Unique Identification Authority of India (UIDAI)
Government of India (GoI)
Bangla Sahib Road, Behind Kali Mandir, Gole Market
New Delhi 110001

AADHAAR AUTHENTICATION

API SPECIFICATION - VERSION 2.5 (REVISION-1)

JANUARY-2022

Table of Contents

1. Introduction

The Unique Identification Authority of India (UIDAI) has been created, with the mandate of providing a Unique Identity (Aadhaar) to all Indian residents. The UIDAI provides online authentication to verify the identity claim of the Aadhaar number holder.

Aadhaar "*authentication*" means the process wherein Aadhaar Number or Virtual ID or UID Token, along with other attributes, including biometrics, are submitted to the Central Identities Data Repository (CIDR) for its verification on the basis of information or data or documents available with it. UIDAI provides an online service to support this process. Aadhaar authentication service only responds with a "yes/no" and no personal identity information is returned as part of the response.

1.1 Target Audience and Pre-Requisites

This is a technical document and is targeted at software professionals working in technology domain and interested in incorporating Aadhaar authentication into their applications.

Readers should also read the following API related documents for complete understanding.

1. Aadhaar OTP Request API - https://uidai.gov.in/ecosystem/authentication-devices-documents/authentication-documents.html

2. Aadhaar Registered Devices Specification - https://uidai.gov.in/ecosystem/authentication-devices-documents/authentication-documents.html

3. Aadhaar e-KYC API - https://uidai.gov.in/ecosystem/authentication-devices-documents/authentication-documents.html

1.2 Terminology

Authentication User Agency (AUA) and Sub-AUA: An organization or an entity using Aadhaar authentication as part of its applications to provide services to Aadhaar number holders. Examples include Government Departments, Banks, and other public or private organizations. All AUAs(Authentication User Agencies) must be registered within Aadhaar authentication server to perform secure authentication. Sub-AUA is an entity having a business relationship with AUA offering specific services in a particular domain.

Authentication Service Agency (ASA): An organization or an entity providing connectivity using private secure network to UIDAI's data centres for transmitting authentication requests from various AUAs.

Authentication Factors: Aadhaar authentication supports authentication using multiple factors. These factors include demographic data, biometric data, PIN, OTP, possession of mobile, or combinations thereof. Adding multiple factors increases the strength of authentication. Applications using Aadhaar authentication need to choose appropriate authentication factors based on risk level of the transaction. AUAs can add their own factors to strengthen authentication.

1.3 Legal Framework

The Aadhaar (Targeted Delivery of Financial and Other Subsidies, Benefits and Services) Act 2016was published in gazette notification on March 26, 2016. The Act is to provide for, as a good governance, efficient, transparent, and targeted delivery of subsidies, benefits and services to Aadhaar Number holders. A gazette notification was issued by Central Government on 12th July 2016 to establish UIDAI as an Authority and operationalize certain provisions of Aadhaar Act 2016. Authentication regulations are also published under this Act. These documents specify legal framework for authentication usage, AUA/ASA engagements, audits, and other details. Detailed partner documents are also published. These documents are available at https://uidai.gov.in/ecosystem/authentication-devices-documents/authentication-documents.html

1.4 Objective of this document

This document provides Aadhaar Authentication API (Application Programming Interface) specification. It contains details including API data format, protocol, and security specifications.

For latest documents related to Aadhaar authentication, partner guidelines, other APIs, and related documents, see https://uidai.gov.in/ecosystem/authentication-devices-documents/authentication-documents.html

2. Understanding Aadhaar Authentication

This chapter describes Aadhaar authentication, some of the envisioned usage scenarios, and working details. Technical details follow in subsequent chapters.

2.1 Aadhaar Number

The Unique Identification (Aadhaar) Number gives individuals the means to clearly establish their identity to public and private agencies across the country. Three key characteristics of Aadhaar number are:

1. Permanency (Aadhaar number remains same during lifetime of the person)
2. Uniqueness (one Aadhaar number holder has one ID and no two Aadhaar number holders have same ID)
3. Global (same identifier can be used across applications and domains)

Aadhaar number is provided during the initiation process called *enrolment* where his/her demographic and biometric information are collected and uniqueness of the provided data is established through a process called *de-duplication*. Post de-duplication, an Aadhaar number is issued and a letter is sent to Aadhaar number holder informing the details.

2.2 Virtual Identity (VID) Number and UID Token

Virtual ID is a temporary, revocable 16-digit random number mapped with the Aadhaar number. VID can be used in lieu of Aadhaar number whenever authentication or e-KYC services are performed. Residents are currently required to share Aadhaar Number to authenticate their identity to avail various services. With the introduction of Virtual ID, a fungible number mapped to Aadhaar Number, residents have an option to share their Virtual ID at the time of authentication instead of Aadhaar number to further improve privacy.

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. From this document perspective, it is important to note that VID or UID Token can be used in lieu of Aadhaar

number within authentication request. In addition, authentication response contains the UID Token for whom authentication was done so that agencies can store that within their database or audits.

2.3 Aadhaar Authentication at a Glance

Aadhaar authentication is the process wherein Aadhaar Number (wherever Aadhaar number is used, in future, an encrypted Aadhaar number may also be used) or Virtual ID or UID Token, along with other attributes, including biometrics, are submitted online to the CIDR for its verification on the basis of information or data or documents available with it.

In all forms of authentication the Aadhaar Number/Virtual ID/UID Token needs to be submitted so that authentication is reduced to a 1:1 match. In addition, Aadhaar authentication service only responds with a "yes/no" and no Personal Identity Information (PII) is returned as part of the response.

Aadhaar authentication provides several ways in which an Aadhaar number holder can authenticate themselves using the system. At a high level, authentication can be using Demographics data and/or Biometric (FP/Iris/Face) data, and/or OTP.

During the authentication transaction, the Aadhaar number holder's record is first selected using the Aadhaar Number/Virtual ID/UID Token and then the demographic/biometric inputs are matched against the stored data within CIDR which was provided by the Aadhaar number holder during enrolment/update process.

2.4 Aadhaar Authentication Usage

Aadhaar authentication enables agencies to verify identity of Aadhaar number holders using the UIDAI authentication service that is only made available via a private secure network. Aadhaar authentication service provides services to instantly verify the identity of the Aadhaar number holder against the available data in CIDR. Based on the needs of the service, different factors could be used along with Aadhaar Number/Virtual ID/UID Token. These factors could be combination of biometrics (such as fingerprints, iris, face impressions) and/or demographic information (such as Name, Date of birth, Address)and/or a secret PIN (currently not implemented) or OTP (One Time Pin).

For further details on usage of Aadhaar in various service delivery scenarios, refer to "Aadhaar Enabled Service Delivery" White Paper published on UIDAI website.

3. Aadhaar Authentication API

This chapter describes the API in detail including the authentication flow, communication protocol, and data formats.

3.1 Authentication Flow

Following diagram explains various authentication scenarios and data flow.

