

eSign Remote API Specifications

Version 1.1

09 Sep 2026



Controller of Certifying Authorities
Ministry of Electronics and Information Technology

Document Control

Document Name	eSign Remote API Specifications
Status	Release
Version	1.1
Release date	09.09.2026
Last update	09.09.2026
Document Owner	Controller of Certifying Authorities, India

Table of Contents

1. Introduction	1
1.1. Target Audience	1
1.2. Objective of the document	1
1.3. Terminology	1
1.4. Legal Framework.....	1
2. Understanding eSign Remote Service	2
2.1. eSign Remote Service at a glance	2
2.2. Participants	2
2.3. Detailed Framework.....	2
2.3.1. Architecture Diagram	3
2.3.2. eSign User	3
2.3.3. SAM & HSM.....	4
2.3.4. Key Protocols	4
2.3.5. Key Management Server (KMS)	5
2.3.6. ESP Database Server	5
3. eSign Remote Service Implementation Scenarios.....	5
3.1. Implementation Scenarios	5
3.1.1. Remote key-storage hosted at CA	5
3.1.2. Remote key-storage hosted at TTP	5
3.2. Process Flow.....	5
3.2.1. Certificate Enrolment process	5
3.2.2. Authentication & Signing process.....	6
4. eSign Remote API for ASPs.....	7
4.1. Certificate Enrolment.....	7
4.2. Initiate eSign	8
4.2.1. eSign API: Input Data Format - eSign Service.....	8
4.2.1.1. eSign XML structure	8
4.3. eSign: User Authentication Page	11
4.4. eSign API: Response Data Format - eSign Service.....	11
4.4.1. Element Details	12
4.5. eSign API: Check Signing Status - Request	13
4.5.1. Request XML format	14
4.5.1.1. Element Details	14
5. eSign Remote API for ESPs	14
5.1. KYC Account: Initiate Authentication.....	15
5.2. KYC Account: Authenticate (HTTPS API)	15

5.3. CA: Generate Certificate (HTTPS API)	15
5.4. SAM: Initiate Enrolment (CHI protocol)	16
5.5. SAM: Enrol Certificate (CHI protocol)	16
5.6. SAM: Initiate Signing (SHI protocol)	16
5.7. SAM: Authenticate user (AI protocol)	16
5.8. SAM: Perform Signing (SHI protocol)	17
6. Change History	17

1. Introduction

The Government introduced enabling law in the second schedule of IT Act, in 2015 , known as “e-authentication technique using Aadhaar and other e-KYC services”. eSign API 2.x and eSign API 3.x are based on the above-mentioned notification. In the year 2020, the IT Act Gazette notification titled “e-authentication technique and procedure for creating and accessing subscriber's signature key facilitated by trusted third party” was published to enable subscribers to store their private key in a remote HSM managed by CA. This eSign Remote API is addressing the functions relating to remote key storage, authentication to HSM and signature functions. The pre-requisite for this API will be the operational eKYC account created using eSign API 3.x

In this model, key pair is stored in remote HSM managed by Certifying Authorities. As in the case of token, subscribers generate the key-pair and obtain Digital Signature certificate from CA once and the same key will be used subsequently after authentication to HSM. The key-pair generation and subsequent usage of private key for the signature will be under the sole activation control of the subscriber though HSM may be under the physical custody or managed by CA.

1.1. Target Audience

This technical document is targeted to CA, who implement the infrastructure for remote key storage and Application Service Providers who require signing of digital document(s) in their application.

1.2. Objective of the document

This document provides eSign Remote Service API specification for remote key management and authentication functions. This includes necessary API Data format, protocol and other related specifications.

1.3. Terminology

SAM

Subscriber Authentication Module (SAM) is software which shall reside in the FIPS boundary of HSM. SAM shall utilize the secure channel establishment function of the HSM to securely communicate (through eSign Service providers) with the subscribers, including during the subscriber authentication.

" eSign" or “eSign Service” is an online Electronic Signature Service in which the key pair generation, certification of the public key by the CA and digital signature creation for electronic document are facilitated by the eSign online Electronic Signature Service provider instantaneously within a single online service based on successful authentication of individual using e-KYC services. If the remote key storage by CA is enabled for subscriber, the eSign service will be based on authentication of subscriber’s key and corresponding certificate.

