ASTES

Advances in Science, Technology and Engineering Systems Journal
Vol. 4, No. 4, 347-358 (2019)
www.astesj.com
Special Issue on Advancement in Engineering and Computer Science

ASTES Journal
ISSN: 2415-6698

Using Privacy Enhancing and Fine-Grained Access Controlled eKYC to implement Privacy Aware eSign

Puneet Bakshi*, Sukumar Nandi

Department of Computer Science and Engineering, Indian Institute of Technology, Guwahati, 781039, India

ARTICLE INFO

Article history:
Received: 28 May, 2019
Accepted: 23 July, 2019
Online: 19 August, 2019

Keywords:
eSign
eKYC
Electronic Signature
Privacy
Access Control
Authentication
Token
BAN logic

ABSTRACT

eSign is an online electronic signature service which is recently gaining more prominence in India. eSign is based on two online services from UIDAI, viz. a viz., Aadhaar based authentication and retrieval of resident's eKYC information after taking his/her consent. With increased adoption of Aadhaar based services, privacy of user data has become more and more important. Present method of taking boolean consent from resident through non-UIDAI entity may not be acceptable for two main reasons, first is that the consent does not include in itself a proof from resident that the consent is indeed taken from him/her and second is that the resident may wish to have better privacy and fine grained access control rules to access his/her eKYC data. Bakshi et.el have introduced a mechanism to improve amortized performance of eSign using a digital access token. In this work, the digital access token is enhanced to include Privacy Enhancing and Fine-Grained Access Control (PEaFGAC) Statements for facilitating Privacy Aware eSign. These tokens can be used by other entities to access eKYC data of the resident with better access controls enforced by the resident. This paper briefly describes the present model of eSign, the earlier proposed model of eSign followed by the proposed model of Privacy Aware eSign. The proposed model of Privacy Aware eSign is also analyzed using BAN logic assuming Dolev-Yao security environment.

1 Introduction

eSign is an online electronic signature service in India which is being promoted by Government of India as part of its Digital India Initiative. As opposed to traditional dongle based electronic signature, *eSign* provides benefits such as less cost, no manual authentication, no requirement of special hardware device and no requirement for the end user to keep any key secret. With the passage of *Information Technology Act (ITA-2000)*, an electronically signed digital document is considered equivalent to a handwritten signed paper document. In India, *eSign* is being regulated by *Controller of Certifying Authority (CCA)* and is being operated by certain designated empaneled agencies known as *eSign Service Providers (ESP)*. *ESP* provides its services to application specific agencies known as *Application Service Providers (ASP)*. *ASP* provides *eSign* service to the end users. *eSign* is governed by *Public Key Infrastructure (PKI)* which is further governed in legal matters by the national legislature of the country.

To avail *eSign* service, a resident needs to enroll with *Unique Identification Authority of India (UIDAI)* and receive a 12-digit identity number called *Aadhaar* [1] [2]. As part of the enrollment process, resident needs to provide information about his/her identity and address to *UIDAI* such as *Name*, *Date of Birth*, *Address*, *Phone number*, *Email-id*, *Biometric (fingerprint-scan, iris-scan)* etc. The process of obtaining this information from the end user is known as *Know Your Customer (KYC)* and is initially introduced by *Reserve Bank of India (RBI)* for financial banks [3]. Traditionally, this process involves submission of a self-attested physical form along with necessary physical documents, followed by verification and approval by receiving organization. eKYC is an online service which facilitates completion of KYC process electronically. eKYC has some significant benefits over traditional KYC, eKYC eliminates submission of physical documents by customer, is faster and is less error prone. *UIDAI*'s eKYC service facilitates a third entity to retrieve resident's identity, address and other details after taking explicit consent and authorization from the resident.

With increased adoption of *Aadhaar* based identification,

*Corresponding Author: Puneet Bakshi, IIT Guwahati, b.puneet@iitg.ac.in

many online services are now using *Aadhaar* based services and with its such wide adoption, privacy of user data has become even more important. Although *Aadhaar* based eKYC service provides access to eKYC data only after taking an explicit consent from the resident, this way of taking consent from resident has two shortcomings. First is that the consent is taken by *non-UIDAI* entity and does not encode in itself a proof from resident that the consent is indeed given by the resident. Second is that providing a boolean consent is too broad, either an unconditional access is given to the whole eKYC information or no access is given at all. A resident may wish to have a better privacy enhancing fine-grained access control to his/her eKYC data. Resident may wish to define a privacy and access control policy dictating the *scope* of information which can be provided, the *purpose* for which the information can be provided and *recipients* to whom the information can be provided. For example, a resident may wish to disclose only his/her name and address, only for electronic signature purpose and only to a specific *eSign* Service Provider.

In [4], the author explained two limitations of present *eSign* model. The first limitation is that the eKYC data access reflects a restrictive *self-only*, *full-resource* and *unlimited* access control. Author pointed out that a resident may wish to have a better access control mechanism which allows third entities to access part of a resource which is to be used for a specific purpose and for a limited time period. The second limitation is that for each *eSign* request, resident has to authenticate itself each time and to include the authentication proof in each such request. Author pointed out that if a resident needs to *eSign* multiple times, time taken by initial authentication phase will be a major bottleneck. The author proposed that amortized performance of eSign can be improved using *digital access token* which encodes in itself the authentication proof and other information such as how many *eSign* requests can be made using this token and the expiry time of the token.

This paper is an extension of [4] and the digital access token is enhanced to implement *Privacy Aware eSign*. Our main contribution in this paper is to introduce a method to implement *Privacy Aware eSign* using *Privacy Enhancing and Fine-Grained Access Control (PEaFGAC) Statements*. A resident can encode these statements in digital access token for better access to his/her eKYC data. This token can be provided to third entities so that they can present this token for claiming protected resource from *UIDAI*. This paper also presents security analysis of the proposed scheme using *Burrows-Abadi-Needham (BAN)* logic. The analysis shows that in the proposed scheme, even if the network is unreliable, the exchanged information is reliable and is secured against eavesdropping.

The remainder of this paper is organized as follows. Section 2 presents related work. Notations used in the paper are reported in figure 1. Section 3 presents Aadhaar based eKYC service. Section 4 presents *eSign* version 2.0 model. Section 5 presents *eSign* model proposed in [4] to improve amortized performance of *eSign*. Section 6 presents proposed *Privacy Aware eSign* model using privacy enhancing and fine-grained access controlled eKYC. Section 7 presents formal security analysis of the proposed model using *BAN logic* and finally section 8 concludes the paper.

$\{X\}_Y$	X is signed by key Y
SK_Y	Symmetric key of entity Y
SK_{Y_Z}	Symmetric key shared by entities Y and Z.
PR_Y	Private asymetric key of entity Y
PB_Y	Public asymetric key of entity Y
R_i	Resident
ASP_i	Application Service Provider
ESP_i	eSign Service Provider
$UIDAI$	Unique Identification Authority of India
ID_{R_i}, ID_{ASP_i} ID_{ESP_i}	Identities of R_i, ASP_i and ESP_i
TID_{ESP_i}, TID_{ASP_i}	Unique transaction identifiers generated by ESP_i and ASP_i
PW_{R_i}	Password of R_i for login to ASP_i portal
$AadhaarNo_{R_i}$	Aadhaar No of R_i
C_{R_i}	Cookie associated with R_i's logged-in session, assigned by ASP
PR_{B_i}, PR_{ASP_i} PR_{ESP_i}, PR_{UIDAI}	Private keys of R_i browser, ASP_i, ESP_i and $UIDAI$
PB_{B_i}, PB_{ASP_i} PB_{ESP_i}, PB_{UIDAI}	Public keys of R_i browser, ASP_i, ESP_i and $UIDAI$
$n*_!$	nonces such as $n1_{ASP_i}$, where $*$ is any integer and ! can be R_i, ASP_i, ESP_i or $UIDAI$
$DataR_i(DataA_i,$ $DataE_i, DataU_i)$	Intermediate data in plaintext to be send by R_i (ASP_i, ESP_i, $UIDAI$)
$SignR_i(SignA_i,$ $SignE_i, SignU_i)$	$\{H(DataR_i)\}_{PR_{Ri}}$
$consent_{use\_ekyc}$	Consent from resident whether his/her eKYC can be used
$consent_{genuse\_at}$	Consent from resident whether a digital access token can be generated for later use
$License_{ASP_i}$ $License_{ESP_i}$	License for ASP_i (ESP_i) to use services from ESP (UIDAI)
M_i	Message (in plaintext) to be eSign
DSC_{Ri_Mi}	Digital Signature Certificate generated for message M_i for resident R_i
$\{M\}_{eSign\_R_i\_ESP_i}$	eSigned message (by R_i) through ESP_i
$H()$	One way cryptographically secure hash fn
$\|\|$	Concatenation operator
$\oplus$	XOR operator

Figure 1: Notations used in this paper

2 Related Work

Digital tokens are increasingly being used in many cryptography related applications to achieve varied objectives.