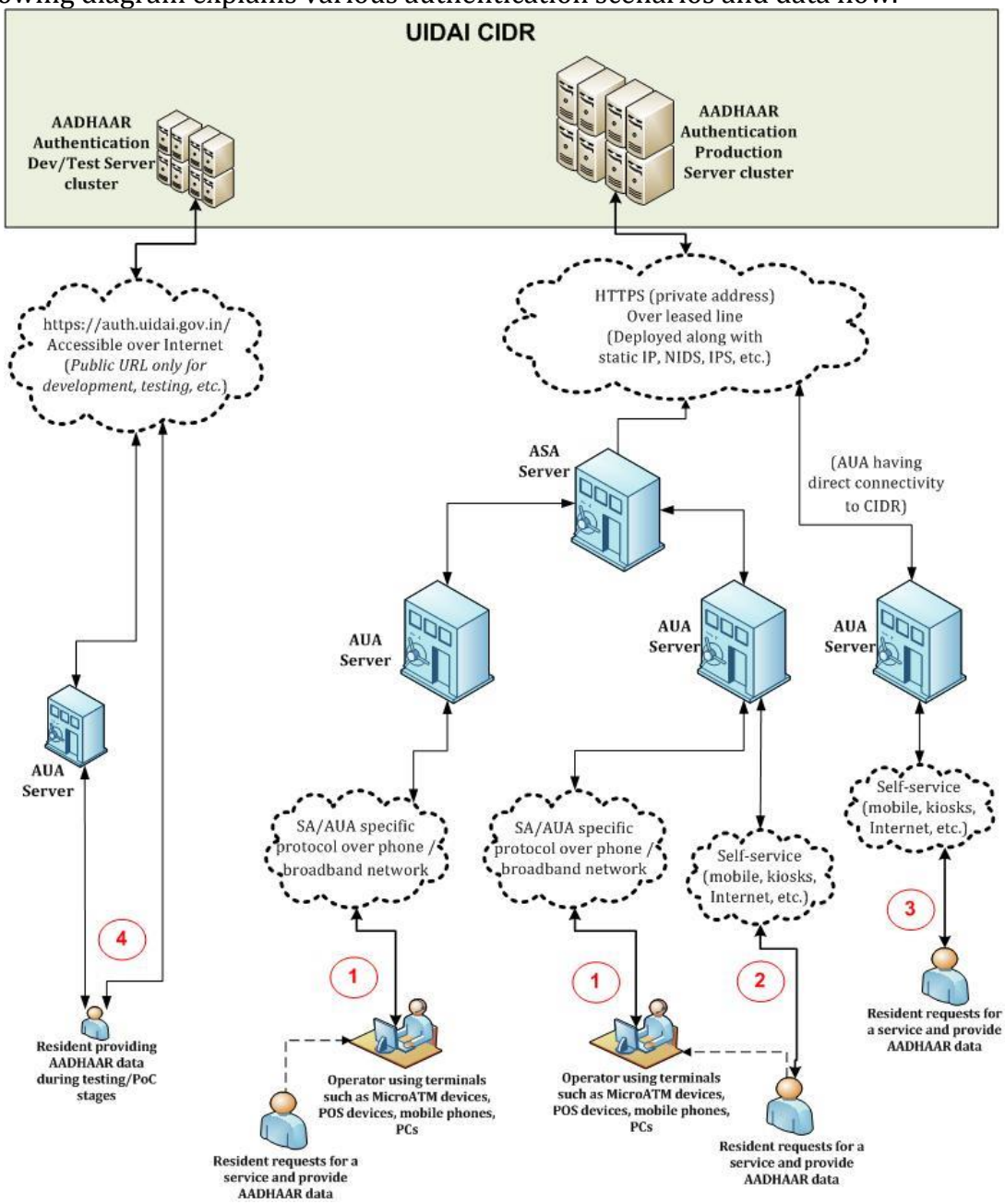

Figure 1: Aadhaar authentication flow under various scenarios

Scenario 1 in the diagram is a typical authentication flow and is a case of an operator assisted transaction:

a) Aadhaar number holder provides Aadhaar Number or Virtual ID or AUA/Sub-AUA specific identifier, necessary demographic and/or OTP and/or biometric details which is captured using registered devices belonging to the AUA/Sub-AUA (or merchant/operator appointed by AUA/Sub-AUA).
b) Aadhaar authentication enabled application software that is installed on the device packages these input parameters, encrypts, and sends it to AUA server over either a mobile/broadband network using a secure protocol such as HTTPS.
c) AUA server, after validation, adds necessary headers (AUA specific wrapper XML with license key, signature, etc.), and passes the request through ASA server to UIDAI CIDR.
d) Aadhaar authentication server returns a "yes/no" based on the match of the input parameters.
e) Based on the response from the Aadhaar authentication server, AUA/Sub-AUA conducts the transaction.

Scenario 2 below depicts the Aadhaar number holder conducting assisted transactions with Aadhaar authentication on his/her mobile or on the Internet.

a) In this case, transaction data is captured on the mobile/Internet application provided by AUA/Sub-AUA.
b) Aadhaar number holder provides necessary demographic data long with OTP (fingerprint/iris is also possible using registered devices) in addition to AUA specific attributes (account number, password, PIN, etc.). Aadhaar authentication enabled application software that is installed on the device packages these input parameters, encrypts, and sends it to AUA server over either a mobile/broadband network using a secure protocol such as HTTPS.
c) Step c, d, and e are same as in scenario 1 above.

Scenario 3 is a slight variant of 2nd scenario where AUA also plays the role of ASA and has direct connectivity to UIDAI data centers. Scenario 4 is how AUAs and application developers can test Aadhaar authentication using the public URL.

3.2 APIProtocol

Aadhaar authentication service is exposed as stateless service over HTTPS. Usage of open data format in XML and widely used protocol such as HTTPS allows easy adoption and deployment of Aadhaar authentication. To support strong end to end security and avoid request tampering and man-in-the-middle attacks, it is essential that encryption of data happens at the time of capture on the capture device.

Following is the URL format for Aadhaar authentication service:

```
https://<host>/<ver>/<ac>/<uid[0]>/<uid[1]>/<asalk>
```

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

For security reason data collected for Aadhaar authentication must not be stored in the devices or log files. It's essential for ASA and AUA to maintain audit records for all the authentication request along with the response in compliance with regulations published by the authority.

As a best practice, for all secure communications the agencies should automatically validate the SSL/TLS certificate and ensure it is validated against the revocation list

3.2.1 Element Details

host – Aadhaar authentication 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. (*For development and testing purposes, public URL "auth.uidai.gov.in" can be used.*)

ver – Authentication API version (mandatory). UIDAI may host multiple versions for supporting gradual migration. For this specification, version is "**2.5**".

ac – A unique code for the AUA which is assigned by UIDAI. This is an alpha-numeric string having maximum length 10.(A default value "public" is available for testing.)

uid[0] and **uid[1]** – First 2 digits of Aadhaar Number. When VID, UID Token, or encrypted Aadhaar number (future) is used, pass "0" and "0" for these.

asalk – A valid ASA license key. ASAs must send one of their valid license keys at the end of the URL. 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.

ASA server must send one of their valid license keys as part of the URL (see details above). Authentication related APIs are enabled only for valid ASAs and only for their registered static IP addresses coming through a secure private network.

3.3 Authentication API: Input Data Format

Aadhaar authentication 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 for authentication API:

```
<Auth uid="" rc="" tid="" ac="" sa="" ver="" txn="" lk="">
<Uses pi="" pa="" pfa="" bio="" bt="" pin="" otp=""/>
<Device rdsId="" rdsVer="" dpId=""dc="" mi=""mc=""/>
<Skey ci="">encrypted and encoded session key</Skey>
<Hmac>SHA-256 Hash of Pid block, encrypted and then encoded</Hmac>
<Data type="X|P">encrypted PID block</Data>
<Signature>Digital signature of AUA</Signature>
</Auth>
```

“Data” element contains “Pid” (Personal Identity Data) element which is a base-64 encoded encrypted block. Complete “Data” block should be encrypted at the time of capture on the client. But, encoding (base-64) of “Data” block and packaging it with enveloping XML under “Auth” element can either be done on the device or on the AUA server based on the AUA needs. Device capability, protocol between devices and AUA server, and data format used between devices and AUA server, etc. should be considered for making that choice.