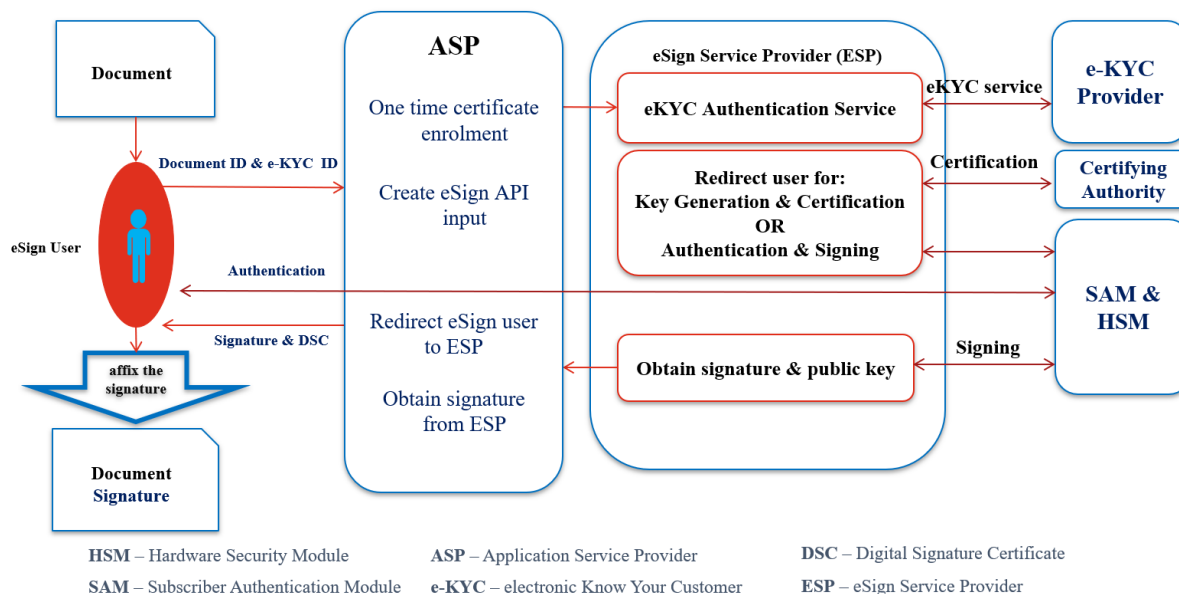
1.4. Legal Framework

eSign service based on remote key-storage will operate under the provisions of the Second Schedule of Information Technology Act, 2000 (“e-authentication technique and procedure for creating and accessing subscriber's signature key facilitated by trusted third party”).

2. Understanding eSign Remote Service

This chapter describes eSign Service based on remote key-storage, the remote key management and authentication follow in subsequent chapters.

2.1. eSign Remote Service at a glance



2.2. Participants

The various participants involved in the framework of eSign Remote are described as under.

eSign User: eSign user is the Subscriber for Digital Signature Keys as defined under the Act. eSign user is required to have enrolled successfully with the Certifying Authority under the eKYC requirements, and thus have the corresponding eKYC account functional. eSign User shall meet the verification requirements for KYC account as per Identity Verification Guidelines of CCA.

Application Service Provider (ASP): ASP is the front ending application to the eSign User, necessarily a software application, facilitating digital signing to the eSign user. This application is required to comply with the on-boarding requirements laid down by CCA.

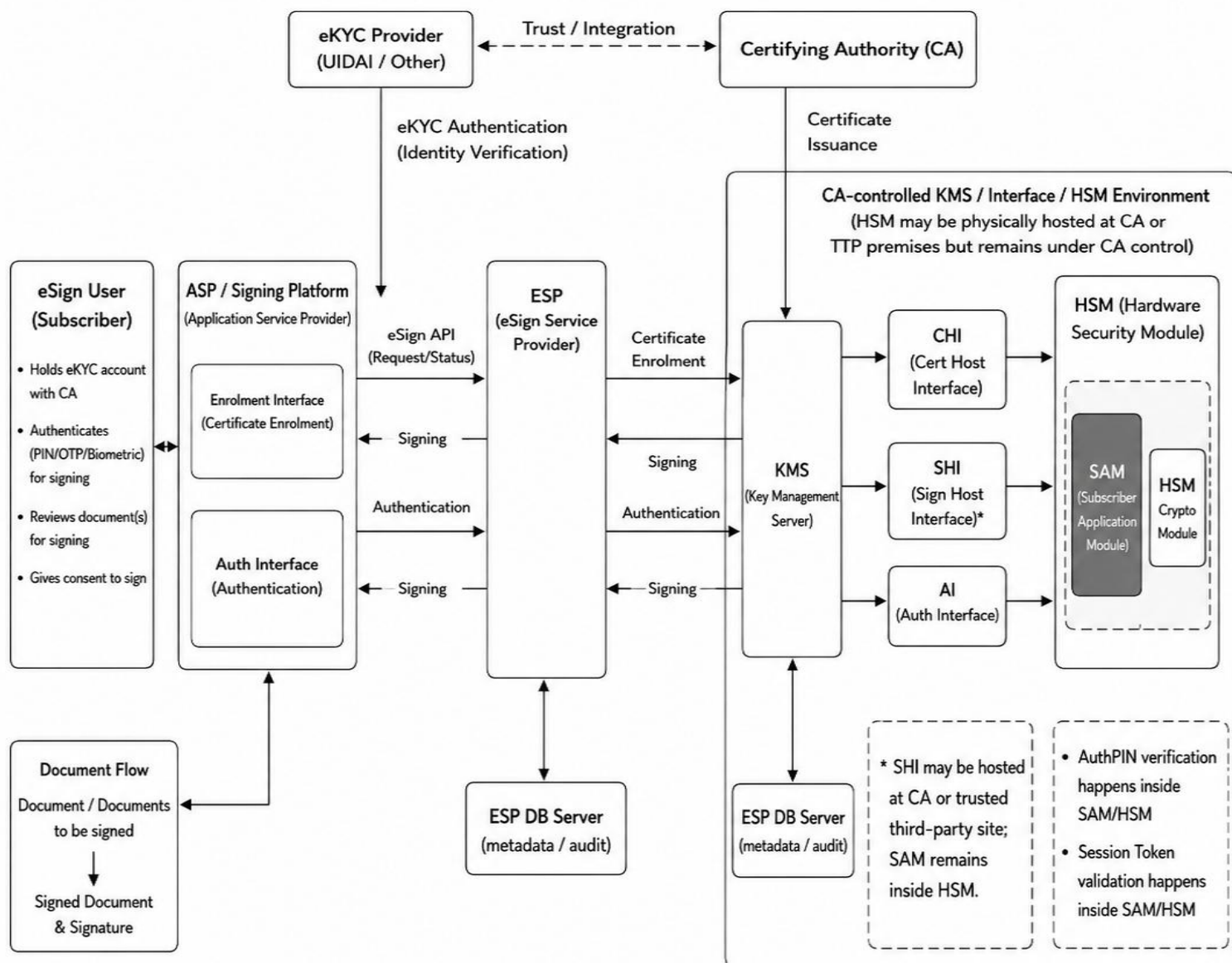
eSign Service Provider (ESP): ESP is the empanelled/licensed entity by CCA to act as an eSign Service Provider.