U-Prove [5] is an identity management solution based on blind signatures [6] which uses digital tokens to achieve objectives of privacy and anonymity. *U-Prove* consists of two protocols, viz., issuance protocol and presentation protocol. In issuance protocol, identity provider issues digital token to the subscriber which (s)he can later present to the verifier in presentation protocol so that the service provider can grant resource access to the subscriber. A *U-Prove* token consists of a unique token identifier, a public key of the token which aggregates information in the token, a token information field which encodes token specific information, a prover information field which is opaque to the issuer, issuer signature on all the other token contents and a boolean value which indicates whether the token is protected by a device. *U-Prove* uses digital tokens effectively by encoding necessary information in it in cryptographically secure way to achieve objectives such as privacy and anonymity.

OAuth2 [7] is an authorization framework which allows delegation of access to protected resources to a third party by using digital tokens referred to as access tokens. Access tokens are issued to Clients by *Authorization Server* after taking permission from *Resource Owner*. An access token can be of two types, viz., a bearer token and a MAC token. A bearer token is an opaque string which can be used to claim access to a resource by any entity who presents the

token. A MAC token is essentially a shared symmetric key which is used to sign a challenge by the client to prove its possession of the token to authorization server. *OAuth2* uses digital tokens effectively for access delegation and is used by many organizations for data sharing.

Bitcoin [8] is a decentralized digital currency which can be transacted over peer-to-peer bitcoin network. A bitcoin network is composed of cryptographically secure linear chains of blocks with each block containing a header and a collection of transactions. A transaction is essentially a digital token that changes ownership of bitcoins from one entity to another. Each transaction in bitcoin network is broadly composed of three parts, viz., input, output and amount. Input refers to the previous owner of the bitcoins, output refers to the new owner of the bitcoins and amount refers to the amount of bitcoin that is transacted. Bitcoins uses cryptographically secure digital information containers (similar to digital tokens) effectively for the realization of digital currency.

Although Attribute Based Encryption (ABE) is also evolved to protect privacy of user data, it is based on Identity Based Encryption (IBE). An agency may not shift from PKI to IBE framework for a number of reasons.

3 Aadhaar based e-KYC service (v2.1)

Aadhaar based eKYC service is available to general citizens only through certain empaneled agencies such as *eSign Service Provider (ESP)* and the infrastructure network is secured by certain designated agencies known as *Authentication Service Agency (ASA)* and *KYC User Agency (KUA)*. eKYC service is hosted as a stateless REST based web service over HTTPS and the details are sent as input data encoded in XML. Figure 2 depicts Aadhaar's eKYC webservice as specified in eKYC v2.1 specification [9]. The specification provides following information about element *rc* which represents the resident consent.

"rc – (mandatory) Represents resident's explicit consent for accessing the resident's identity and address data from Aadhaar system. Only valid value is "Y". Without explicit consent of the Aadhaar holder application should not call this API [9]."

As can be seen from the specification, *rc* is a boolean consent and assumes that it has been transferred from resident to *UIDAI* unaltered. Although intermediate communication channels between various entities from resident to *UIDAI* are well secured and access to eKYC data is provided only after receiving explicit consent from resident, this way of taking consent from resident has two shortcomings. First is that the consent is taken by a *non-UIDAI* entity and does not encode in itself a proof from resident that it is (s)he who provided the consent. This is because residents do not have any registered tamper proof crypto device which can be used to encrypt user consent using resident specific PIN or password. Second is that providing a boolean consent is too broad, either an unconditional access is given to the whole eKYC information or no access is given at all. A resident may wish to have a better privacy enhancing fine-grained access control to his/her eKYC data indicating details on *scope*, *purpose* and *recipient*.

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

Input Data:
  <Kyc ver="" ra="" rc="" lr="" de="" pfr="">
       <Rad>base64 encoded fully valid Auth XML for
           resident
       </Rad>
  </Kyc>

Response Data:
  <Resp status="" ko="" ret="" code="" txn="" ts=""
        err=""> encrypted and base64 encoded KycRes
        element
  </Resp>