When using PID block in XML format (which is the default), following is the format for “Pid” element:

```
<Pid ts="" ver=""wadh=””>
<Demo lang="">
<Pi ms="E" mv="" name="" lname="" lmv="" gender="M|F|T" dob="" dobt="V|D|A"
age="" phone="" email=""/>
<Pa ms="E" co="" house="" street="" lm="" loc=""
        vtc="" subdist="" dist="" state="" country="" pc="" po=""/>
<Pfa ms="E" mv="" av="" lav="" lmv=""/>
</Demo>

<Biosdih="">
<Bio type="FMR|FIR|IIR|FID" posh="" bs="">encoded biometric</Bio>

</Bios>

<Pv otp="" pin=""/>
</Pid>
```

Instead of XML format, this version also allows PID block to be in binary format based on Protocol Buffers standard (http://code.google.com/p/protobuf/). Notice that “Auth” XML must be in XML format. Binary format is only supported for PID block to enable smaller packet sizes to be transmitted from devices. See Appendix A for details.

3.3.1 Element Details

Element:**Auth** (mandatory)

- Root element of the input XML for authentication service.

Attributes:

- **uid** – (mandatory) This can be Aadhaar Number(wherever Aadhaar number is used, in future, an encrypted Aadhaar number may also be used) or Virtual ID or agency specific UID token of the person being authenticated.
- **rc** – (mandatory) Aadhaar number holder consent to do the Aadhaar based authentication using OTP or Biometrics. Only allowed value is "Y". Without explicit informed consent of the Aadhaar number holder AUA/Sub-AUA application should not call this API.
- **tid** – (mandatory) Value should be passed as "**registered**". When not using biometric authentication, value of this attribute must be "".
- **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.(A Default value "public" is available which is ONLY for testing.)
- **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 sub-AUAs and the sub-AUA code will be assigned by UIDAI.
- **ver** – (mandatory) version of the API. Currently only valid value is "2.5".
- **txn** – (mandatory) AUA specific transaction identifier. AUA 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, hyphen, backward & forward slash, left & right parenthesis, and colon. No other characters are supported. Note that the character comma is no longer supported in txn attribute.
    - It is highly recommended that AUAs use unique transaction ID for each request and use this attribute for correlating requests with responses for auditing and verification. All the cases where a response is received from UIDAI, response code attribute should be considered for any issue resolution.
    - **In case of OTP Authentication using Request OTP API call this value of txn MUST be same as the txn value used for Request OTP API call**. This is to ensure OTP cannot be intercepted and used by other applications.
    - 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 AUA/Sub-AUA.

- These license keys have expiry built into them and AUA/Sub-AUA administrator need to ensure that they generate new license keys before current ones expires through self-service portal.
- This is an alpha-numeric string of length up to 64 characters.

Element: **Uses** (mandatory)

- This element specifies the authentication factors used by the request. When an authentication factor is specified in this element, that specific attribute must be present in the encrypted data block. This is particularly useful in situations where the AUA does not fully control the terminal device, but wishes to maintain a certain level of control on the authentication transaction.

Attributes:

- **pi** – (mandatory) Valid values are "y" or "n". If the value is "y" then at least one attribute of element "Pi" (part of "Demo" element) should be used in authentication. If value is "n", "Pi" element is not mandated.
- **pa** – (mandatory) Valid values are "y" or "n". If the value is "y" then at least one attribute of element "Pa" (part of "Demo" element) should be used in authentication. If value is "n", "Pa" element is not mandated.
- **pfa** – (mandatory) Valid values are "y" or "n". If the value is "y" then element "Pfa" (part of "Demo" element) should be used in authentication. If value is "n", "Pfa" element is not mandated.
- **bio** – (mandatory) Valid values are "y" or "n". If the value is "y" then at least one biometric element "Bio" (part of "Bios" element) should be used in authentication. If value is "n", "Bio" element is not mandated.
- **bt** – (mandatory only if "bio" attribute has value "y") provide a comma separated list of biometrics used. Valid values that can be used in this comma separated list are "FMR", "FIR", "IIR",and"FID". In case if both FMR and FIR is used for single finger "bt" will have"FMR,FIR". If "FMR" is part of the list, then at least one "Bio" element with type FMR should be used. Similarly, if "FIR" or "IIR" or "FID" are part of the list, then at least one "Bio" element with those types must be used. Multiple bio elements may be used as per the API specifications.
- **pin** – (mandatory) Valid values are "y" or "n". If the value is "y" then PIN should be used in authentication. Otherwise, "pin" is not mandated.
- **otp** – (mandatory) Valid values are "y" or "n". If the value is "y" then OTP should be used in authentication. Otherwise, "otp" is not mandated.

Element: **Device**(Mandatory)

This element specifies information related to the device. Except for udc, all other attributes are valid only for biometric authentication.

Attributes:

- **dpId** – (mandatory for bio auth) Unique code assigned to registered device provider. Returned by RD Service when using biometric authentication.
- **rdsId** – (mandatory for bio auth) Unique ID of the certified registered device service. Returned by RD Service when using biometric authentication.

- **rdsVer** – (mandatory for bio auth) Registered devices service version. Returned by RD Service when using biometric authentication.
- **dc** – (mandatory for bio auth) Unique Registered Device Code. Returned by RD Service when using biometric authentication.
- **mi** – (mandatory for bio auth) Registered device model ID. Returned by RD Service when using biometric authentication.
- **mc** – (mandatory for bio auth) This attribute holds registered device public key certificate. This is signed with device provider key. Returned by RD Service when using biometric authentication.

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**. See next chapter for encryption details.

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 "Pid" element in base-64 format. See "Pid" element definition later.

Attributes:

- **type**– (optional) Type of the PID block format. It can have two values – "X" for XML and "P" for Protobuf binary format. Default value is assumed to be "X".

Element: **Hmac** (mandatory)

- Devices which is constructing the "Pid" element must perform the following to provide the Hmac value:
    - If Pid type is "X" (XML), then:
        - After forming Pid XML, compute SHA-256 hash of Pid XML string
        - Then encrypt using session key
        - Then encode using base-64 encoding (as described earlier, encoding can be done on the AUA server when forming final Auth XML)
    - If Pid type is "P" (Protobuf), then:
        - After forming Protobuf byte array for Pid, compute SHA-256 hash of Pid protobuf bytes.
        - Then encrypt using session key
        - Then encode using base-64 encoding (as described earlier, encoding can be done on the AUA server when forming final Auth XML)

Authentication server performs the following processing on each authentication request:

1. Decode and Decrypt the Pid from Data element. Based on type attribute of the "Data" element, the value of Data is either interpreted as XML or Protobuf bytes.
2. Re-compute the SHA-256 Hash of Pid.
3. Decode and decrypt the value of Hmac element.
4. Compare the re-computed SHA-256 hash with Hmac value received in authentication request.
    a. If both values match, then, integrity of authentication request is preserved and server will proceed with further processing of the request.
    b. If values do not match, reject the authentication request with error code representing "HMAC Validation failed".

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 final request XML
    - AUA can digitally sign after forming the API input XML. This is almost always the case. In such cases, AUA ensures the message security and integrity between AUA servers and its client applications.
    - ASA can digitally sign the request XML if it is a domain-specific aggregator and forms the request XML on behalf of the AUA. In such cases, ASA and AUA ensure the message security and integrity between their servers.
- Procuring digital signature certificates:
    - It should be procured from a valid certification authority as per Indian IT Act (seehttp://www.cca.gov.in/cca/?q=licensed_ca.html)
    - Digital certificates have two parts:
        - X.509 certificate representing public key.
        - Private Key which is used for digital signing. Private Key should be stored securely and is the responsibility of the owner of the certificate to ensure that it is not compromised.
    - It should be a class II or class III certificate.
    - X.509 certificate contains information about the owner of the certificate; in this case it will be details of the person and the organization to which he/she belongs. UIDAI server checks to ensure that certificate belongs to the ASA or AUA organization. Hence, it is mandatory that "O" attribute of "Subject" in the X.509 certificate matches the name of the organization.
- Digital signing of request XML
    - XML digital signature algorithm as recommended by W3C.
    - Signature should include key info element that contains X.509 certificate details. This is needed for UIDAI server to validate the signer.
- Verification of digital signature by UIDAI servers. UIDAI server validates the signature in the following sequence:
    - Checks if the signature element is present. If not, it throws an error.