Certifying Authority: Certifying Authority is the entity empanelled/licensed by CCA to perform the duties of Certifying Authority under the provisions of the Act.

2.3. Detailed Framework

This chapter describes the eSign Remote API framework.

2.3.1. Architecture Diagram



2.3.2. eSign User

This framework expects that the user has eKYC account and successfully authenticated to eKYC account for the purposes of Certificate Enrolment in HSM. The enrolment to eKYC account needs to be carried out independent of this document.

2.3.3. SAM & HSM

Subscriber Authentication Module (SAM) shall reside within the HSM-protected boundary. The HSM may be physically hosted at the CA premises or at application owner/Trusted Third Party (TTP) premises, but shall remain under the sole administrative control of the CA. The implementation shall meet the requirements laid down under the e-Authentication Guidelines of CCA. Key Management Server (KMS) interfaces with SAM/HSM for executing the functions.

HSM shall be FIPS 140-3 Level 3 validated/certified and shall secure the signing keys of the eSign Users. Subscriber private keys shall not be exported in cleartext. Keys output for backup or migration shall be HSM-wrapped under a non-exportable HSM key and unwrappable only within an authorised FIPS-validated HSM environment. HSM shall also host the cryptographic keys used by SAM for authentication protection. Access to signing and authentication-protection keys shall be limited to SAM and shall not be available to any external component with administrative or overriding privileges.

SAM Component shall reside within the HSM-protected boundary and shall perform subscriber authentication and key-activation controls. SAM may access the database for eSign User, Certificate ID, Key ID and other non-secret metadata. AuthPIN verification material shall be maintained with the eKYC ID in HSM-protected storage and shall not permit recovery of the AuthPIN. SAM shall not log or expose AuthPIN or authentication secrets outside the SAM/HSM-protected boundary.

The SAM and HSM shall never be exposed to Public Internet, and shall attend incoming requests only through Key Management Server.

2.3.4. Key Protocols

The framework implements secure protocols through defined interfaces for interaction between HSM/SAM and the eSign User/ASP through Key Management Server (KMS)

The CHI, SHI and AI software interfaces shall operate outside SAM and interface with SAM/HSM through KMS. SAM shall remain within the HSM-protected boundary. All communication between these interfaces and SAM/HSM shall be signed and encrypted over secure channels with mutual authentication and Perfect Forward Secrecy (PFS).

Auth Interface (AI): This interface supports subscriber authentication between the eSign User and SAM/HSM through KMS. The AuthPIN shall be entered through the KMS-controlled secure interface, encrypted with the authenticated SAM Public Key, bound to the Transaction ID and securely forwarded to SAM. AuthPIN verification shall occur only within the SAM/HSM-protected boundary. A unique random nonce shall be used for freshness. On successful authentication, SAM shall issue a single-use, time-bound Session Token bound to the Transaction ID and the exact authorised document hash(es) and count.