  <KycRes ret="" code="" txn="" ts="" ttl="" actn=""
      err="">
      <Rar>base64 encoded fully valid Auth response
          XML for resident
      </Rar>
      <UidData uid="">
          <Poi name="" dob="" gender="" />
          <Poa co="" house="" street="" lm="" loc=""
               vtc="" subdist="" dist="" state=""
               country="" pc="" po=""/>
          <LData lang="" name="" co="" house=""
               street="" lm="" loc="" vtc=""
               subdist="" dist="" state=""
               country="" pc="" po=""/>
          <Pht> base64 encoded JPEG photo of the
               resident
          </Pht>
          <Prn type="pdf"> base64 encoded signed
               Aadhaar letter for printing
          </Prn>
      </UidData>
      <Signature/>
  </KycRes>
```

Figure 2: Aadhaar's eKYC 2.1 API

4 Present model of eSign in India

In *eSign* version 2.0 [10], a resident first registers itself with a front end application specific agency viz. a viz., *Application Service Provider (ASP)*. A resident can use either OTP based authentication or biometric based authentication. In case of OTP based authentication, OTP generation request is forwarded to *UIDAI* via *ASP* and *ESP*. *UIDAI* generates an OTP and sends it to resident's registered mobile number. In case of biometric based authentication, resident gets his fingerprint/iris scanned through a registered device. After authentication phase, resident now initiates an eSign request through *ASP* by providing it the consent to use his/her eKYC, the document to be signed and his/her *Aadhaar* number. *ASP* calculates cryptographic hash of the document and sends it along with the resident's consent and resident's *Aadhaar* number to *ESP*. *ESP* takes authentication proof from resident, creates a random symmetric key SK_{ESP_UIDAI} and a

Personal Identity Data Object (*PID*). *PID* encodes in itself the resident's authentication proof and the cryptographic hash of the *PID* object ($SHA256(PID)$). *ESP* first encrypts *PID* with SK_{ESP_UIDAI}, second encrypts cryptographic hash of *PID* ($SHA256(PID)$) with SK_{ESP_UIDAI} and third encrypts SK_{ESP_UIDAI} with public key of *UIDAI* (PB_{UIDAI}). *ESP* now wraps them in a new object called "Auth" and sends it to *UIDAI* in eKYC request. *UIDAI* provides eKYC information to *ESP*. Using received eKYC information, *ESP* generates a Digital Signature Certificate (DSC) and provides it to *ASP*. *ASP* provides the digitally signed document to the resident.

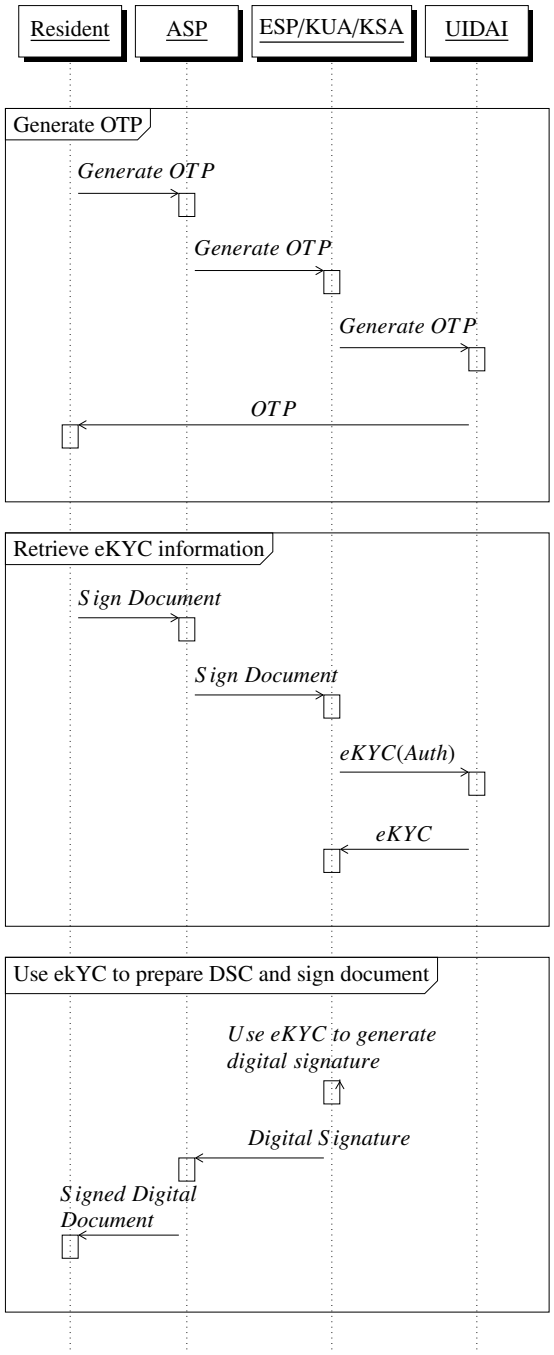

Figure 3: Sequence diagram of eSign 2.0

In practice, the initial authentication phase in *eSign* request is most time consuming since it involves either the manual text input (in case of OTP based authentication) or the physical scan of the fingerprint/iris (in case of biometric based authentication). Other than that, in some use cases such as *Create Birth Certificate*, *Create Death Certificate*, *Student Enrollment*, etc., the application is most heavily used during a certain time period (nearing the end of a deadline) which puts a sudden nationwide load on *UIDAI* services.

5 eSign model as proposed in [4]

In [4], author explained two limitations of present *eSign* model and proposed a mechanism to address the same. The first limitation is that in present model of *eSign*, eKYC data access reflects a restrictive *self-only*, *full-resource* and *unlimited* access control. Author pointed out that the resident may wish to have a better access control mechanism reflecting *third-entity-also*, *partial resource*, *use-limited* and *time-limited*.

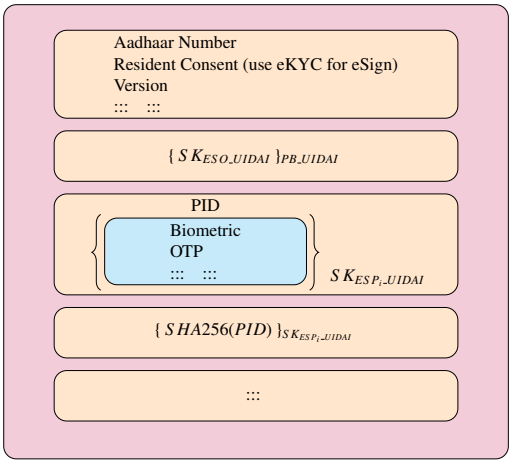

Figure 4: Auth Object (eSign 2.0)

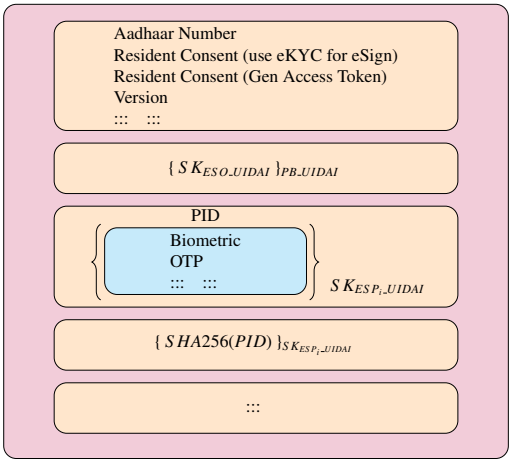

Figure 5: Auth Object as proposed in [4]

The second limitation is that a resident has to authenticate himself/herself for each *eSign* request and include the corresponding authentication proof in each *eSign* request.

If a resident wishes to *eSign* a large number of documents, the initial authentication phase will comprise most of the overall *eSign* time. After taking necessary consent from the resident, his/her authentication proof be stored with *ESP* in first request and is reused in rest of the requests.