- If signature element is present, then it validates if the certificate is issued by one of the valid certification authority. If not valid, throws error.
- If it is a valid certificate, then it validates whether the "O" attribute in the X.509 certificate's subject matches the AUA organization name. If yes, proceeds with API logic.
- If it does not match AUA organization name, it checks configuration to see if ASA is allowed to sign on behalf of that AUA. If not, throws error.
- If ASA is allowed to sign on behalf of that AUA, it checks whether the "O" element of the certificate matches with the organization name of the ASA. If not, throws error.
- If it matches, it proceeds with API logic.
- In future, UIDAI may choose to conduct additional validations against white listed certificates within UIDAI database.

Element:**Pid** (mandatory)

Attributes:

- **ts** – (mandatory) Timestamp at the time of capture of authentication input. This is in format"YYYY-MM-DDThh:mm:ss" (derived from ISO 8601). Time zone should not be specified and is automatically defaulted to IST (UTC +5.30).**Since timestamp plays a critical role, it is highly recommended that devices are time synchronized with a time server**.

AUAs can buffer authentication requests and send it to Aadhaar authentication server to support occasional lack of network connectivity on the field. Maximum time up to which requests can be queued (buffered) will be defined by UIDAI policy. Currently, this will be configured to 24 hours and may be changed as per policy. All requests with "ts" value older than this limit will be rejected. If resident changes his VID number before the buffer authentication request is received in CIDR, the request will fail with invalid VID number.

- **ver** – (mandatory) version of the "Pid" element. Currently only valid value is "2.0" Notice that this is NOT same as Authentication API version. Pid version 1.0 is only for Authentication API versions 1.x.
- **wadh**– (optional) "Wrapper API data hash". SHOULD BE empty for all regular authentication transactions. ONLY to be used for specific transaction types such as eKYC and Update APIs. See those API documents for detail. This is a hash value passed by those wrapper APIs for PID binding. No other authentication call should pass any value in this attribute and must be left empty.

Element: **Demo** (optional)

- Contains child elements "Pi", "Pa" and "Pfa", all of which are optional.
- All demographic data fields as per KYR specifications.

Attributes:

- **lang** – (optional) "Indian Language Code" in the case of using Indian language data for demographic match (see lname, lav attributes). This must be a valid language code from the following table

Language	Language code
Assamese	01
Bengali	02
Gujarati	05
Hindi	06
Kannada	07
Malayalam	11
Manipuri	12
Marathi	13
Oriya	15
Punjabi	16
Tamil	20
Telugu	21
Urdu	22

NOTE: Indian language matching of name and address allows data to be matched in any of the above languages using a fuzzy matching logic. In the case of address where multiple fields are provided as a single string (using "lav" attribute), it is recommended to separate each field (house, street, locality, vtc, district, etc) by comma.

Element: **Pi** (Optional)

- This element captures attributes related to "Personal Identity"

Attributes:

- **ms** – (optional) "Matching Strategy" for "name" attribute. Valid values are "E" (Exact match) and "P" (Partial match). This is used only for "name" attribute. Defaulted to "E".
- **mv**– (optional) "Match value" for "name" attribute, Only valid value is 100.
- **name**– (optional) Name of the Aadhaar number holder in English. Maximum length is 60.

  NOTE: If "ms" and/or "mv" are provided, but, "name" attribute is not provided or empty value is provided, no name matching will be performed.

  When using matching strategy "Exact" (ms="E"), the name attribute is compared for exact match with the name stored in Aadhaar database. Though comparison is case insensitive, all the words of the name must be specified in the exact same order as provided by the Aadhaar number holder during Aadhaar enrolment.

- **lname** – (optional) Aadhaar number holder's name in Indian language. This is a Unicode String in the language specified by the "lang" attribute of "Demo" element. Notice that this is a phonetic matching against the data stored in CIDR.

NOTE: If this attribute is provided, "lang" attribute must be specified for "Demo" element.

- **lmv** – (optional) Local Language Match Value to adjust phonetic match threshold. Only valid value is 100.
- **gender** – (optional) Valid values are "M" for male, "F" for female, and "T" for transgender.
- **dob**– (optional) Date of Birth in "YYYY-MM-DD" format. If only year needs to be authenticated, then use format "YYYY".
- **dobt** – (optional) Date of Birth Type as indicated in Aadhaar system. This attribute can have only 3 values – "V" (for Verified), "D" (for Declared), and "A" (Approximate). When the Aadhaar number holder is enrolled, DoB may be recorded along with any of these types.
- **age** – (optional)In certain use cases such as checking whether a Aadhaar number holder can be considered a senior citizen or an adult (age above or equal to 18 years), it may be desired that only age of a Aadhaar number holder can be verified using Aadhaar Authentication instead of verifying against complete data of birth. When "age" attribute is specified, authentication will pass if Aadhaar number holder's age is "equal to or greater than" the input age. Else, it will fail with appropriate authentication error code.
- **phone** – (optional) Registered mobile phone number of the Aadhaar number holder.
- **email**– (optional) Registered email address of the Aadhaar number holder. This is case-insensitive match removing trailing and leading spaces.

Element: **Pa** (Optional)

- This element captures attributes related to "Personal Address". These are address fields as provided by the Aadhaar number holder during enrolment or later updates. Only attributes that are sent as part of input will be compared.
- This element should not be used when using "Pfa" element as "Pa" and "Pfa" are mutually exclusive.

Attributes:

All attributes are compared case insensitive after leading and trailing spaces are trimmed and all the occurrences of consecutive spaces are replaced with single space.

- **ms** – (optional) "Matching Strategy" for address attributes. Only the value "E" (Exact match) is supported. This is used only when at least one address attribute is specified.
- **co**– (optional) "Care of" person's name.
- **house** – (optional) House identifier.
- **street**– (optional) Street name.
- **lm** – (optional) Landmark if any.
- **loc**– (optional) Locality if any.
- **vtc** – (optional) Name of village or town or city.
- **subdist** – (optional) Sub-District name.

- **dist** – (optional) District name.
- **state** – (optional) State name.
- **country** – (optional) Country name.
- **pc**– (optional) Postal pin code.
- **po** – (optional) Post Office name.

Element: **Pfa** (Optional)

- This element captures attributes related to "Personal Full Address". It corresponds to the address of the Aadhaar number holder as present in enrolment receipt or Aadhaar letter.
- This element should not be used when using "Pa" element as "Pa" and "Pfa" are mutually exclusive.

Attributes:

- **ms** – (optional) "Matching Strategy" for address attributes. Valid value is "E" (Exact match) .
- **mv**– (optional) Valid value is 100 and it is used only when matching strategy (ms attribute) is "P" (Partial match).
  It represents the percentage of full words from the address stored in Aadhaar database that must be specified in the "av" attribute for the match to be considered successful,
- **av** – (optional) Aadhaar number holder's full address specified as a single string value.

  Normalization:
  "av" value and the Aadhaar number holder's address stored in Aadhaar database, both are normalized using following rules before comparison.

  1. Following characters/phrases are ignored:
     a. Period - (.)
     b. Comma (,)
     c. Hyphen (-)
     d. Asterisk (*)
     e. Opening and closing braces - '(' and ')'
     f. Opening and closing square brackets – '[' and ']'
     g. Apostrophes – `
     h. Single quotes – '
     i. Double quotes – "
     j. Forward slash - /
     k. Backward slash - \
     l. Hash - #
     m. Care of labels – C/O, S/O, D/O, W/O, H/O
     n. Other labels - "No."
  2. Leading and trailing spaces are trimmed and all the occurrences of multiple consecutive spaces are replaced with single space.

