Hmac** (mandatory)

- AUA server should construct Hmac of the "TokenizeData" element as follows:
  - After forming TokenizeData XML, compute SHA-256 hash of XML string
  - Then encrypt using session key and then encode using base-64 encoding

Element: **Signature** (mandatory)

- The request XML should be digitally signed for message integrity and non-repudiation purposes.
- Digital signing should always be performed by the entity that creates the request XML
- See Authentication API for details of API input signing.

Element: **TokenizeData** (mandatory)

- Element containing actual input data for Tokenize API.
- For procedure to be adopted for encryption, please refer the latest authentication API.

Attributes:

- **id –** (mandatory) This field contains the Aadhaar number that needs to be tokenized.

2.4 Tokenize API: Response Data Format

Response XML is as follows:

```
<TokenizeRes ret="y|n" code="" txn="" err="" ts="" eKycFlag="”>
   <Skey ci="">encrypted and encoded session key</Skey>
   <Hmac>SHA-256 Hash of data block, encrypted and then encoded</Hmac>
   <Data>encrypted TokenData element</Data>
   <Signature>Digital signature of UIDAI</Signature>
</TokenizeRes>
```

Data element contains encrypted (if it contain eKYC, ie eKycFlag is “Y”), TokenData element which is given below:

```
<TokenData id="">
      Base-64 encoded eKYC 2.5 XML which was part of input, after
tokenization
</TokenData>
```

2.4.1 Element Details

Element: **TokenizeRes**

Attributes:

- **ret** – this is the main Tokenize response. It is either “y” or “n”.
- **code** – unique alphanumeric “Tokenize response” code having maximum length 40. If the input is “not” processed due to errors such as decryption, wrong hmac value, etc., a value of “NA” will be returned. But, due to some transmission errors or changes in deployments, if code is returned as “NA”, AUAs may retry the transaction and if it continues to fail, may report to UIDAI technical support.
- **txn** – Authenticator specific transaction identifier. This is exactly the same value that is sent within the request.
- **ts** – Timestamp when the response is generated. This is of type XSD dateTime.

- **eKycFlag** – (mandatory) This field contains either “Y” or “N”. This will be set to “Y” if input contained an eKYC XML that has been tokenized.
    - If flag is “Y” then tokenized eKYC 2.5 XML is returned in base-64 encoded form as the value of TokenizeData element which is encrypted. In this case Skey and Dat elements also will be present in the response.
    - If this flag is “N”, then no eKYC data or skey/hmac will be available. In this case TokenData element will only contain unencrypted “id” attribute
    - Response eKYC XML will be of the same format as in eKYC API version 2.5.

- **err –** Failure error code. If tokenisation fails ("ret" attribute value is "n"), this attribute provides any of the following codes:

    - **"T100"** – Invalid value (Aadhaar number) in 'id' attribute
    - **"T200" -** The aadhaar number in 'id' (Aadhaar number) and inside eKYC XML must be the same.
    - **"T500" –** Invalid encryption of session key.
    - **"T501"** – Invalid certificate identifier in "ci" attribute of "Skey".
    - **"T502"** – Invalid encryption of TokenizeData.
    - **"T503"** – Invalid encryption of Hmac.
    - **‶T510"** – Invalid Tokenize API XML format.
    - **"T542"** – AUA not authorized for ASA. This error will be returned if AUA and ASA do not have linking in the portal.
    - **"T561"** – Expired Request (value specified in Tokenize element "ts" is older than CIDR time beyond acceptable threshold).
    - **"T562"** – Timestamp value is future time (value specified in "ts" is ahead of CIDR time beyond acceptable threshold).
    - **"T563"** – Duplicate request (this error occurs when exactly same Tokenize request was re-sent by AUA).
    - **"T564"** – HMAC Validation failed.
    - **"T565"** – AUA license has expired.
    - **"T566"** – Invalid non-decryptable license key.
    - **"T569"** – Digital signature verification failed (means that Tokenize request XML was modified after it was signed).
    - **"T570"** – Invalid key info in digital signature (this means that certificate used for signing the tokenization request is not valid – it is either expired or does not belong to the AUA or is not created by a well-known Certification Authority).
    - **"T930 to T939"** – Technical error that are internal to CIDR.
    - **"T940"** – Unauthorized ASA channel.
    - **"T941"** – Unspecified ASA channel.
    - **"T999"** – Unknown error.

Element: **Skey** (optional)

- If there is some eKYC data to return (ie if eKycFlag is “Y”), then this element will be present in output (i.e. if only token need to be returned, this element will be missing).
- Value of this element if present, is base-64 encoded value of encrypted (**using KUA public key**) 256-bit AES session key.

Attributes:

- **ci** – (optional) Public key certificate identifier using which “skey” was encrypted. Value of this attribute is the agency certificate expiration date in the format “YYYYMMDD”. If not present, caller should assume that latest public key is used.

Element: **Data** (optional)

- If there is some data to return, then this element will be present in output.
- Contains the “TokenizeData” element either encrypted or unencrypted depending on presence of eKYC data in base-64 format.

Element: **Hmac** (optional)

- If there is some ekyc data to return (ie if eKycFlag is “Y”),, then this element will be present in output.
- UIDAI server constructs Hmac of the “TokenizeData” element as follows:
  - After forming TokenizeData XML, compute SHA-256 hash of XML string
  - Then encrypt using session key and then encode using base-64 encoding

Element: **Signature** (mandatory)

- The response XML is digitally signed by UIDAI server.

Element: **TokenData** (optional)

- Element containing actual output data for Token API. Have base64 encoded encrypted ekyc data along with “id”, if eKycFlag is set to “Y”. Else contain only unencrypted “id” attribute.

Attributes:

id – (mandatory) This field contains the UID token corresponding to the Aadhaar number in the input.

2.5 Note

This API is for converting the existing Aadhaar numbers to UID token and for converting eKYC 2.1 and earlier responses available with KUAs to eKYC 2.5 format. Mobile Number, email ID, Sub District and PO names will no longer be present in response as part of this API. Other than these parameters, there will not be any change in the data sent as part of previous response (which is used as input to this API), including the timestamp of earlier response. Hence the response as part of this API will NOT be a fresh eKYC of the resident. If a fresh eKYC is required for the KUA, it is requested to do a new transaction with resident presence and informed consent. AUAs/KUAs/ASAs are required to use this API wisely to convert existing aadhaar numbers and eKYCs to UID token and eKYC 2.5 formats, only if they are allowed to do so by any prevailing law.