A digital access token [figure 6] can be used to include claims from participating entities (*ESP* and *UIDAI*). A new service named *GenerateAccessToken* is proposed to be introduced by *UIDAI*.

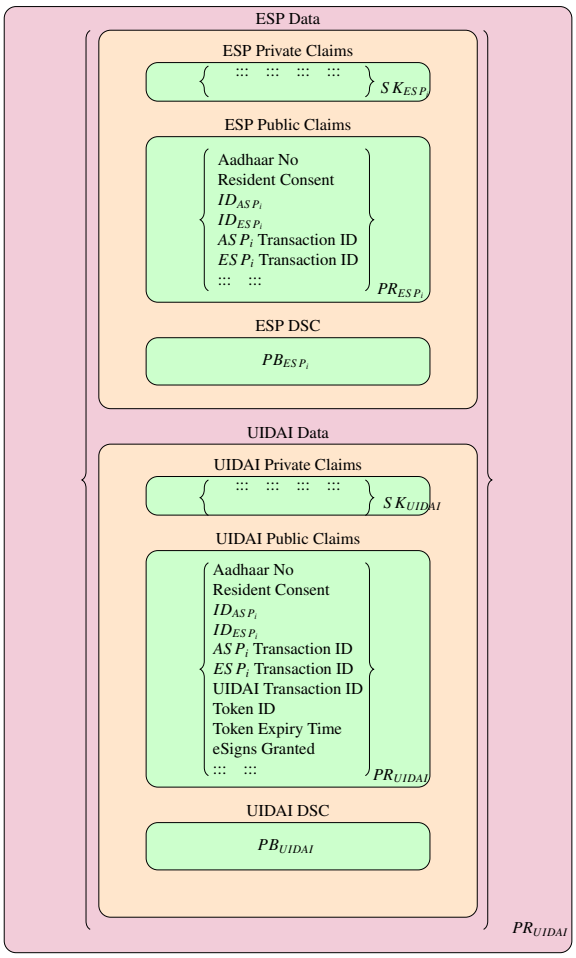

Figure 6: Access Token Structure as proposed in [4]

In this proposed model of eSign [figures 7, 8], resident first authenticates himself/herself using OTP or biometric based authentication and sends *eSign* request to *ASP*. *ASP* forwards this *eSign* request to *ESP*. *ESP* takes OTP and permission to generate access token from resident and creates an "Auth" object. This "Auth" object is created as before but additionally including *ESP* claims as well. *ESP* sends *GenerateAccessToken* request to *UIDAI* including "Auth" object. After receiving this request, *UIDAI* creates an access token including its own claims as well as claims received from *ESP*. *UIDAI* sends this access token back to the *ESP*. Now, *ESP* sends eKYC request to *UIDAI* including this access token instead of the "Auth" object. After receiving eKYC information from *UIDAI*, *ESP* generates *Digital Signature Certificate (DSC)* from it and provides the same to *ASP*. *ASP* embeds DSC in the document and sends the digitally signed document to the resident. For all rest of the *eSign* requests, *ESP* can reuse the same access token in eKYC requests and avoid the initial authentication phase.

The paper also presented two usability scenarios, based on whether the eKYC information can be cached by *ESP* or not. If *ESP* is permitted to reliably and securely store eKYC information of the resident, even the repeated eKYC requests from *ESP* to *UIDAI* can be avoided.

The paper also presented performance comparison analysis of proposed model with present model and found substantial improvement in amortized performance of *eSign*.

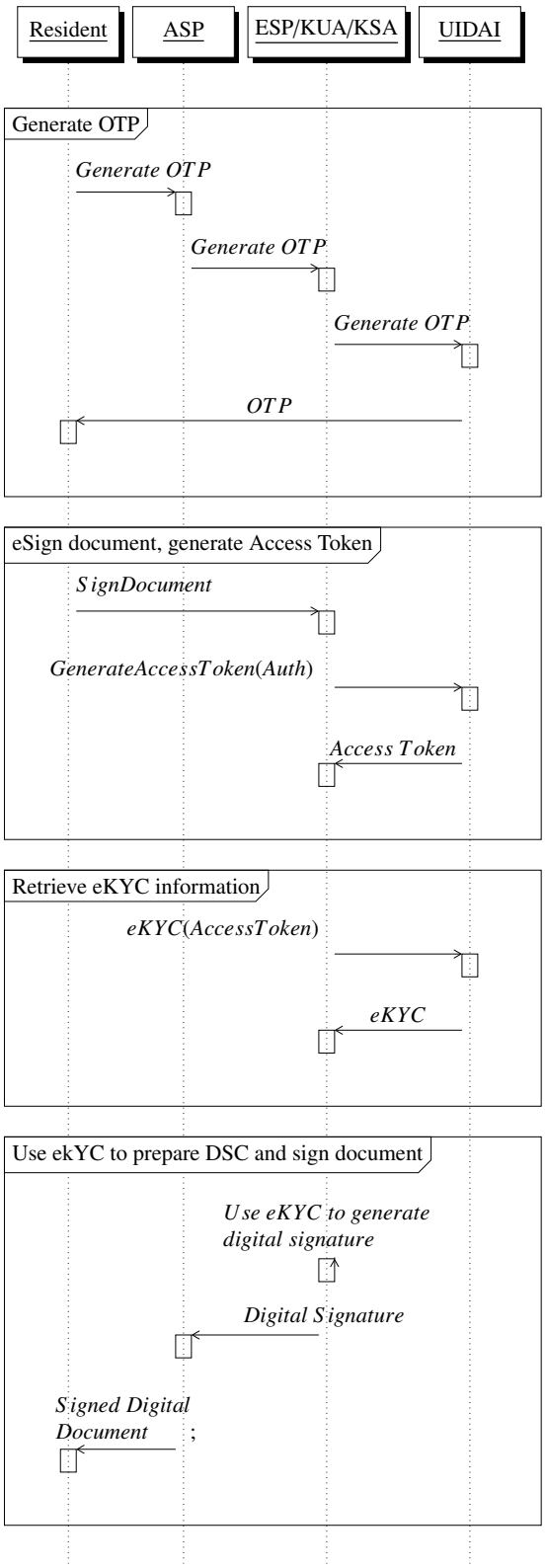

Figure 7: First call to eSign in eSign model proposed in [4]

6 Privacy Aware eSign model

In earlier proposed model [section 5] digital access token is used to increase amortized performance of *eSign* by storing necessary claims from *UIDAI* and *ESP*. The same token can be enhanced to include claims from resident as well. A resident can encode claims related to privacy and fine grained access control of his/her eKYC data. A stricter *PEaFGAC* statements may be enforced centrally at *UIDAI* level and an

overriding less strict rule can be supplied with each eKYC request to grant access to the requesting entity.

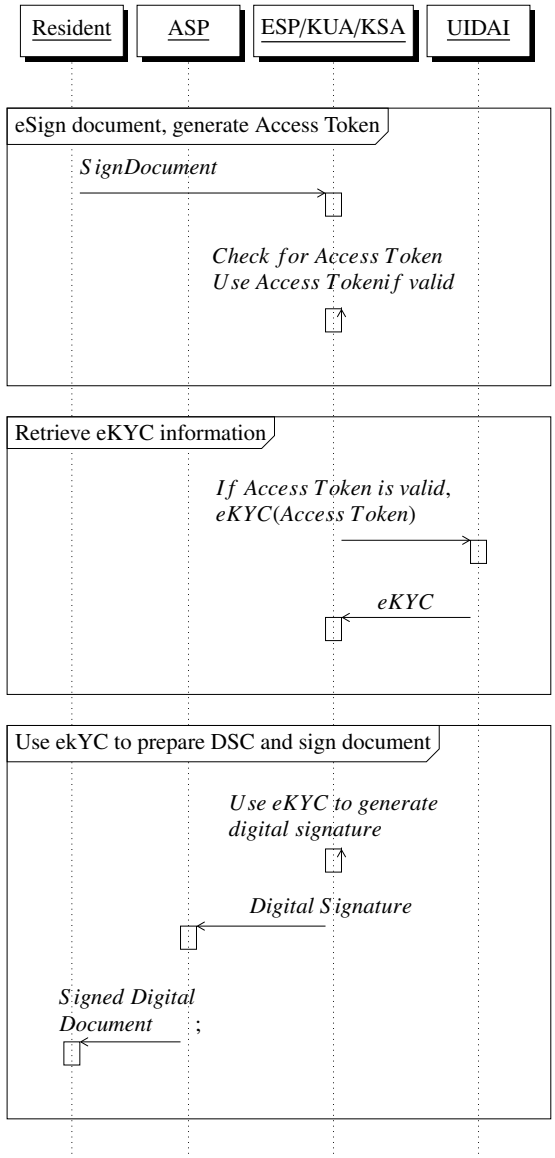

Figure 8: Second call to eSign in case eKYC needs to be fetched again

6.1 Privacy and Fine-Grained Access Control (PEaFGAC) Statements for eKYC data

A *PEaFGAC* statement encodes in itself the *scope* of information which can be provided, the *purpose* for which the information can be provided and attributes of *recipients* to whom the information can be provided. These statements are comprised of small sub-statements which are combined using relational operators. Each statement is identified by a numeric *id* and an alphanumeric *tag*.

An example of a *PEaFGAC* statement is presented in figure 10. This statement encodes in it that the purpose for seeking eKYC information should be *eSign*, seeking entity must either have the email in domain *finance.iitg.ac.in*, or must be working in *finance* department of *Indian Institute of Technology, Guwahati (IITG)*, or must have a designation of *director* or above. The statement is uniquely identified by a statement identifier (*ID*) and has a small alphanumeric representational string (*TAG*). Other than these, the staement also encodes in it the purpose for which eKYC can be accessed (*Purpose*), required (*eKYC*) attributes of information seeker (*AP*) and eKYC information which can be provided to the requester (*scope*). If required, a user can have multiple privacy statements for his/her eKYC data represented by different tags.