When using matching strategy "Exact" (ms="E"), the normalized "av" attribute is compared for exact match with the normalized Aadhaar number holder's address stored in Aadhaar database.

- **lav** – (optional) Aadhaar number holder's Address in Indian language. This is similar to "av" attribute described above except that this is represented in an Indian language. This will be matched against address data available in the CIDR database. If this attribute is provided, "lang" attribute must be specified for "Demo" element.
- **lmv** – (optional) Local Language Match Value to adjust phonetic match threshold. Valid value is 100.

Element: **Bios** – (optional)
This element can have one or many "Bio" elements carrying biometric records to be matched.
Note:- In case both FIR and FMR is sent for single finger, two "Bio" elements are allowed. In case of dual finger authentication with both FIR and FMR, upto four such elements are allowed.Only in the case of BFD, more than four and up to ten number of bio elements are allowed. Only FMR should be used in case of BFD transaction.

Attributes:
- **dih** – (mandatory) "Device Info Hash". This will be calculated by Registered Device (RD) service as part of PID block capture when using biometrics.

Element: **Bio** (optional)
- If XML data format is used for PID block, this element contains single base 64 encoded biometric record. This is typically in plain ISO data format.
- In the case of registered devices, this contains encrypted FMR/FIR/IIR/FID record (see "Using Registered Devices" section later in the document).

Attributes:
- **type** – (mandatory) This attribute specifies type of the biometric. Valid values are "FMR" (Finger Minutiae), "FIR" (Finger Image), and "IIR" (Iris Image).
    - ***FMR*** - The biometric data is of type "Fingerprint Minutiae Record". This data is in ISO minutiae format with no proprietary extensions allowed.
    - ***FIR*** - The biometric data is of type "Fingerprint Image Record". The data is a fingerprint image packaged in ISO 19794-4 format, which could contain a lossy compressed image of type Jpeg2000.
    - ***IIR*** - The biometric data is of type "Iris Image Record". The data is an iris image packaged in ISO 19794-6 format, which could contain a lossy compressed image having type Jpeg2000.
    - ***FID*** - The biometric data is of type "Face Image Data". The data is face image packaged in ISO 19794-5 format, which could contain a lossy compressed image having type Jpeg2000. Only one FID need to be there along with another modality. Usage of face alone as biometric modality will result in error "914".
- **bs** – (mandatory for registered device) For registered device, Base-64 encoded signed biometric hash of the bio record. AUA application should call the

registered device capture function to obtain the bio record as well as the signature string (bs).

- **posh**– (mandatory) In general, it is highly recommended that applications pass "UNKNOWN" unless it clearly knows which finger was used. Valid values are:

LEFT_IRIS
RIGHT_IRIS
LEFT_INDEX
LEFT_LITTLE
LEFT_MIDDLE
LEFT_RING
LEFT_THUMB
RIGHT_INDEX
RIGHT_LITTLE
RIGHT_MIDDLE
RIGHT_RING
RIGHT_THUMB
FACE
UNKNOWN

Element: **Pv** (optional)

- This element ("Pin Value") allows support for additional factors "pin" and "otp".

Attributes:

- **pin** – (optional) Actual value of PIN as set by Aadhaar number holder. This is an alpha numeric value. **This option is NOT available for AUAs and is restricted to internal UIDAI usage only**.
- **otp**– (optional) This One Time Pin (OTP) value should be obtained from the Aadhaar number holder. To obtain OTP, following two ways can be used:
    - By the Aadhaar number holder her/himself using UIDAI portal, by sending an inbound SMS from registered phone, by calling IVR from registered phone, or by using UIDAI provided mobile app running on registered phone (TOTP). Some of these options may not be enabled at this time.
    - By programmatically initiating the request from the AUA/sub-AUA application in which authentication is used. Whichever application needing to validate OTP can thus initiate OTP request on behalf of the Aadhaar number holder via Request OTP API.

3.4 Authentication API: Response Data Format

Authentication API does not provide any identity data as part of the response. All it does is to match given input and respond with a "yes/no". Response XML is as follows:

```
<AuthRes ret="y|n"code=""txn=""err="" ts="" actn=""info="">

<BfdRanks>
      <!—following element repeats for all matchable fingers ONLY when the
      request is for BFD (see Best Finger Detection chapter) -->
<BfdRank pos="" value=""/>
</BfdRanks>
```

```
<Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
<SignedInfo>
<CanonicalizationMethod
        Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315" />
<SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha256"
/>
<Reference URI="">
<Transforms>
<Transform
            Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
signature" />
</Transforms>
<DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha256"/>
<DigestValue></DigestValue>
</Reference>
</SignedInfo>
<SignatureValue></SignatureValue>
</Signature>
</AuthRes>
```

Agencies that use the authentication response need a mechanism to validate the authenticity of the authentication response for non-repudiation purposes. In order to enable verification and audit, authentication response will be digitally signed by UIDAI and the signature will be part of the response. AUAs are expected to preserve the entire response XML for non-repudiation purposes.

3.4.1 Element Details

Element: **AuthRes**

Attributes:

- **ret** – this is the main authentication response. It is either "y" or "n".
- **code** – unique alphanumeric "authentication 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 date Time.
- **actn** –This attribute may or may not exist in response. This attribute, alpha-numeric of max length 5, provides specific action codes (published from time to time) **meant to be shown to Aadhaar number holder or operator**.
    - One possible use is to provide feedback to Aadhaar number holder for improving authentication outcomes and required data update notifications.
    - **This attribute MUST be sent to front-end application by ASA/KSA and AUA/KUA to ensure action and corresponding message is displayed to Aadhaar number holder or operator**.
    - For specific ACTION CODES (ACTN) corresponding to the authentication API errors, please refer authentication document link

https://uidai.gov.in/ecosystem/authentication-devices-documents/authentication-documents.html

- **info**–This attribute provides information on the details included in auth. This is composed of the following parts:

Version 4.0structure of "info" (latest):

<Version>{UID token, UID Type, SHA-256 of Demo element, Encoded Usage Data, pid_version, timestamp, fmrcount, fircount, iircount, fidcount, auth_api_ver, SHA-256 of ASA code, SHA-256 of AUA code, SHA-256 of SUB AUA code, lang, pi-ms, pi-mv, pi-lmv, pa-ms, pa-mv, pa-lmv, pfa-ms, pfa-mv, pfa-lmv, tid, rdsId, rdsVer, dpId, mi, rdLevel, wadh}

- "Version" – is the version of the info structure which is "04"
- "UID token" – is the Agency specific and unique Aadhaar token generated for the resident. If this field may have value "NA" if there is a system error (such as signature/LK errors, internal errors, etc.)
- "UID Type" – either "A" or "V" or "T" or "E" indicating if Aadhaar number or Virtual ID or UID Token or Encrypted Aadhaar Number (future) was used in the input during authentication.
- "SHA-256 of Demo Element" –
  - When Pid block is of type "X" (XML), then, it is the SHA-256 hash value of the XML string representing Demo element.
  - When Pid block is of type "P" (Protobuf), then, it is the SHA-256 hash of the Protobuf value (byte array) of the Demo element.
- "Encoded Usage Data" - is 48-bit representation in HEX format of various attribute usage of this authentication request. Hexadecimal digits within the "Encoded Usage Data" should be interpreted based on below rules:

1st hexadecimal digit:
Bit 3-0: Version number of encoding. It will be hexadecimal "1" (binary: 0001) for encoding specified in this document.