CA HSM Interface (CHI): This interface interacts between ESP, CA & the HSM for the Certificate Enrolment functions. During this process, it interacts with eSign User to capture certificate enrolment related information, eKYC Account Authentication, Key Generation/Management, Certificate Generation (from CA), Auth PIN enrolment, Auth PIN reset functions. The interface also extends through KMS to directly interact with the eSign user in ASP application.

Signing HSM Interface (SHI): This interface supports signature functions between KMS and SAM/HSM. SHI shall forward the Session Token and authorised document hash(es) for signing. SAM shall validate the Session Token within the SAM/HSM-protected boundary; only after successful validation shall SHI request HSM signing. SHI shall not independently validate the Session Token or access the subscriber private key. SHI may be hosted at the CA or at a trusted third-party site.

2.3.5. Key Management Server (KMS)

ESP shall provide a Key Management Server (KMS) to expose the protocol interfaces on necessary manner, to fulfil the requirements of eSign Remote API to the ASPs. This KMS shall prevent directly exposing SAM & HSM to public internet. For this purpose, the HTTP interface exposed by this KMS will be the only channel for interaction with the ASP / eSign User.

While KMS frontends Enrolment/Authentication of the eSign User, it shall use the authenticated SAM Public Key for protection of AuthPIN. KMS shall not perform entry-level hashing of AuthPIN. AuthPIN shall be encrypted, bound to the Transaction ID and securely forwarded to SAM. KMS shall never store or log the clear AuthPIN/Secret.

2.3.6. ESP Database Server

ESP shall facilitate a Database Server to store/manage eSign User and transaction metadata required by SAM/KMS. AuthPIN verification material shall not be stored as a recoverable credential in the database; it shall be maintained with the eKYC ID in HSM-protected storage. The database may reside within the protected environment or with KMS, subject to the e-Authentication Guidelines.

3. eSign Remote Service Implementation Scenarios

This chapter describes the eSign Remote Service implementation scenarios & the process flow.

3.1. Implementation Scenarios

The eSign Remote Service acts as an extension to eSign Service API 3.x in case of key management related functions and can work independently at CA premises or TTP location for key authentication and signature creation. The API is common to both scenarios; however, the parameter values that will vary for each ESP/TTP are 'eSign Service URL' and 'CERT ID' (Unique CERT ID issued by the CA/ESP).

3.1.1. Remote key-storage hosted at CA

In this case, access to the subscriber key for authentication and signing shall be provided after successful authentication to the eKYC account. CHI, AI and SHI shall operate in CA-controlled infrastructure and interface securely with SAM/HSM. SAM shall remain within the HSM-protected boundary.

3.1.2. Remote key-storage hosted at TTP

In this case, access for signing shall be provided after the required authentication at the trusted third-party application. The key-management function shall continue after authentication to the eKYC account maintained by the CA. SHI may be hosted at the trusted third-party site, while CHI and AI remain CA-controlled. SAM shall remain within the HSM-protected boundary and the HSM shall remain under the sole administrative control of the CA.

3.2. Process Flow

The process flow consists of two stages, that is, Certificate Enrolment process by the eSign User (usually a one-time process per certificate) and Authentication & Signing process.

3.2.1. Certificate Enrolment process

During this process:

1. Prerequisite:
 - a. User shall have a valid eKYC Account with the ESP/CA.
2. User visits ESP / ASPs application and initiates enrolment.
3. User is redirected to KMS interface for Certificate Enrolment.

- a. User performs the KYC Account authentication (2-factor) through KMS interface.
 - i. KMS communicates with KYC Provider to authenticate.
 - ii. KMS Receives eKYC Response.
 - b. KMS invokes CHI based user enrolment and requests for new Certificate Auth PIN.
 - i. CHI provides the authenticated SAM Public Key to KMS.
 - ii. KMS encrypts AuthPIN with the authenticated SAM Public Key, binds it to the Transaction ID and securely forwards it to SAM.
 - c. CHI associates the KYC User ID with the encrypted AuthPIN and enrolment context; key algorithm and validity may be provided as applicable (default SHA256/ECC & 2 years).
 - d. CHI generates new Key Pair and CSR
 - e. KMS communicates with CA for the certificate with given CSR and eKYC response.
 - f. CA generates the certificate for desired validity.
 - g. User Accepts the Certificate.
 - h. CHI completes Certificate enrolment with the corresponding key. SAM registers non-recoverable AuthPIN verification material with the eKYC ID in HSM-protected storage.
4. User is redirected back to ESP/ASP's application.