```
AadhaarNo
POI:
  name, dob
POA:
  co, house, street, lm, loc, vtc, subdist, dist,
  state, country, pc, po
LData:
  lang, name, co, house, street, lm, loc, vtc,
  subdist, dist, state, country, pc, po
Pht:
  <Base64 encoded JPEG photo of resident
Org:
  dep, desig
Other:
  email
```

Figure 9: Least information assumed to be available from eKYC for this paper

```
PS.ID:
  5
PS.Tag:
  eSignDoc
PS.Purpose:
  eSign
PS.AP:
  (email = *@finance.iitg.ac.in)       OR
  (org = IITG AND org.dep = finance)   OR
  (desig >= director)
PS.Scope:
  poi.name, poi.dob, poa.country, Ldata.lang
```

Figure 10: Example of a PEaFGAC statement

It is assumed that all entities which request eKYC data also have their eKYC information available with *UIDAI*. This include not just the users but the organizations such as *ESP*s as well. To better know an entity (both users and organizations), it is proposed that eKYC fields are expanded to include more details such as entity type (indicating whether the subject is a human or an organization), resident's organization, resident's department, resident's designation, etc. When an entity attempts to access eKYC data of a resident, entity's eKYC data and purpose for which the eKYC data is sought are verified against *PEaFGAC* statement protecting eKYC data to decide whether the requisite access can be granted or not. Only if the access can be granted, will the eKYC data be provided to the requesting entity. The eKYC data provided to the entity is limited in scope by *PEaFGAC* statement. For rest of this paper, eKYC data is assumed to consists of at least the information presented in figure 9.

6.2 PEaFGAC Token

The token structure introduced in [4] can be enhanced to include resident claims including *PEaFGAC* statement [fig-

ure 11]. Before sending an *eSign* request, resident creates a *PEaFGAC* token by sending a token generation request to *UIDAI* through *ASP* and *ESP*. During token generation process, resident is redirected to *UIDAI* web page where (s)he provides OTP value for authentication and *PEaFGAC* statement for privacy and fine-grained access to his/her eKYC data. Subsequently *UIDAI* verifies the OTP value and signs cryptographic hash of the statement with its private key and stores the signed hash in resident's private claims and stores the statement in plaintext in resident's public claims section of *PEaFGAC* digital access token.

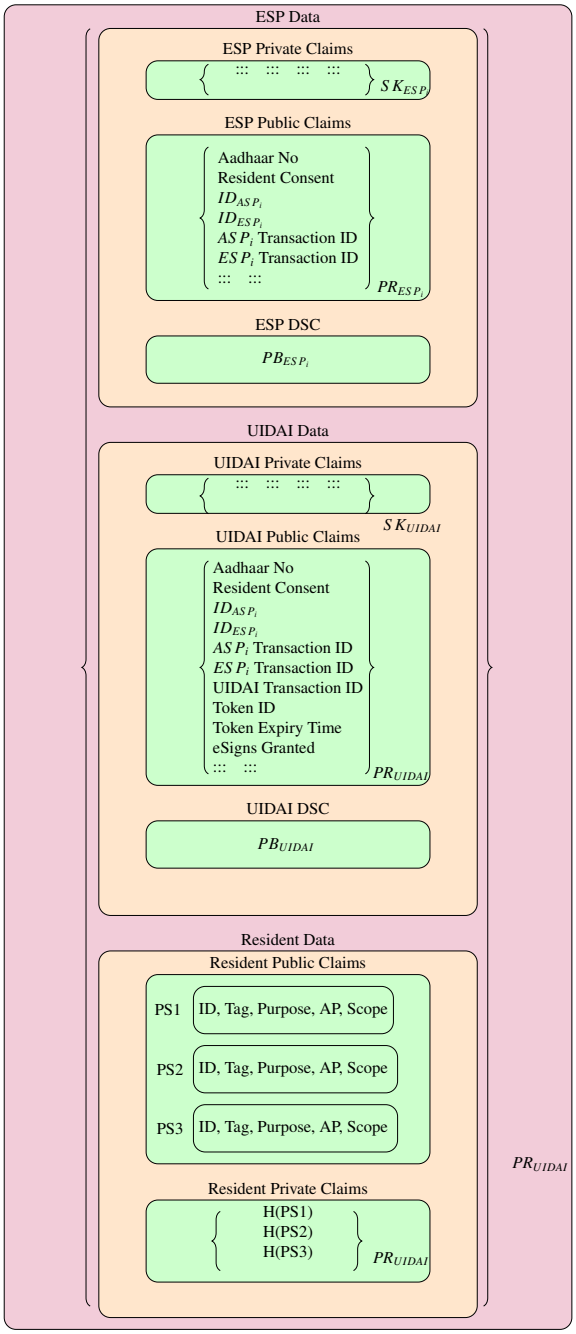

Figure 11: Proposed PEaFGAC Token Structure

Figure 12 depicts the sequence and details of communication messages among participating entities for generation of a token. First column indicates the message identifier, second column indicates the participating entities and the direction of communication and third column indicates what message is sent and how it is constructed.

Login to ASP		
$TGM1$	$R \rightarrow ASP$	$Generate\ nonce\ n1_{R_i}$ $DataR_1 = ID_{R_i} \| PW_{R_i} \| PB_{B_i} \| n1_{R_i}$ $SignR_1 = \{H(DataR_1)\}_{PR_{B_i}}$ $\{\ loginReq(DataR_1, SignR_1)\ \}_{PB_{ASP_i}}$
$TGM2$	$R \leftarrow ASP$	$DataA_1 = C_{R_i} \oplus (n1_{R_i} + 1)$ $SignA_1 = \{H(DataA_1)\}_{PR_{ASP}}$ $\{\ loginRes(DataA_1, SignA_1)\}_{PB_{B_i}}$
Generate OTP		
$TGM3$	$R \rightarrow ASP$	$Generate\ nonce\ n2_{R_i}$ $DataR_2 = AadhaarNo_{R_i} \| C_{R_i} \| n2_{R_i}$ $SignR_2 = \{H(DataR_2)\}_{PR_{B_i}}$ $\{\ getotpASPReq(DataR_2, SignR_2)\ \}_{PB_{ASP_i}}$
$TGM4$	$ASP \rightarrow ESP$	$Generate\ nonce\ n1_{ASP_i}$ $DataA_1 = AadhaarNo_{R_i} \| ID_{ASP_i} \|$ $License_{ASP_i} \| TID_{ASP_i} \| n1_{ASP_i}$ $SignA_1 = \{H(DataA_1)\}_{PR_{ASP_i}}$ $\{\ getotpESPReq(DataA_1, SignA_1)\ \}_{PB_{ESP_i}}$
$TGM5$	$ESP \leftarrow UIDAI$	$Generate\ nonce\ n1_{ESP_i}$ $DataE_1 = AadhaarNo_{R_i} \| ID_{ESP_i} \|$ $License_{ESP_i} \| TID_{ESP_i} \| n1_{ESP_i}$ $SignE_1 = \{H(DataE_1)\}_{PR_{ESP_i}}$ $\{\ getotpReq(DataE_1, SignE_1)\ \}_{PB_{UIDAI}}$
$TGM6$	$R \leftarrow UIDAI$	$\{\ OTP\ \}_{SecureCellularNetwork}$
$TGM7$	$ESP \leftarrow UIDAI$	$DataU_1 = returnStatus \| TID_{ESP_i} \|$ $MaskedMobileNo \| (n1_{ESP_i} + 1)$ $SignU_1 = \{H(DataU_1)\}_{PR_{UIDAI}}$ $\{\ getotpRes(DataU_1, SignU_1)\}_{PB_{ESP_i}}$
$TGM8$	$ESP \leftarrow ASP$	$DataE_2 = returnStatus \| TID_{ASP_i} \|$ $MaskedMobileNo \| (n1_{ASP_i} + 1)$ $SignE_2 = \{H(DataE_2)\}_{PR_{ESP_i}}$ $\{\ getotpESPRes(DataE_2, SignE_2)\}_{PB_{ASP_i}}$
$TGM9$	$ESP \leftarrow R$	$DataA_2 = returnStatus \|$ $MaskedMobileNo \| (n2_{R_i} + 1)$ $SignA_2 = \{H(DataA_2)\}_{PR_{ASP_i}}$ $\{\ getotpASPRes(DataA_2, SignA_2)\}_{PB_{B_i}}$
Generate Token		
$TGM10$	$R \rightarrow ASP$	$Generate\ nonce\ n3_{R_i}$ $DataR_3 = C_{R_i} \| n3_{R_i}$ $SignR_3 = \{H(DataR_3)\}_{PR_{B_i}}$ $\{\ gentokASPReq(DataR_3, SignR_3)\ \}_{PB_{ASP_i}}$
$TGM11$	$ASP \rightarrow ESP$	$Generate\ nonce\ n2_{ASP_i}$ $DataA_3 = AadhaarNo_{R_i} \| ID_{ASP_i} \| License_{ASP_i} \|$ $TID_{ASP_i} \| n2_{ASP_i}$ $SignA_3 = \{H(DataA_3)\}_{PR_{ASP_i}}$ $\{\ gentokESPReq(DataA_3, SignA_3)\ \}_{PB_{ESP_i}}$
$TGM12$	$ESP \rightarrow UIDAI$	$Generate\ nonce\ n3_{ESP_i}$ $DataE_3 = AadhaarNo_{R_i} \| ID_{ESP_i} \|$ $License_{ESP_i} \| TID_{ESP_i} \|$ $n3_{ESP_i}$ $SignE_3 = \{H(DataE_3)\}_{PR_{ESP_i}}$ $\{\ gentokUIDAIReq(DataE_3, SignE_3)\ \}_{PB_{UIDAI}}$
$TGM13$	$UIDAI \leftarrow ESP$	$Generate\ nonce\ n1_{UIDAI}$ $DataU_1 = UIDAIRedirectURL$ $(ForTakingPrivacyStatements) \|\|$ $PB_{UIDAI} \| TID_{ESP_i} \| n1_{UIDAI}$ $SignU_1 = \{H(DataU_1)\}_{PR_{UIDAI}}$ $\{\ genpsUIDAIReq(DataU_1, SignU_1)\ \}_{PB_{ESP_i}}$
$TGM14$	$ESP \leftarrow ASP$	$Generate\ nonce\ n4_{ESP_i}$ $DataE_4 = UIDAIRedirectURL$ $(ForTakingPrivacyStatements)$ $\| PB_{UIDAI} \| TID_{ASP_i} \| n4_{ESP_i}$ $SignE_4 = \{H(DataE_3)\}_{PR_{ESP_i}}$ $\{\ genpsESPReq(DataE_4, SignE_4)\ \}_{PB_{ASP_i}}$
$TGM15$	$ASP \leftarrow R$	$Generate\ nonce\ n4_{R_i}$ $DataA_4 = \{\ Present\ UIDAIRedirectURL\ to$ $Resident\ which\ requests\ him$ $to\ provide\ OTP\ Value\ and\ Privacy$ $statements \| PB_{UIDAI} \| n4_{R_i}\}$ $SignA_4 = \{H(DataA_4)\}_{PR_{ASP_i}}$ $\{\ genpsASPReq(DataA_4, SignA_4)\ \}_{PB_{B_i}}$
$TGM16$	$R \rightarrow UIDAI$	$\{\ PEaFGACPrivacyStatements\ \}_{PB_{UIDAI}}$
PEaFGAC Token Generation Request		
$TGM17$	$R \rightarrow ASP$	$DataR_4 = C_{R_i} \| (n4_{R_i} + 1)$ $SignR_4 = \{H(DataA_4)\}_{PR_{B_i}}$ $\{\ genpsASPRes(DataA_4, SignA_4)\ \}_{PB_{B_i}}$
$TGM18$	$ASP \leftarrow ESP$	$DataA_5 = (n4_{ESP_i} + 1)$ $SignA_5 = \{H(DataA_4)\}_{PR_{ASP_i}}$ $\{\ genpsESPRes(DataA_4, SignA_4)\ \}_{PB_{ESP_i}}$
$TGM19$	$ESP \leftarrow UIDAI$	$DataE_5 = (n1_{UIDAI} + 1)$ $SignE_5 = \{H(DataE_5)\}_{PR_{ESP_i}}$ $\{\ genpsUIDAIRes(DataE_5, SignE_5)\ \}_{PB_{UIDAI}}$
$TGM20$	$ESP \leftarrow UIDAI$	$Create\ PEaFGACTokenAT_{Ri}$ $DataU_2 = AT_{Ri} \| (n3_{ESP_i} + 1)$ $SignU_2 = \{H(DataE_5)\}_{PR_{ESP_i}}$ $\{\ gentokUIDAIRes(DataE_5, SignE_5)\ \}_{PB_{UIDAI}}$
$TGM21$	$ASP \leftarrow ESP$	$DataE_6 = (n2_{ASP_i} + 1)$ $SignE_6 = \{H(DataE_5)\}_{PR_{ESP_i}}$ $\{\ gentokESPRes(DataE_6, SignE_6)\ \}_{PB_{ASP_i}}$
$TGM22$	$R \leftarrow ASP$	$DataA_5 = (n3_{R_i} + 1)$ $SignA_5 = \{H(DataE_5)\}_{PR_{ASP_i}}$ $\{\ gentokESPRes(DataA_5, SignA_5)\ \}_{PB_{B_i}}$

Figure 12: Proposed PEaFGAC Token Generation protocol

6.3 Proposed Privacy Aware eSign model using PEaFGAC statement for eKYC data and PEaFGAC token