2nd hexadecimal digit:
Bit 3: Was "`Pi->name`" attribute used?
Bit 2: Was "`Pi->lname`" attribute used?
Bit 1: Was "`Pi->gender`" attribute used?
Bit 0: Was "`Pi->dob`" attribute used?

3rd hexadecimal digit:
Bit 3: Was "`Pi->phone`" attribute used?
Bit 2: Was "`Pi->email`" attribute used?
Bit 1: Was "`Pi->age`" attribute used?
Bit 0: Was "`Pa->co`" attribute used?

4th hexadecimal digit:
Bit 3: Was "`Pa->house`" attribute used?
Bit 2: Was "`Pa->street`" attribute used?
Bit 1: Was "`Pa->lm`" attribute used?

Bit 0: Was "`Pa->loc`" attribute used?

5th hexadecimal digit:
Bit 3: Was "`Pa->vtc`" attribute used?
Bit 2: Was "`Pa->dist`" attribute used?
Bit 1: Was "`Pa->state`" attribute used?
Bit 0: Was "`Pa->pc`" attribute used?

6th hexadecimal digit:
Bit 3: Was "`Pfa->av`" attribute used?
Bit 2: Was "`Pfa->lav`" attribute used?
Bit 1: Was "`FMR`" used for biometric auth?
Bit 0: Was "`FIR`" used for biometric auth?

7th hexadecimal digit:
Bit 3: Was "`IIR`" used for biometric auth?
Bit 2: Was "`FID`" used for biometric auth?
Bit 1: Was "`Pv->pin`" attribute used?
Bit 0: Was "`Pv->otp`" attribute used?

8th hexadecimal digit:
Bit 3: Was "`Pa->po`"attribute used?
Bit 2: Was "`Pa->subdist`"attribute used?
Bit 1: Was "`Pi->dobt`"attribute used?
Bit 0: Was "SSK" used?

9th hexadecimal digit::
Bit 3: Was "Pi->name" attribute matched?
Bit 2: Was "Pi->lname" attribute matched?
Bit 1: Was "Pi->gender" attribute matched?
Bit 0: Was "Pi->dob" attribute matched?

10th hexadecimal digit::
Bit 3: Was "Pi->phone" attribute matched?
Bit 2: Was "Pi->email" attribute matched?
Bit 1: Was "Pi->age" attribute matched?
Bit 0: Was "Pa->co" attribute matched?

11th hexadecimal digit::
Bit 3: Was "Pa->house" attribute matched?
Bit 2: Was "Pa->street" attribute matched?
Bit 1: Was "Pa->lm" attribute matched?
Bit 0: Was "Pa->loc" attribute matched?

12th hexadecimal digit::
Bit 3: Was "Pa->vtc" attribute matched?
Bit 2: Was "Pa->dist" attribute matched?
Bit 1: Was "Pa->state" attribute matched?

Bit 0: Was “Pa->pc” attribute matched?

13th hexadecimal digit::
Bit 3: Was “Pfa->av” attribute matched?
Bit 2: Was “Pfa->lav” attribute matched?
Bit 1: Was “FMR/FIR” matched for biometric auth?
Bit 0: Was “IIR” matched for biometric auth?

14th hexadecimal digit:
Bit 3: Was “Pa->po” matched for biometric auth?
Bit 2: Was “Pa->subdist” attribute matched?
Bit 1: Was “Pi->dobt” attribute matched?
Bit 0: Was “Registered” device used?

15th hexadecimal digit:
Bit 3: Currently unused. Will have value 0
Bit 2: Currently unused. Will have value 0
Bit 1: Currently unused. Will have value 0
Bit 0: Was “FID” matched for biometric auth?

- pid_version – Version string of the PID block which was part of input
- timestamp – PID “ts” string which was part of input
- fmrcount – Total number of FMR records which was part of input
- fircount – Total number of FIR records which was part of input
- iircount – Total number of IIR records which was part of input
- fidcount – Total number of FID records which was part of input
- auth_api_ver – Version string of auth XML which was part of input
- SHA-256 of ASA code – Hash value of the ASA code
- SHA-256 of AUA code – hash value of the AUA code
- SHA-256 of SUB-AUA code – Hash value of “sc” attribute of input
- lang – same as “lang” attribute of input. If input did not have it, this will contain value “NA”.
- pi-ms – Match strategy used. Same as “ms” attribute of “Pi” element. If input did not have it, this will contain value “NA”.
- pi-mv – Match value used. Same as “mv” attribute of “Pi” element. If input did not have it, this will contain value “NA”.
- pi-lmv – Local language match value used. Same as “lmv” attribute of “Pi” element. If input did not have it, this will contain value “NA”.
- pa-ms – Match strategy used. Same as “ms” attribute of “Pa” element. If input did not have it, this will contain value “NA”.
- pa-lmv – Local language match value used. Same as “lmv” attribute of “Pa” element. If input did not have it, this will contain value “NA”.
- pfa-ms – Match strategy used. Same as “ms” attribute of “Pfa” element. If input did not have it, this will contain value “NA”.
- pfa-mv – Match value used. Same as “mv” attribute of “Pfa” element. If input did not have it, this will contain value “NA”.
- pfa-lmv – Local language match value used. Same as “lmv” attribute of “Pfa” element. If input did not have it, this will contain value “NA”.

- tid – P if public device, R is for Registered device, NA if no device is used.
- rdsId – Registered Device Service ID. If registered devices is not used, this will be NA.
- rdsVer – Registered Device Service version. If registered devices is not used, this will be NA.
- dpId – Registered Device Provider ID. If registered devices is not used, this will be NA.
- mi – Registered Device Provider Model ID. If registered devices is not used, this will be NA.
- SHA-256 of dc – Unique code of the device (dc) in SHA-256 form.
- RD Level – Registered Devices Certification Level (L0 or L1). If registered devices is not used, this will be NA.
- wadh – Wrapper API data hash. If not available, this will be NA.

Example structure given below:

```
04{NA,V,f25073ce9d46b0f720d00f32d8979c4efdab5346868ffac90f4412d02710f
7ef,0100002020000000,2.0,20160603104809,1,0,0,0,2.0,1f5368b4cf6d74290
33a47b8c7963329945c2bdf2690fa3685945b15d3cda2e0,96cae35ce8a9b0244178b
f28e4966c2ce1b8385723a96a6b838858cdd6ca0a1e,,NA,P,50,NA,E,NA,NA,NA,NA
,NA,NA,NA,NA,NA,NA,af76e1ffcb2e308770ac5212acbbc7d93ba5693d828714a513
6b6e1a9f438fc3,NA,NA}
```

- **err**– Failure error code. If authentication fails ("ret" attribute value is "n"), this attribute provides any of the following codes:
  - **"100"** – "Pi" (basic) attributes of demographic data did not match.
  - **"200"** –"Pa" (address) attributes of demographic data did not match.
  - **"300"** – Biometric data did not match.
  - **"310"** – Duplicate fingers used.
  - **"311"** – Duplicate Irises used.
  - **"312"** – FMR and FIR cannot be used in same transaction.
  - **"313"** – Single FIR record contains more than one finger.
  - "**314**" – Total number of FMR/FIR should not exceed 10.
  - "**315**" – Number of IIR should not exceed 2.
  - "**316**" – Number of FID should not exceed 1.
  - "**317**" – Number of biometric modalities (Face/Finger/IRIS) should not exceed 2.
  - "**318**" – BFD transaction should not contain other modalities in input.
  - "**330**" – Biometrics locked by Aadhaar number holder.
  - **"331**" – Aadhaar locked by Aadhaar number holder. This mean that all permanent forms of identifiers are locked for any kind of authentication. Resident can still use VID.
  - **"332**" – Aadhaar number usage is blocked by Aadhaar number holder.
  - **"333" –** Mismatch in "posh" attributes mentioned for FMR and FIR. Value of "posh" if mentioned other than "default" (when both FMR and FIR is used) should have the same finger position for FMR as that of corresponding FIR.