3.2.2. Authentication & Signing process

During this process:

1. Prerequisite:
 - a. User shall have a valid eKYC Account with the ESP/CA.
 - b. User shall have a valid Certificate Enrolment in the HSM.
2. User visits ASP's application and initiates Signature.
3. User shall perform initial authentication in ASP's application prior to or after initiating signature. User shall be in a valid authenticated session before the next step.
4. ASP shall initiate an eSign Transaction with Document hash(es) (maximum 10), Unique Transaction ID, and optional KYC Username.
5. User is redirected to KMS interface for Signing with the Transaction ID.
 - a. User performs the Certificate Authentication (KYC Username & Certificate Auth PIN).
 - i. At this stage, KMS requests and initializes the authentication with SAM before capturing the details.
 - ii. SAM provides a unique random nonce and authenticated SAM Public Key to KMS.
 - iii. KMS encrypts the AuthPIN with the authenticated SAM Public Key, binds it to the Transaction ID and securely forwards it to SAM.
 - b. SAM receives the authentication data and performs authentication only within the SAM/HSM-protected boundary.
 - i. SAM decrypts the AuthPIN only within the SAM/HSM-protected boundary.
 - ii. SAM verifies the AuthPIN against non-recoverable verification material maintained in HSM-protected storage.
 - iii. SAM applies the authentication mechanism specified in the e-Authentication Guidelines using the fresh nonce/random value within the protected boundary.
 - iv. SAM enforces per-subscriber retry and lockout controls; failed or invalid sessions require re-authentication.
 - v. On success, SAM generates a single-use, time-bound Session Token bound to the Transaction ID and exact authorised document hash(es) and count,

- and responds to KMS. Any mismatch, reuse or timeout invalidates the token and requires re-authentication.
- c. KMS forwards the authorised Document Hash(es) and Session Token through SHI for signing.
 - i. SAM validates the Session Token within the SAM/HSM-protected boundary before signing.
 - ii. After successful SAM validation, SHI requests HSM signing of the authorised Document Hash(es). SHI shall not access the subscriber private key.
- d. KMS responds the Signature (and Corresponding Certificate) back to ASP.
- 6. User is redirected back to ASP's application.
 - a. ASP attaches the Signature to the original document and provides it to the user.

4. eSign Remote API for ASPs

This chapter describes the eSign Remote Service API in detail including the communication protocol, and data formats to be used by ASPs.

Following are the APIs exposed by KMS Application to ASP.

1. Certificate Enrolment: HTTPS Redirection
2. Initiate eSign: HTTPS API
3. eSign Authentication: HTTPS Redirection
4. Check Signing Status: HTTPS API

4.1. Certificate Enrolment

ASP shall initiate the Certificate Enrolment with a unique transaction ID. ESP shall receive such enrolment request for the transaction ID and facilitate Certificate Enrolment with HTTPS Redirection. At this stage, ASP is expected to guide the user with proper information before redirecting the user to the Certificate Enrolment page of the ESP.

ESP shall expose a redirection URL with following specifications.

Request:

API URL	ESP shall expose URL as HTTPS redirection page.
Protocol	HTTPS
Method	POST
Content-Type	application/x-www-form-urlencoded
Parameter name	txnreq
Parameter Value	Concatenated transaction ID and Redirection URL (separated with a pipe character) in base 64 encoding.
Example format	txnreq=Base64(transaction ID + " " + Redirection URL)
Data validation	Transaction ID: This shall be an alphanumeric string not exceeding 50 characters. Shall be unique for a given date ASP combination. Redirection URL: This shall be a valid HTTP(S) URL of the ASP, to which ESP shall redirect the user after enrolment steps.

ESP shall redirect back the user to ASP's interface with a appropriate response.

Response:

API URL	ASP shall expose URL as HTTP(S) redirection page.
Protocol	HTTP(S)
Method	POST
Content-Type	application/x-www-form-urlencoded

Parameter name	txnresp
Parameter Value	Concatenated transaction ID, Status, and Data (separated with a pipe character) in base 64 encoding.
Example format	txnresp=Base64(transaction ID + " " + Status + " " + Data)
Data validation	Transaction ID: This shall be the original transaction ID given by ASP. Status: This shall be 0 in case of failure. 1 in case of Success. Data: This shall contain the username of the user, in case of Success. Else, in case of Failure, this shall contain a Custom Error Message by ESP.

4.2. Initiate eSign

eSign Remote 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 this service. To support strong end to end security and avoid request tampering and man-in-the-middle attacks, it is essential that the requests and responses are digitally signed.

ESP shall provision a valid digital signature certificate of ASP application against the ASP account, in order to validate the requests.

The usage of HTTPS shall ensure transport layer encryption, while digital signing of XML shall ensure integrity & authenticity of data.

Following is the URL format and the parameters for eSign service:

API URL	ESP shall expose URL as HTTPS endpoint
Protocol	HTTPS
Method	POST
Content-Type	application/xml
Post data	A well-formed XML, as per the specifications provided in this document.

ASP is required to collect the necessary API URL from the respective ESP.

4.2.1. eSign API: Input Data Format - eSign Service

eSign Service uses XML as the data format for input and output.