This section presents how *PEaFGC* statement for eKYC data and *PEaFGAC* token can be used to implement *Privacy Aware eSign*. It is assumed that *PEaFGAC* token has already been generated as explained in section 6.2. It is also assumed that the communication channel between resident and *ASP* is secured using SSL/TLS, between *ASP* and *ESP* is secured using SSL/TLS and between *ESP* and *UIDAI* is secured using dedicated secure leased lines.

Figure 13 depicts the sequence and details of communication messages transferred in *eSign* request in case eKYC needs to be fetched again.

Login to ASP		
$M1$	$R \rightarrow ASP$	$Generate\ nonce\ n1_{R_i}$ $DataR_1 = ID_{R_i} \| PW_{R_i} \| PB_{B_i} \| n1_{R_i}$ $SignR_1 = \{H(DataR_1)\}_{PR_{B_i}}$ $\{\ loginReq(DataR_1, SignR_1)\ \}_{PB_{ASP_i}}$
$M2$	$R \leftarrow ASP$	$DataA_1 = C_{R_i} \oplus (n1_{R_i} + 1)$ $SignA_1 = \{H(DataA_1)\}_{PR_{ASP}}$ $\{\ loginRes(DataA_1, SignA_1)\}_{PB_{B_i}}$
Initiate eSign request		
$M3$	$R \rightarrow ASP$	$Generate\ nonce\ n2_{R_i}$ $DataR_2 = M_i \| consent_{use\_ekyc} \|$ $consent_{genuse\_at} \| C_{R_i} \| n2_{R_i}$ $SignR_2 = \{H(DataR_2)\}_{PR_{B_i}}$ $\{\ signdocASPReq(DataR_2, SignR_2)\ \}_{PB_{ASP_i}}$
$M4$	$ASP \leftarrow ESP$	$Generate\ nonce\ n1_{ASP_i}$ $DataA_3 = H(M_i) \| AadhaarNo_{R_i} \| ID_{ASP_i} \|$ $License_{ASP_i} \| TID_{ASP_i} \|$ $consent_{use\_ekyc} \| consent_{genuse\_at} \|$ $n1_{ASP_i}$ $SignA_3 = \{H(DataA_3)\}_{PR_{ASP_i}}$ $\{\ signdocESPReq(DataA_3, SignA_3)\ \}_{PB_{ESP_i}}$
Retrieve eKYC (reusing access token) and sign document		
$M5$	$ESP \rightarrow UIDAI$	$Generate\ nonce\ n1_{ESP_i}$ $If\ AT_{R_i}\ is\ valid\ use\ it$ $DataE_5 = AT_{R_i} \| H(M_i) \| n1_{ESP_i}$ $SignE_5 = \{H(DataE_5)\}_{PR_{ESP_i}}$ $\{\ kycESPReq(DataE_5, SignE_5)\ \}_{PB_{UIDAI}}$
$M6$	$ESP \leftarrow UIDAI$	$Retrieve\ eKYC_{ESP_i}$ $Retrieve\ AT_{R_i} \rightarrow UC \rightarrow AP$ $Verify\ whether\ access\ can\ be\ granted$ $based\ on\ above\ two\ parameters.$ $eKYC_{R_i} = eKYC\ of\ resident\ scoped$ $by\ AT_{R_i} \rightarrow UC \rightarrow scope$ $DataU_3 = eKYC_{R_i} \| (n1_{ESP_i} + 1)$ $SignU_3 = \{H(DataU_3)\}_{PR_{UIDAI}}$ $\{\ kycESPRes(DataU_3, SignU_3)\ \}_{PB_{ESP_i}}$
$M7$	$ASP \leftarrow ESP$	$Generate\ key\ pair\ PB_{R_i}\ PR_{R_i}\ using\ eKYC_{R_i}$ $SignChain = \{\ PB_{R_i}\ \}_{PR_{ESP_i}} \|$ $\{\ PB_{ESP_i}\ \}_{PR_{CCA}}$ $DSC_{R_i\_M_i} = \{\ eKYC_{R_i} \| H(M_i)\ \}_{PR_{R_i}} \|$ $PB_{R_i} \| SignChain$ $Delete\ PR_{R_i}$ $DataU_2 = DSC_{R_i\_M_i} \| TID_{ASP_i} \| (n1_{ASP_i} + 1)$ $SignU_2 = \{H(DataU_2)\}_{PR_{ESP_i}}$ $\{\ signdocESPRes(DataU_2, SignU_2)\ \}_{PB_{ASP_i}}$
$M8$	$R \leftarrow ASP$	$\{M_i\}_{eSign\_R_i\_ESP_i} = M_i \| DSC_{R_i\_M_i}$ $DataA_6 = \{M_i\}_{eSign\_R_i\_ESP_i} \| (n2_{R_i} + 1)$ $SignA_6 = \{H(DataA_6)\}_{PR_{ASP_i}}$ $\{\ signdocASPRes(DataA_6, SignA_6)\ \}_{PB_{B_i}}$
$M9$	$R \leftrightarrow R$	$Verify\ correctness\ of\ eKYC,\ H(M)\ and$ $SignChain\ in\ \{M_i\}_{eSign}$

Figure 13: Proposed Privacy Aware eSign model

7 Formal Security Analysis of the proposed model using BAN Logic

This section presents formal security analysis of the proposed scheme using *Burrows-Abadi-Needham (BAN) logic* [11]. Because of space limitation, it is assumed that PEaFGAC token has already been generated securely.Analysis of the token generation request can be done similarly.*BAN logic* is a well-known model used to find beliefs of participants in a cryptographic protocol.

Operator Usage	Description
$P \models X$	P believes statement X
$P \lhd X$	P sees statement X
$P \mapsto X$	P controls X
$\#(X)$	Message X is fresh
$P \overset{K}{\leftrightarrow} Q$	P and Q share key K
$\overset{K}{\mapsto} P$	P has K as its public key
$P \overset{X}{\rightleftharpoons} Q$	Formula X is a secret known only to P and Q
$\{X\}_K$	Formula X is encrypted using K
$\langle X \rangle_Y$	Formula X is combined with formula Y

Figure 14: Fundamental BAN operators

Operator Usage	Description
$M_{eSign\_R\_ESP\_CCA}$	eSign of message M is done by Resident R through ESP approved by CCA $M_{eSign\_R\_ESP\_CCA}$ $= M \| DSC_{R\_M}$ $= M \| \{eKYC_R \| H(M)\}_{PR_R} \|$ $PB_R \| SignChain$ $= M \| \{eKYC_R \| H(M)\}_{PR_R} \|$ $PB_R \| \{PB_R\}_{PR_{ESP}} \|$ $\{PB_{ESP}\}_{PR_{CCA}}$
$P \models E_i \overset{secure}{\longrightarrow} E_j$	P believes that communication from entity E_i to E_j is secure
$P \models E_i \overset{secure}{\longleftarrow} E_j$	P believes that communication from entity E_j to E_i is secure
$P \models E_i \overset{secure}{\longleftrightarrow} E_j$	P believes that communication between entities E_i and E_j is secure in both directions
$P \models E_i \overset{ACTPerm}{\longleftrightarrow} E_j$	P believes that entity E_i has given permission for action ACT to entity E_j
$P \models C_R \rightsquigarrow E_i$	P believes that cookie C_R is associated with logged-in entity E_i
$E_R \models E_R \overset{C_{E_R} \rightarrow ID_{E_R}}{\longrightarrow} E_R$	E_R believes that it has securely communicated its identity ID_{E_R} to entity E_R through cookie C_{E_R}

Figure 15: Extended BAN operators

The security environment is assumed to be based on *Delev-Yao model* in which all messages are communicated over public channels and an attacker can see, modify, compose and replay messages but cannot break cryptographic principles. The security environment also assumes that an attacker can decipher messages if he has a valid decryption key. Some of the fundamental operators used in *BAN logic* are defined in figure 14. An extension to *BAN* logic, defined in figure 15 is required to analyse the proposed model.

Rules of Inference

[R1:] *Message meaning rules* concern the interpretation of messages. They all derive beliefs about the origin of messages.

For shared secrets, the inference rule is

$$\frac{P \models Q \overset{Y}{\rightleftharpoons} P,\ P \lhd \langle X \rangle_Y}{P \models Q \rhd X}$$

That is, if P believes that the secret Y is shared with Q and sees $\langle X \rangle_Y$, then P believes that Q once said X.

[R2:] The *nonce-verification* rule expresses the check that a message is recent, and hence, that the sender still believes in it:

$$\frac{P \models \#(X),\ P \models Q \rhd X}{P \models Q \models X}$$

That is, if P believes that X could have been uttered only recently and that Q once said X, then P believes that Q believes X.

[R3:] The *jurisdiction* rule states that if P believes that Q has jurisdiction over X, then P trusts Q on the truth of X:

$$\frac{P \models Q \Rightarrow X,\ P \models Q \models X}{P \models X}$$

[R4:] The *seeing* rule states that if a principal sees a formula, then he also sees its components, provided he knows the necessary keys:

$$\frac{P \lhd (X,Y)}{P \lhd X},\quad \frac{P \lhd \langle X \rangle_Y}{P \lhd X},\quad \frac{P \models Q \overset{K}{\leftrightarrow} P(.) P \lhd \{X\}_K}{P \lhd X},$$

$$\frac{P \models \overset{K}{\mapsto} P,\ P \lhd \{X\}_K}{P \lhd X},\quad \frac{P \models \overset{K}{\mapsto} P,\ P \lhd \{X\}_{K^{-1}}}{P \lhd X}.$$

Note that if P sees X and P sees Y it does NOT follow that P sees (X, Y) since that means that X and Y were uttered at the same time.

[R5:] The *fresh* rule states that if one part of the formula is fresh, then the entire formula must be fresh.

$$\frac{P \models \#(X)}{P \models \#(X, Y)}.$$