- **"334" –**Mismatch in count of FMR and FIR. ie Number of FMRs and FIRs should be same inside PID.
- "**400**" – Invalid OTP value.
- "**402**" – "txn" value did not match with "txn" value of Request OTP API.
- "**403**" – Maximum number of attempts for OTP match is exceeded or OTP is not generated. Please generate a fresh OTP and try to authenticate again.
- "**430**" – TOTP usage is not allowed for this aadhaar holder. Please install m-aadhaar and generate TOTP.
- **"500"** – Invalid encryption of session key.
- **"501"** – Invalid certificate identifier in "ci" attribute of "Skey".
- **"502"** – Invalid encryption of PID.
- **"503"** – Invalid encryption of Hmac.
- **"504"** – Session key re-initiation required due to expiry or key out of sync.
- **"505"** – Synchronized Key usage not allowed for the AUA.
- **"510"** – Invalid Auth XML format.
- **"511"** – Invalid PID XML format.
- **"512"** – Invalid consent value in "rc" attribute of "Auth".
- **"513"** – Invalid Protobuf Format
- "**514**" – Invalid UID token in input.
- "**515**" – Invalid VID Number in input.
- "**516**" – Invalid/Non Decryptable ANCS Token in input. Use correct ANCS token in request.
- "**517**" – Expired VID is used in input.
- "**518**"– ANCS Token in input is already used or expired.
- "**519**"– Inappropriate ANCS token used. This means that the ANCS token used is not associated with the AUA or with the transaction ID.
- **"520"**– Invalid "tid" value.
- **"521"** – Invalid "dc" code under Device tag.
- **"524**" – Invalid "mi" code under Device tag.
- **"527"** – Invalid "mc" code under Device tag.
- "**528**" – Device key rotation related issue.
- **"530"** – Invalid authenticator code.
- "**531**" – Invalid Sub-AUA.
- "**532**" – VID is not yet generated. To use specific services like UID lock which require VID to be pre generated, please generate a VID before using service.
- **"540"** – Invalid Auth XML version.
- **"541"** – Invalid PID XML version.
- **"542"** – AUA not authorized for ASA. This error will be returned if AUA and ASA do not have linking in the portal.
- **"543"** – Sub-AUA not associated with "AUA". ie The code specified in "sa" attribute is not added as "Sub-AUA" in portal or is not associated with an AUA.
- **"550"** – Invalid "Uses" element attributes.
- **"552"** –Invalid "wadh" element.
- **"553"** – RD service used is not allowed for the AUA.
- "**554**" –Public devices are not allowed to be used.
- "**555**" – rdsId is invalid and not part of certification registry.

- "**556**" – rdsVer is invalid and not part of certification registry.
- "**557**" – dpId is invalid and not part of certification registry.
- "**558**" – Invalid dih
- "**559**" – Device Certificate has expired
- "**560**" – DP Master Certificate has expired
- **"561"**– Request expired ("Pid->ts" value is older than *N* hours where *N* is a configured threshold in authentication server).
- **"562"**– Timestamp value is future time (value specified "Pid->ts" is ahead of authentication server time beyond acceptable threshold).
- **"563"** – Duplicate request (this error occurs when exactly same authentication request was re-sent by AUA).
- **"564"** – HMAC Validation failed.
- **"565"** – AUA license has expired.
- **"566"** – Invalid non-decryptable license key.
- **"567"** – Invalid input (this error occurs when unsupported characters were found in Indian language values, "lname" or "lav").
- **"568"**– Unsupported Language.
- **"569"** – Digital signature verification failed (means that authentication request XML was modified after it was signed).
- **"570"**– Invalid key info in digital signature (this means that certificate used for signing the authentication 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).
- **"571"** –PIN requires reset.
- "**572**" – Invalid biometric position.
- **"573"** – Pi usage not allowed as per license.
- **"574"**– Pa usage not allowed as per license.
- **"575"**– Pfa usage not allowed as per license.
- **"576"** - FMR usage not allowed as per license.
- **"577"** – FIR usage not allowed as per license.
- **"578"** – IIR usage not allowed as per license.
- **"579"** – OTP usage not allowed as per license.
- **"580"** – PIN usage not allowed as per license.
- **"581"** – Fuzzy matching usage not allowed as per license.
- **"582"** – Local language usage not allowed as per license.
- **"586"** – FID usage not allowed as per license.
- **"587"** – Name space not allowed.
- **"588"** – Registered device not allowed as per license.
- **"590"** – Public device not allowed as per license.
- "**591**" – BFD usage is not allowed as per license.
- "**592**" – Device blocked for more than allowed error percentage.
- "**593**" – Device blocked for more than allowed velocity.
- **"710"** – Missing "Pi" data as specified in "Uses".
- **"720"** – Missing "Pa" data as specified in "Uses".
- **"721"** – Missing "Pfa" data as specified in "Uses".
- **"730"** – Missing PIN data as specified in "Uses".
- **"740"** – Missing OTP data as specified in "Uses".
- **"800**" – Invalid biometric data.
- **"810"** – Missing biometric data as specified in "Uses".

- **"811"** – Missing biometric data in CIDR for the given Aadhaar Number/Virtual ID.
- **"812"** – Aadhaar number holder has not done "Best Finger Detection". Application should initiate BFD to help Aadhaar number holder identify their best fingers.
- **"820"** – Missing or empty value for "bt" attribute in "Uses" element.
- **"821"** – Invalid value in the "bt" attribute of "Uses" element.
- **"822"** – Invalid value in the "bs" attribute of "Bio" element within "Pid".
- **"901"** – No authentication data found in the request (this corresponds to a scenario wherein none of the auth data – Demo, Pv, or Bios – is present).
- **"902"** – Invalid "dob" value in the "Pi" element (this corresponds to a scenarios wherein "dob" attribute is not of the format "YYYY" or "YYYY-MM-DD", or the age is not in valid range).
- **"910"** – Invalid "mv" value in the "Pi" element.
- **"911"** – Invalid "mv" value in the "Pfa" element.
- **"912"** –Invalid "ms" value.
- **"913"** – Both "Pa" and "Pfa" are present in the authentication request (Pa and Pfa are mutually exclusive).
- **"914"** – Face alone is of allowed as biometric modality. You should send face along with another biometric modality like Finger or IRIS or OTP.
- **"915"** – Face auth is not allowed for this age of resident.
- **"916"** – Invalid face Image format in input.
- **"917"** – Invalid face capture type.
- **"930 to 939"** – Technical error that are internal to authentication server.
- **"940"** – Unauthorized ASA channel.
- **"941"** – Unspecified ASA channel.
- **"950"** – OTP store related technical error.
- **"951"** – Biometric lock related technical error.
- **"980"** – Unsupported option.
- **"995"** – Aadhaar suspended by competent authority.
- **"996"** –Aadhaar cancelled (Aadhaar is not in authenticable status).
- **"997"** –Aadhaar suspended (Aadhaar is not in authenticatable status).
- **"998"** –Invalid Aadhaar Number/Virtual ID/ANCS/UID TOKEN.
- **"999"** – Unknown error.

Element: **BfdRanks (*this element will be present ONLY when doing BFD*)**

- Biometric matching ranks for each finger that was part of input for BFD service.

Element: **BfdRank**

- Rank element for each finger ordered from top ranked to bottom ranked fingers.

Attributes:

- **pos** – Finger position for which matching rank is provided. Possible values are same as that of "pos" attribute within "Bio" element of Authentication PID XML.

- **value**– This attribute indicates a value between 1 and 10. This attribute indicates the order of preference in which resident should use his/her fingers for authentication. Value 1 indicates first choice and 10 indicates last choice.