4.2.1.1. eSign XML structure

Following is the XML data format for eSign XML.

```
<Esign ver="" signerid="" ts="" txn="" maxWaitPeriod="" aspld="" responseUrl="" redirectUrl="" >
  <Docs>
    <InputHash id="" hashAlgorithm="" docInfo="" docUrl=""
      responseSigType="">Document Hash in Hex</InputHash>
  </Docs>
  <Signature>Digital signature of ASP</Signature>
</Esign>
```

Element Details:

Element Name: Esign

- Description: Root element of the eSign xml
- Requirement of tag: Mandatory
- Value: Sub-elements
- Attributes: Table below

SI No	Attribute	Required?	Value
1.	Ver	Mandatory	eSign version (mandatory). ESP may host multiple versions for supporting gradual migration. As of this specification, API Version is "1.0".
2.	username	Optional	ASP collects the username of the signer. If username is not present, ESP may facilitate the new Certificate Enrolment, however KYC authentication of user should be carried out before Enrolment.
3.	ts	Mandatory	Request timestamp in ISO format. The value should be in Indian Standard Time (IST), and should be within the range of maximum 30 minutes deviation to support out of sync server clocks.
4.	txn	Mandatory	Transaction ID of the ASP calling the API, this is logged and returned in the output for correlation. Should be unique for the given ASP-ESP combination for that calendar day.
5.	maxWaitPeriod	Mandatory	Expiry time in minutes. This is maximum wait time for the ESP to allow Signer to complete the signing. In case the user does not sign within ASP's expected duration, ESP should mark the transaction as error 'User timeout' error code. Default = 1440 minutes
6.	aspld	Mandatory	Organization ID of ASP allocated by ESP.
7.	responseUrl	Mandatory	ASP URL to receive the response from ESP. This should be a valid URI accessible from ESP system to make a call and submit the response XML packet using HTTP(S)-POST with Content-Type as application/xml. On success or failure including cancellation by user, ESP shall perform a background call to this response URL with 'eSign Response Format' which contains the status success/failure (status = 1/0).
8.	redirectUrl	Optional	ASP URL to redirect the user after completion of transaction. This is supported only in case where ASP uses redirection to ESP authentication page. If present, ESP shall redirect the user back to ASP's designated URL. Such redirection shall have a HTTP(S)-POST and Content-Type of 'application/x-www-form-urlencoded' with parameter of 'txnref' containing concatenated transaction ID and

			responseCode (separated with a pipe character) in base 64 encoding. txnref=Base64(transaction ID + " " + responseCode)
--	--	--	---

Element Name: Docs

- Description: Contains one sub-element with Document Hash
- Requirement of tag: Mandatory
- Value: Sub-elements
- Attributes: Not applicable

Element Name: InputHash

- Description: Contains the value of Document Hash, which has to be signed.
- Requirement of tag: Mandatory
- Value: SHA256 hash value of the document in Hex format
- Attributes: Table below

SI No	Attribute	Required?	Value
1.	id	Mandatory	The index number of the document. Should start with one. Maximum 10. Should be sequential. Shall not repeat.
2.	hashAlgorithm	Mandatory	Should be fixed to "SHA256"
3.	docInfo	Mandatory	Description for the respective document being signed, not more than 50 characters. docInfo should be strictly adhere to the content of document. Multiple documents of same type or different types should not be included in a single file. ESP shall display this information against docUrl, so that user can identify the same.
4.	docUrl	Mandatory	URL of the document. Should be a HTTP / HTTPS URL for the document, accessible by the signer during the transaction permitted duration (maxUserWait Time). ESP shall display this URL with hyperlink, so that user can access the document to view.
5.	responseSigType	Mandatory	This value represents the response signature type, where ASP can request for a specific type of signature from one of the following Allowed Values are: 1. raw 2. PKCS7(with only the signer certificate in the certificate section and no revocation information) 3. PKCS7pdf(all issuer certificates up to and including root CA certificate and CRLs/OCSP responses of each issuer certificates should be included in the response. In case, the number CRL entries are more than 5, only OCSP

			responses are allowed. The revocation information should be included as a signed attribute under pdfRevocationInfoArchival (1.2.840.113583.1.1.8). The signature may also be optionally time stamped using the time stamping services of CA. 4. PKCS7complete(All issuer certificates & its revocation information in unsigned info)
--	--	--	--

Element Name: Signature

- Description: Contains the signature of ASP.
- Requirement of tag: Mandatory
- Value:
 - Signed value of Input XML, as per the W3C recommendation on XML Signature Syntax and Processing (Second Edition)
 - Refer <http://www.w3.org/TR/xmlsig-core/> for more information
- Attributes: Not applicable

4.3. eSign: User Authentication Page

Once ASP submits the Request XML, ESP provides a 'pending for completion' (status=2) response which will contain the response code (as an acknowledgement). At this stage, ASP is expected to guide the user with proper to redirect the user to the authentication page of the ESP.

ESP shall expose a redirection URL with following specifications.