[R6:] The *belief* rule states that if P believes one part of the formula, then it also believe part of the formula.

$$\frac{P \models (X, Y)}{P \models (X)}.$$

Extended Rules of Inference

[R7:] If receiver entity E_R believes that C_R is a cookie associated with a unique session from resident R, PB_B is public key with browser used by resident R, PB_{E_R} is public key of receiver entity E_R, n_R is a fresh nonce generated by R, E_R receives message of the form $\{CommMsgReq(X||C_R||n_R, \{H(X||C_R||n_R)\}_{PR_{B_i}})\ \}_{PB_{E_R}}$, then E_R believes that X is sent by entity R and communication channel from R to E_R is secure and no message is observed, modified or replayed by an intruder.

$$\begin{array}{lll} E_R & \models & \overset{PB_{E_R}}{\longmapsto} E_R, \\ E_R & \models & C_R \rightsquigarrow S, \\ E_R & \models & \#n_R, \\ E_R & \models & \{\{Y\}_{PR_R}\}_{PB_R} = Y \\ E_R & \lhd & \{\ CommMsgReq\ (\\ & & \quad X||C_R||n_R, \\ & & \quad \{H(X||C_R||n_R)\}_{PR_{R_i}})\ \}_{PB_{E_R}} \\ \hline E_R & \models & R \xrightarrow{Secure} E_R, \\ E_R & \models & R \rhd X \end{array}$$

[R8:] If receiver entity E_R believes that PB_{E_R} is public key of receiver entity E_R, n_{E_R} is a fresh nonce generated by E_R, E_R receives message of the form $\{\ CommMsgReq\ (X||n_{E_R}, \{H(X||n_{E_R})\}_{PR_{E_R}})\ \}_{PB_{E_R}}$, then E_R believes that X is sent by entity E_R and communication channel from E_R to E_R is secure and no message is observed, modified or replayed by an intruder.

$$\begin{array}{lll} E_R & \models & \overset{PB_{E_R}}{\longmapsto} E_R, \\ E_R & \models & \{\{Y\}_{PR_{E_R}}\}_{PB_{E_R}} = Y, \\ E_R & \models & \#n_{E_R} \\ E_R & \lhd & \{\ CommMsgReq\ (\\ & & \quad X||n_{E_R}, \\ & & \quad \{H(X||n_{E_R})\}_{PR_{E_R}})\ \}_{PB_{E_R}} \\ \hline E_R & \models & E_R \xrightarrow{Secure} E_R \\ E_R & \models & E_R \rhd X \end{array}$$

[R9:] If receiver entity E_R believes that communication from all possible sender entities E_{R_i} to E_R $(\forall i = 1...n)$ is secure, then E_R believes that communication channel to E_R is secure and no message is observed, modified or replayed by an intruder.

[R10:] If resident R believes that C_R is a cookie associated with a unique session from resident R, PB_{E_R} is public key of entity E_R, n_R was a fresh nonce generated by R and used in a previous request call from R to E_R, R receives a message of the form $\{CommMsgRes(X||(n_R+1), \{H(X||(n_R+1)\}_{PR_{E_R}})\}_{PB_{E_R}}$, then E_R believes that X is sent by entity R and communication channel from R to E_R is secure and no message is observed, modified or replayed by an intruder.

[R11:] If sender entity E_R believes that PR_{E_R} is private key of sender entity E_R, PB_{E_R} is public key of receiver entity E_R, n_{E_R} was a fresh nonce generated by R and used in a previous request call from E_R to E_R, E_R receives message of the form $\{CommMsgRes\ (X||(n_{E_R}+1), \{H(X||(n_{E_R}+1)\}_{PR_{E_R}})\}_{PB_{E_R}}$, then E_R believes that X is sent by entity E_R and communication channel from E_R to E_R is secure and no message is observed, modified or replayed by an intruder.

[R12:] If sender entity E_R believes that communication from all possible receiver entities E_{R_i} $(\forall i = 1...n)$ is secure, then E_R believes that communication channel to E_R is secure and no message is observed, modified or replayed by an intruder.

[R13:] An electronic signature (M_{ieSign}) is a valid signature only when resident verifies that three main parts in signature, viz., $eKYC$, $H(M)$ and $SignChain$ are as expected.

Assumptions

The protocol makes several assumptions. The assumptions relevant for the discussion of this paper are listed below.

[A1:] It is assumed that all sessions from all residents R_i keeps their cookie C_{R_i} secret.

[A2-A6:] The scheme makes several assumptions about public keys. For example, R_i believes that PB_{ASP_i} is public key of ASP_i. Similar to this, other entities also make similar assumptions. These assumptions are listed below.

$$\begin{array}{llll} R_i & \models & \overset{PB_{ASP_i}}{\longmapsto} & ASP_i \quad ...(A2) \\ ESP_i & \models & \overset{PB_{ASP_i}}{\longmapsto} & ASP_i \quad ...(A3) \\ ASP_i & \models & \overset{PB_{ESP_i}}{\longmapsto} & ESP_i \quad ...(A4) \\ UIDAI & \models & \overset{PB_{ESP_i}}{\longmapsto} & ESP_i \quad ...(A5) \\ ESP_i & \models & \overset{PB_{UIDAI}}{\longmapsto} & UIDAI \quad ...(A6) \end{array}$$

[A7:] ASP_i assumes that all valid cookies C_{R_i} are associated with a valid ongoing session from a unique valid user R_i already logged in to ASP_i portal.

$$ASP_i \models C_{R_i} \rightsquigarrow ID_{R_i} \quad \forall i = 1..n$$

[A8-A15:] R_i and ASP_i assumes that all nonce $n*_{R_i}$ (where $*$ is any integer used in the scheme) are fresh. Similar to this, other entities also make similar assumptions. These assumptions are listed below.

$$\begin{array}{lll} R_i & \models & \#n*_{R_i} \quad ...(A8) \\ ASP_i & \models & \#n*_{R_i} \quad ...(A9) \\ ASP_i & \models & \#n*_{ASP_i} \quad ...(A10) \\ ESP_i & \models & \#n*_{ASP_i} \quad ...(A11) \\ ESP_i & \models & \#n*_{ESP_i} \quad ...(A12) \\ UIDAi & \models & \#n*_{ESP_i} \quad ...(A13) \\ UIDAI & \models & \#n*_{UIDAI} \quad ...(A14) \\ ESP_i & \models & \#n*_{UIDAI} \quad ...(A15) \end{array}$$

[A16:] It is assumed that when ASP_i receives message of the form $CommMsg(DataA_j, SignA_j)_{PB_{ASP_i}}$ from ESP_i, it has verified the validity of data, i.e., $\{SignA_j\}_{PB_{ESP_j}} = H(DataA_j)$. The same assumption is made for all entities receiving messages of this form.

Goals to be achieved.

Following are the goals which are envisaged to be achieved by the proposed model.

[G1-G6:] Sender entity must be sure that the data received by receiver entity is same as what was sent by it and is not modified, observed or replayed by an intruder after it was sent by the sender entity. Similarly, receiver entity must be sure that the data received by it is same as what was sent by sender entity and is not modified, observed or replayed by an intruder after it was sent by the sender entity.

$$\begin{array}{llll} ASP_i & \models & R_i & \overset{secure}{\longleftrightarrow} \quad ASP_i \\ R_i & \models & R_i & \overset{secure}{\longleftrightarrow} \quad ASP_i \\ ASP_i & \models & ASP_i & \overset{secure}{\longleftrightarrow} \quad ESP_i \\ ESP_i & \models & ASP_i & \overset{secure}{\longleftrightarrow} \quad ESP_i \\ ESP_i & \models & ESP_i & \overset{secure}{\longleftrightarrow} \quad UIDAI \\ UIDAI & \models & ESP_i & \overset{secure}{\longleftrightarrow} \quad UIDAI \end{array}$$

[G7:] Resident R_i must be sure that at the end what he receives is indeed a digital signature on message M_i, signed by resident's private key and generated by the genuine ESP_i. $R_i \models M_{ieSign} = M_{eSign\_R_i\_ESP_i\_CCA}$

Idealization

BAN idealization of communication messages in communication phase is shown in table 1

Table 1: BAN Idealization of Proposed Protocol (Part I: M1-3) and (Part II: M4-8)

$M1$	ASP_i	$\lhd$	$\{login\ ($ $ID_{R_i} \| PW_{R_i} \| PB_{B_i} \| n1_{R_i}$ $\{H(ID_{R_i} \| PW_{R_i} \| PB_{B_i} \| n1_{R_i})\}_{PR_{B_i}}$ $)\}_{PB_{ASP_i}}$
$M2$	R_i	$\lhd$	$\{loginRes\ ($ $C_{R_i} \oplus (n1_{R_i}+1)$ $\{H(C_{R_i} \oplus (n1_{R_i}+1))\}_{PR_{ASP_i}}$ $)\}_{PB_{B_i}}$