NOTE: If some of the fingers are not match able at all (that means those fingers cannot be used for authentication), then records corresponding to those fingers will not be part of the "Ranks" XML. So, it is important to note that number of "Rank" elements may be less than what was provided in the input.

4. API and Data Security

Aadhaar authentication requests containing biometrics can only be originated from a "Registered" device. See registered Devices Specification for details of how they differ in working with biometric input.

4.1 Authentication Data Security

PID block data should be encrypted with a dynamic session key using **AES-256**symmetric algorithm (AES/GCM/NoPadding). Session key, in turn, is encrypted with**2048-bit UIDAI public key** using asymmetric algorithm (RSA/ECB/PKCS1Padding).Reference implementation demonstrates this in detail. Session key must not be stored anywhere except in memory and should not be reused across transactions. Only re-use of session key that is allowed is its use as seed key when using synchronized session key scheme. To increase assurance multi-factor authentication using one-time pin(OTP) could also be used in conjunction with biometrics. See Request OTP API Specification for details on requesting OTP.

GCM encryption - The GCM encryption is performed as follows:

1. IV or nonce for GCM encryption must be used for encryption. Last 12 bytes of the ts (String formatted date) is used as the IV or nonce. Refer ts attribute in the PID element.
2. AAD should also be used for GCM. Last 16 bytes of the ts (String formatted date) is used as the AAD.
3. Now encrypt and append the authentication tag to the end of the cipher text.
4. Now the ts is appended to the final encrypted pid and then its base64encoded.

The encryption flow when using biometrics is as defined below:

1. Aadhaar Number/Virtual ID, demographic, and biometric details as required by the application are entered into the application along with other factors such as OTP if it is used.
2. AUA/Sub-AUA application calls the registered Device service to obtain encrypted PID block and other device info.
3. AUA application sends the encrypted PID block along with HMAC, Skey, Device attributes, etc. data to AUA server.
4. AUA server forms the final authentication XML input for API including license key, digitally signs the XML, and sends the data to Aadhaar authentication server through an ASA network.
5. Aadhaar authentication server validates input, decrypts data, validates decrypted data, does the matching, creates audit, and responds with a "yes/no" as part of the digitally signed response XML.

4.2 Using Binary format for PID block

Protocol Buffers are a way of encoding structured data in an efficient yet extensible format. Google uses Protocol Buffers for almost all of its internal RPC protocols and file formats. It provides a flexible, efficient, automated mechanism for serializing structured data. .

This version of the API supports PID block to be sent to AUA server in protobuf format as an alternate to default XML format. This allows compact binary representation of the device data and avoids extra encoding required for XML format. If this scheme is used, device applications are expected to form the PID block in protobuf format using the ".proto" file for PID block.

4.3 Authentication Audits

Aadhaar authentication system records all the requests and their responses for audit purposes. See Aadhaar regulations to see audit retention policy of UIDAI.

All authentication responses are digitally signed by UIDAI and AUA's are recommended to validate the response integrity the keep track of these for audit purposes. In addition, attributes "ts", "info" within the API response can be used to verify if it the request was indeed for a particular Aadhaar Number/Virtual ID, if the request indeed had a biometric factor, when was the authentication done, etc. Such self-verifiability of the authentication response allows 3rd party applications to trust and electronically verify the digitally signed response quite similar to that of an offline trust establishment against a gazetted officer signed paper.

5. Best Finger Detection

This chapter describes the principles of Best Finger Detection (BFD), its usage, and how to do BFD using authentication API.

5.1 Best Finger

When authenticating a resident using any single finger, the accuracy or the chances of being matched would be different due to differences in quality across all his/her fingers. This variation may also be present due to the manner in which the resident normally interacts with a typical fingerprint scanner and the different fingers may inherently have different amount of identifying information depending on the size of the finger and the commonness of the pattern it carries. It may thus be useful to appraise each resident of finger providing the best accuracy and successful matching results. We shall refer to this finger with best accuracy as the **best finger**. Resident may possess one or more best fingers. This knowledge allows the resident to provide his/her best finger(s) during authentication thereby increasing the chances of successful match.

5.2 Best Finger Detection (BFD)

Since many residents in India are engaged in manual labour, the quality of fingerprints vary considerably even between fingers of the same resident. So it is important to identify the best finger(s) to improve authentication accuracy and hence be more inclusive in supporting Aadhaar authentication across all sections of society.

The Best Finger for a resident is the one that, when selected for authentication, provides the highest chance of successful authentication for that resident. The best finger to be used for authentication depends on the intrinsic qualities of the finger (e.g. ridge formation, how worn out they are, cracked, etc.), as well as the quality of images captured during enrolment process and the authentication transaction.

5.3 Best Finger Detection Process

AUAs who use fingerprint based Aadhaar authentication within their applications should implement BFD application as part of their Aadhaar biometric authentication enabled applications. BFD process helps the resident understand how Aadhaar biometric authentication work and which of their fingers are best usable.

There are two scenarios under which BFD application needs to be used:

1. Authentication API returns error and action codes asking resident BFD to be done (error code “812”, see Aadhaar Authentication API Specification 2.5). Whenever this error comes back, either automatically or by operator, BFD application must be launched and have the resident do the BFD to identify his/her best fingers.
2. If resident specifically wants to get his best finger identified and get a BFD receipt, proactively BFD application could be launched and used. This may

happen due to authentication errors even after initial BFD typically resulting from wear and tear of fingers, climatic conditions, etc. or due to the fact that resident may have re-enrolled updating his/her biometrics in Aadhaar system.

AUA/Sub-AUA application initiating BFD should do the following:

1. Capture all fingerprints of the Aadhaar number holder using registered device.
    - During the capture of fingerprints by the RD service, all captured fingerprint images should be subjected to image quality check to make sure it is a good capture.
    - Use "posh" attribute to indicate the labelled position of the finger.PID block MUST NOT contain demo data or Iris/face data or OTP.
    - Only FMR data should be allowed in case of BFD.
2. Ensure resident and operator clearly knows which finger is being scanned.
    - When scanning all 10 fingers together, operator must ensure resident is not placing finger that is already scanned again by mistake.
    - Operator must make sure resident is placing the finger correctly for enabling best capture.
3. Once all fingers are captured, RD service creates the encrypted PID block and returns it back to the BFD application.
4. AUA/Sub-AUA application invokes the Authentication API through AUA server.
    - **When doing BFD, the "txn" attribute MUST START WITH "ubfd:" as the namespace**.
    - When doing BFD, PID block MUST NOT contain demo data or Iris/face data or OTP. It must contain only fingerprint records.
5. Based on the response, provide a receipt to the resident indicating the ranking for each finger.
6. Provide for exception where resident may not have all ten fingers.
7. BFD application must provide a receipt UI indicating rank details from API response, preferably with a picture of the hand

BFD server at UIDAI end processes incoming requests as described below:

1. Fingers are ranked in descending order based on the matching score.
2. BFD application feedback helps resident to clearly identify which of his/her fingers are good for authentication.
3. Resident is expected to use his/her fingers in the order of rank starting from one while doing authentication.

NOTE: Above is a high level logic and UIDAI may decide to enhance this logic at the backend from time to time. Change of such logic will not have any impact on the actual API input/output. It is provided for general understanding of how BFD works.

6. Appendix

6.1 Changes in Version 2.5 from Version 2.0 Revision 1

New (2.5)
Usage of VID and UID token incorporated
Info attribute ver changed to 04 from 03.
Incorporated BFD feature into core authentication
Additional error codes added as part of VID/Face Authentication/BFD
Incorporated changes to allow FMR and FIR in the same PID block (Jan 2022 Revision 1)
Added/Modified error codes