API URL	ESP shall expose URL as HTTPS redirection page.
Protocol	HTTPS
Method	POST
Content-Type	application/x-www-form-urlencoded
Parameter name	txnref
Parameter Value	Concatenated transaction ID and responseCode (separated with a pipe character) in base 64 encoding.
Example format	txnref=Base64(transaction ID + " " + responseCode)

4.4. eSign API: Response Data Format - eSign Service

Below is the response format of eSign Service API. This response shall be used in following situations:

1. Once the subscriber authorizes (or cancels or expire), ESP shall provide a completed response to the ASP on the responseUrl (status = 1/0).
2. ESP shall also respond to 'Check Signing Status' API call with this response format including 'pending for completion' statuses.

Note that, the API does not give any identity related data of the eSign user.

```
<EsignResp ver="" status="" ts="" txn="" resCode=" " error="">
  <UserX509Certificate>base64 value of eSign user certificate (.cer)</UserX509Certificate>
  <Signatures>
```

```

        <DocSignature id="" sigHashAlgorithm="SHA256" error="">
        Signature data in raw (PKCS#1) or raw (ECDSA) or PKCS7 (CMS) signature as
        requested
        </DocSignature>
    </Signatures>
    <Signature>Signature of ESP</Signature>
</EsignResp>

```

4.4.1. Element Details

Element Name: EsignResp

- Description: This element is the root element of the response and contains the meta values.
- Value: Sub-elements
- Attributes: Table below

SI No	Attribute	Presence	Value
1.	ver	Mandatory	Should be set to 3.3
2.	status	Mandatory	In case of success, it will be "1" In case of failure, it will be "0" In case of pending for completion, it will be "2"
3.	ts	Mandatory	Will contain the response timestamp in ISO format.
4.	txn	Mandatory	The Transaction ID provided by ASP in the request.
5.	resCode	Mandatory	A unique response code provided by ESP. This is a unique id for the transaction (eSign user authentication & eSign request) provided by ESP. It shall make the transaction traceable, and ASP is expected to store this code in their audit log. The response code shall be maintained same for particular transaction. Being asynchronous, there may be need for providing the response multiple times including the acknowledgement stage and final response stage. All the responses shall carry same response code for the particular transaction.
6.	error	Optional	In case of failure, this will contain an error code. OR blank, in case of success.

Element Name: UserX509Certificate

- Description: This element will contain the Base 64 value of the Certificate. No private key information is shared. For manual verification, this value can be copied and saved as .cer file (With begin and end statements - PEM Format).
- Presence: Mandatory, if success.

- Value: Base 64 value of eSign user certificate (public).
- Attributes: Not Applicable

Element Name: Signatures

- Description: This element contains the sub-elements of signatures corresponding to InputHash.
- Presence: Mandatory, if success.
- Value: Sub-elements.
- Attributes: Not Applicable

Element Name: DocSignature

- Description: This element will contain the signed value which will be verifiable against original document.
- Presence: Mandatory
- Value: Signed value in raw (PKCS#1) or raw (ECDSA) or PKCS7 (CMS) signature format as per the request XML.
- Attributes: Table Below

SI No	Attribute	Presence	Value
1.	Id	Mandatory	Contains the corresponding ID to the Input Hash received
2.	sigHashAlgorithm	Mandatory	Should be fixed to "SHA256"
3.	error	Optional	In case of failure, this will contain an error code. OR blank, in case of success. ESP shall provide the signer an option to reject any document before Session Token issuance. The Session Token shall be bound only to the exact authorised document hash(es) and count. Rejected documents shall not be signed and shall be returned with the error "User Rejected".

Element Name: Signature

- Description: This element will contain the signature of ESP, which can be used for verification by ASP and protect the response from any kind of tamper.
- Value:
 - Signed value of response XML, as per the W3C recommendation on XML Signature Syntax and Processing (Second Edition)
 - Refer <http://www.w3.org/TR/xmlsig-core/> for more information
- Attributes: Not Applicable

4.5. eSign API: Check Signing Status - Request

This is an additional option for ASP to check the status of the transaction, in case necessary.

On a successful & timely flow, ESP will automatically call back the ASP's responseUrl with necessary eSign response. However, in case of any need, ASP can call the signing status API and receive the response again.

ESP shall provide this service for minimum of 30 days from the date of transaction, for the ASP.

4.5.1. Request XML format

```
<EsignStatus ver="" ts="" txn="" aspId="" >  
    <Signature>Digital signature of ASP</Signature>  
</EsignStatus>
```

4.5.1.1. Element Details

Element Name: Esign

- Description: Root element of the eSign xml
- Requirement of tag: Mandatory
- Value: Sub-elements
- Attributes: Table below

	Attribute	Required?	Value
1.	ver	Mandatory	Should be set to 3.3
2.	ts	Mandatory	Request timestamp in ISO format. The value should be in Indian Standard Time (IST), and should be within the range of maximum 30 minutes deviation to support out of sync server clocks.
3.	txn	Mandatory	Transaction ID of the ASP provided in original request.
4.	aspId	Mandatory	Organization ID of the ASP

Element Name: Signature

- Description: Contains the signature of ASP.
- Requirement of tag: Mandatory
- Value:
 - Signed value of Input XML, as per the W3C recommendation on XML Signature Syntax and Processing (Second Edition)
 - Refer <http://www.w3.org/TR/xmlsig-core/> for more information
- Attributes: Not applicable

This will respond with an eSign response data as defined in this document. The status attribute of the response will indicate the success or pending for completion.