$M3$	ASP_i	$\lhd$	$\{signdocASPReq\ ($ $M_i \| consent_{use\_ekyc} \| consent_{genuse\_at} \| C_{R_i} \| n2_{R_i})$, $\{H(M_i \| consent_{use\_ekyc} \| consent_{genuse\_at} \| C_{R_i} \| n2_{R_i})\}_{PR_{B_i}}$ $)\}_{PB_{ASP_i}}$
$M4$	ESP_i	$\lhd$	$\{signdocESPReq\ ($ $H(M_i) \| AadhaarNo_{R_i} \| ID_{ASP_i} \| License_{ASP_i} \| TID_{ASP_i} \| consent_{use\_ekyc} \| consent_{genuse\_at} \| n1_{ASP_i}$, $\{H(H(M_i) \| AadhaarNo_{R_i} \| ID_{ASP_i} \| License_{ASP_i} \| TID_{ASP_i} \| consent_{use\_ekyc} \| consent_{genuse\_at} \| n1_{ASP_i})\}_{PR_{ASP_i}}$ $)\}_{PB_{ESP_i}}$
$M5$	$UIDAI$	$\lhd$	$\{kycESPReq\ ($ $AT_{R_i} \| H(M_i) \| n1_{ESP_i}$, $\{H(AT_{R_i} \| H(M_i) \| n1_{ESP_i})\}_{PR_{ESP_i}}$ $)\}_{PB_{UIDAI}}$

$M6$	ESP_i	$\triangleleft$	$\{kycESPRes\ ($ $eKYC_{R_i} \| (n1_{ESP_i} + 1)$ $\{H(eKYC_{R_i} \|$ $(n1_{ESP_i} + 1))\}_{PR_{UIDAI}}$ $\}_{PB_{ESP_i}}$
$M7$	ASP_i	$\triangleleft$	$\{signdocESPRes\ ($ $\{eKYC_{R_i} \| H(M_i)\}_{PR_{R_i}} \|$ $PB_{R_i} \| \{PB_{R_i}\}_{PR_{ESP_i}} \|$ $\{PB_{ESP_i}\}_{PR_{CCA}} \|$ $TID_{ASP_i} \| (n1_{ASP_i} + 1),$ $\{H(eKYC_{R_i} \| H(M_i)\}_{PR_{R_i}} \|$ $PB_{R_i} \| \{PB_{R_i}\}_{PR_{ESP_i}} \|$ $\{PB_{ESP_i}\}_{PR_{CCA}} \|$ $TID_{ASP_i} \|$ $(n1_{ASP_i} + 1))\}_{PR_{ESP_i}}$ $)$ $\}_{PB_{ASP_i}}$
$M8$	R_i	$\triangleleft$	$\{signdocASPRes\ ($ $M_i \| \{eKYC_{R_i} \| H(M_i)\}_{PR_{R_i}}$ $\| PB_{R_i} \| \{PB_{R_i}\}_{PR_{ESP_i}} \|$ $\{PB_{ESP_i}\}_{PR_{CCA}} \| (n2_{R_i} + 1)$ $\{H(M_i \| \{eKYC_{R_i} \| H(M_i)\}_{PR_{R_i}}$ $\| PB_{R_i} \| \{PB_{R_i}\}_{PR_{ESP_i}} \|$ $\{PB_{ESP_i}\}_{PR_{CCA}} \|$ $(n2_{R_i} + 1))\}_{ASP_i}$ $)$ $\}_{PB_{B_i}}$

Analysis

[P1-P6:] Using messages M1, M3 and rule R7, it can be deduced that ASP_i believes that communication from R_i to ASP_i is secure. Using messages M2, M8 and rule R11, it can be deduced that ASP_i believes that communication from ASP_i to R_i is secure. From these two deductions, it can further be deduced that ASP_i believes that communication between R_i and ASP_i is secure in both directions.

Using M1, M3, R7, R8,
$ASP_i \models R_i \xrightarrow{secure} ASP_i$ ($I1$)

Using M2, M8, R11,
$ASP_i \models R_i \xleftarrow{secure} ASP_i$ ($I2$)

Using I1 and I2,
$ASP_i \models R_i \xleftrightarrow{secure} ASP_i$ ($G1$: *Proved*)

Using M1, M3, R10, R11
$ASP_i \models R_i \xrightarrow{secure} ASP_i$ ($I3$)

Using M2, M8, R8,
$ASP_i \models R_i \xleftarrow{secure} ASP_i$ ($I4$)

Using I3 and I4,
$ASP_i \models R_i \xleftrightarrow{secure} ASP_i$ ($G2$: *Proved*)

Using M4, R7,
$ESP_i \models ASP_i \xrightarrow{secure} ESP_i$ ($I5$)

Using M7, R11,
$ESP_i \models ASP_i \xleftarrow{secure} ESP_i$ ($I6$)

Using I5 and I6,
$ESP_i \models ASP_i \xleftrightarrow{secure} ESP_i$ ($G3$: *Proved*)

Using M4, R11,
$ASP_i \models ASP_i \xrightarrow{secure} ESP_i$ ($I7$)

Using M7, R8,
$ASP_i \models ASP_i \xleftarrow{secure} ESP_i$ ($I8$)

Using I5 and I6,
$ASP_i \models ASP_i \xleftrightarrow{secure} ESP_i$ ($G4$: *Proved*)

Using M5, R7,
$UIDAI \models ESP_i \xrightarrow{secure} UIDAI$ ($I7$)

Using M6, R11,
$UIDAI \models ESP_i \xleftarrow{secure} UIDAI$ ($I8$)

Using I7 and I8,
$UIDAI \models ESP_i \xleftrightarrow{secure} UIDAI$
($G5$: *Proved*)

Using M5, R11,
$ESP_i \models ESP_i \xrightarrow{secure} UIDAI$ ($I9$)

Using M6, R8,
$ESP_i \models ESP_i \xleftarrow{secure} UIDAI$ ($I0$)

Using I9 and I10,
$ESP_i \models ESP_i \xleftrightarrow{secure} UIDAI$ ($G6$: *Proved*)

[P7:] Using message M9 and rule R13, it can be deduced that R_i believes that $\{M_i\}_{eSign}$ is a valid electronic signature.

Using M9 and R13,
$R_i \models \{M_i\}_{eSign} = \{M_i\}_{eSign\_R_i\_ESP_i\_CCA}$
($G7$: *Proved*)

8 Conclusion

This work is an extension of the work [4] on enhancing amortized performance of *eSign* by using digital access tokens including claims from *ESP* and *UIDAI*. In this work, the digital access token introduced in [4] is extended to include privacy and fine-grained access control statements for access to resident's eKYC data. This enhanced token can be used by third entities to access the protected eKYC data with better privacy and fine-grained access control rules enforced by the resident. A formal security analysis of the proposed model using *BAN logic* is also presented.

References

[1] UIDAI, "Unique Identification Authority of India", [Online]. Available: https://uidai.gov.in (2017).

[2] Wikipedia, "Aadhaar", [Online]. Available: https://en.wikipedia.org/wiki/Aadhaar (2017).

[3] Reserve Bank of India, "Master Direction - Know Your Customer (KYC) Direction, 2016", [Online]. Available: https://rbidocs.rbi.org.in/rdocs/notification/PDF s/18MDKYCD8E68EB1 3629A4A82BE8E06E606C57E57.PDF, 2018.

[4] Bakshi, Puneet, Neelakantan Subramanian, and Sukumar Nandi. "Using digital tokens to improve amortized performance of eSign." 2018 IEEE 16th Intl Conf on Dependable, Autonomic and Secure Computing, 16th Intl Conf on Pervasive Intelligence and Computing, 4th Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress (DASC/PiCom/DataCom/CyberSciTech). IEEE, 2018.

[5] Paquin, Christian, and Greg Zaverucha. "U-prove cryptographic specification v1. 1." Technical Report, Microsoft Corporation (2011).

[6] Chaum, David. "Blind signatures for untraceable payments." Advances in cryptology. Springer, Boston, MA, 1983.

[7] Hardt, Dick. The OAuth 2.0 authorization framework. No. RFC 6749. 2012.

[8] Nakamoto, Satoshi. "Bitcoin: A peer-to-peer electronic cash system." (2008).

[9] UIDAI, "Aadhaar e-KYC API Specification - Version 2.1", [Online]. Available: http://www.cca.gov.in/cca/sites/default/files/files/eSign-APIv2.1.pdf, 2017

[10] CCA, "eSign Service", [Online]. Available: http://cca.gov.in/cca/?q=eSign.html (2017).

[11] Burrows, Michael, Martin Abadi, and Roger Michael Needham. "A logic of authentication." Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences 426.1871 (1989): 233-271.

[12] Rivest, Ronald L., Adi Shamir, and Leonard Adleman. "A method for obtaining digital signatures and public-key cryptosystems." Communications of the ACM 21.2 (1978): 120-126.

[13] Merkle, Ralph C. "A digital signature based on a conventional encryption function." Conference on the theory and application of cryptographic techniques. Springer, Berlin, Heidelberg, 1987.

[14] Paquin, Christian, and Greg Zaverucha. "U-prove cryptographic specification v1. 1." Technical Report, Microsoft Corporation (2011).

[15] Chaum, David. "Blind signatures for untraceable payments." Advances in cryptology. Springer, Boston, MA, 1983.

[16] Hardt, Dick. The OAuth 2.0 authorization framework. No. RFC 6749. 2012.

[17] Nakamoto, Satoshi. "Bitcoin: A peer-to-peer electronic cash system." (2008).

[18] PKIA2017, "Development of Smart Authentication and Identification in Asia", [Online]. Available: http://pkiindia.in/pkia/#preceding, 2017.

[19] CCA, "Pki framework in India", [Online]. Available: http://www.cca.gov.in/cca/?q=pki˙frame work.html (2015).

[20] CCA, "Public key certificate classes", [Online]. Available: http://www.cca.gov.in/cca/?q=node/45 (2017).

[21] CCA, "Empanelled eSign Service Providers", [Online]. Available: http://www.cca.gov.in/cca/?q=service-providers.html (2017).

[22] Jones, Michael, John Bradley, and Nat Sakimura. Json web token (jwt). No. RFC 7519. 2015.

[23] Jones, Mike, et al. Cbor web token (cwt). No. RFC 8392. 2018.