5. eSign Remote API for ESPs

This chapter describes the eSign Remote Service API in detail including the communication protocol, and data requirements to be used by ESPs.

Following are the APIs exposed by SAM / KYC Provider / CA to ESP.

1. KYC Account: Initiate Authentication (HTTPS API)
2. KYC Account: Authenticate (HTTPS API)
3. CA: Generate Certificate (HTTPS API)
4. SAM: Initiate Enrolment (CHI protocol)
5. SAM: Enrol Certificate (CHI protocol)

6. SAM: Initiate Signing (SHI protocol)
7. SAM: Authenticate user (AI protocol)
8. SAM: Perform Signing (SHI protocol)

For security reasons, these APIs are not detailed with data formats, but are given with data requirements. The implementation shall use secure transport, mutual authentication where applicable, transaction binding and freshness controls to prevent replay attacks.

5.1. KYC Account: Initiate Authentication

Protocol	HTTPS API
Request Fields	• Unique Access Credentials for each deployment • Unique Transaction ID • Time Stamp • KYC Username
Response Fields	• Corresponding Transaction ID • Response Time Stamp • Status • KYC Username • Unique Nonce (128 bits or longer in accordance with RFC 2104) • Error (if any)

5.2. KYC Account: Authenticate (HTTPS API)

Protocol	HTTPS API
Request Fields	• Unique Access Credentials for each deployment • Transaction ID correlating to Initialization • Time Stamp • KYC Username • PIN hashed along with Nonce with minimum of SHA256. • Second Factor data
Response Fields	• Corresponding Transaction ID • Response Time Stamp • Status • KYC Username • KYC Information • Error (if any)

5.3. CA: Generate Certificate (HTTPS API)

Protocol	HTTPS API
Request Fields	• Unique Access Credentials for each deployment • Unique Transaction ID • Time Stamp • KYC Username • KYC information • CSR for the Certificate
Response Fields	• Corresponding Transaction ID • Response Time Stamp • Status • KYC Username • Certificate Data • Error (if any)

5.4. SAM: Initiate Enrolment (CHI protocol)

Protocol	Custom Protocol
Request Fields	• Unique Transaction ID • Time Stamp • KYC Username
Response Fields	• Corresponding Transaction ID • Response Time Stamp • Status • Unique Nonce • CSR • Key ID • Authenticated SAM Public Key / Encryption Certificate • Error (if any)

5.5. SAM: Enrol Certificate (CHI protocol)

Protocol	Custom Protocol
Request Fields	• Transaction ID correlating to Initialization • Time Stamp • KYC Username • AuthPIN encrypted with authenticated SAM Public Key and bound to Transaction ID • Certificate Data • Key ID
Response Fields	• Corresponding Transaction ID • Response Time Stamp • Status • KYC Username • Certificate ID • Error (if any)

5.6. SAM: Initiate Signing (SHI protocol)

Protocol	Custom Protocol
Request Fields	• Unique Transaction ID • Time Stamp • KYC Username • Certificate ID • Document Hash(es) and count (maximum 10)
Response Fields	• Corresponding Transaction ID • Response Time Stamp • Status • Unique Nonce • Authenticated SAM Public Key / Encryption Certificate • Error (if any)

5.7. SAM: Authenticate user (AI protocol)

Protocol	Custom Protocol
Request Fields	• Transaction ID correlating to Initialization • Time Stamp • KYC Username

	• Certificate ID • AuthPIN encrypted with authenticated SAM Public Key and bound to Transaction ID
Response Fields	• Corresponding Transaction ID • Response Time Stamp • Status • KYC Username • Certificate ID • Single-use, time-bound Session Token bound to Transaction ID and authorised Document Hash(es) and count • Error (if any)

5.8. SAM: Perform Signing (SHI protocol)

Protocol	Custom Protocol
Request Fields	• Unique Transaction ID • Time Stamp • KYC Username • Certificate ID • Session Token (to be validated by SAM within SAM/HSM-protected boundary) • Document Hash(es) exactly matching authorised list and count (maximum 10)
Response Fields	• Corresponding Transaction ID • Response Time Stamp • Status • KYC Username • Signature(s) • User Certificate(s) • Error (if any)

6. Change History

Date Version Sections Description			
09.09.2026	1.0	2.3, 3.1-3.2, 4.2, 5	Aligned with e-Authentication Guidelines v1.9 security requirements.