A New Delegated Authentication Protocol based on PRE

Anass Sbai, Cyril Drocourt[a] and Gilles Dequen[b]
MIS Laboratory, University of Picardie Jules Verne, Amiens, France
{

Keywords: Authentication, Delegation, Single Signe On, Proxy Re-Encryption.

Abstract: New trends highlight the use of delegated authentication solutions where identity providers do not need to synchronize user credentials with services. It is a facility for service providers and also for users who do not have to create multiple accounts. Different solutions for single sign-on and delegated authentication exist. Most of these solutions require many exchanges between the different actors involved in the protocol, an additional TLS layer and/or the use of signature schemes which, in terms of security, rely on random oracles for reasons of efficiency. In this article, we recall the concept of the best known solutions (e.g. Kerberos, OpenID, ...), briefly discuss the possibility of using one-way accumulators and define the Proxy Re-Encryption (PRE). Next, we propose a new delegated authentication protocol that allows users to authenticate anonymously on insecure networks and therefore asynchronously without direct communication between identity providers and service providers while minimizing the number of interactions. We based our solution on the use of PRE which could be instantiated by schemes based on standard assumptions. We first show how our protocol behaves against different types of attacks. Then in a more formal manner, we present the proof of security based on an adaptation of BAN logic method that supports the use of PRE functionalities.

1 INTRODUCTION

We are increasingly using remote services, whether on the web or other platforms. Most of these require the user to create an account and then log in. This allows them to limit access. There are several methods for authentication depending on the sensitivity of data or service requested. Typically, login/password, OTP (one time password) and certificates are the most commonly used. But recently, due to the technological evolution, biometrics are also used to identify a user through his physiological characteristics such as face, iris and fingerprint (Alizadeh et al., 2016). For users, having multiple accounts is restrictive because the most used method is login/password authentication, which implies that the user must remember each password since it should ideally be different for each service. Then appeared the single sign-on approach (SSO) which allows users to access multiple services by authenticating only once. The model used by this authentication system is mainly based on three actors: the client who requests access to a resource or a service, the identity provider (Idp) that stores clients' credentials and the service provider that provides ressources and services based on clients' permissions communicated by the Idp. There are three main approaches proposed for implementing the SSO system:

- Centralized Approach: it is mainly used for services that depend on the same entity which manages its own security policy. For this purpose, it is possible to use a Kerberos authentication mechanism.
- Federated Approach: it enables extended access control and SSO across organizational boundaries by distributing the control and maintenance of the different activities, thus providing more convenience and efficiency. It allows each entity to manage their own security policy and certain parts of data users could be shared with other interactive services. Different solutions exist, mainly SAML, OpenID...
- Cooperative Approach: in the same sense, it allows each partner to manage their own authenticating system but it differs from the latter in the sense that the users' credentials are not shared between the different services, example is Central Authentication Service (CAS).

The different systems mentioned above are based on what we consider as conventional methods e.g. sym-

[a] https://orcid.org/0000-0003-1636-9462
[b] https://orcid.org/0000-0001-7497-1182

Sbai, A., Drocourt, C. and Dequen, G.
A New Delegated Authentication Protocol based on PRE.
DOI: 10.5220/0010542404680478
In *Proceedings of the 18th International Conference on Security and Cryptography (SECRYPT 2021)*, pages 468-478
ISBN: 978-989-758-524-1

metric ciphers as it is the case for Kerberos, or asymmetric crypto-based systems which necessarily use signatures. On the other hand, there are cryptographic solutions that we consider to be unconventional such as signature groups, one way accumulators, etc. Which are not or rarely used in practice usually due to efficiency. All cryptographic solutions used in practice for signing rely on the random oracle model for efficiency reasons (Chaidos and Couteau, 2018).

Can we avoid these conventional methods and solve delegation issues e.g. authenticating users asynchronously through identity providers and without synchronizing their credentials with services, while minimizing the number of communications required and increasing the robustness of the system? This is the main question that we aim to answer through this paper.

2 RELATED WORKS

***Kerberos*:** is based on Needham-Schroeder symmetric key protocol. In terms of delegation, Kerberos proposes two types of delegations: unconstrained and constrained. In the first one, the server or the service account that is granted this right is able to impersonate the user to communicate with any service on any machine. Historically, this was the only choice when the principle of delegation was introduced, but it has been supplemented by the principle of Constrained Delegation. It allows service administrators to specify and enforce application approval limits by limiting the scope to which application services can act on behalf of a user. Service administrators can specify which front-end service accounts can perform delegation on their primary services.

***Shibboleth*:** is one of the most popular open sources SSO platforms for local identity and access management. The Shibboleth framework is based mainly on the OASIS Security Assertion Markup Language (SAML). It provides a SSO service and an attribute-based authorization while maintaining user's privacy. Like most of the other solutions, the protocol defines two functional components which are the Service Provider (SP) that handle authorization and the IdP. The latter manages and maintains the identity of users. It is designed to provide federated identity with the main assumption: the IdP and the SP trust each other within a federation. This trust is based on Public Key Infrastructure (PKI) and managed using PKI certificates.

***OpenId*:** was created for federated authentication. It is designed as a user-centric identity management system where any identity provider can be used by the users (with the exception of whitelists). They can even establish their own provider. There is no need to pre-choose or negotiate a deal with the providers to allow users to use any other account they have. In 2007 OpenID Attribute Exchange extension (OpenID AX) was published (Hardt et al., 2007). It defines how to store or update attributes information on the OpenID Provider (OP) and how to retrieve those attributes. In 2014 OpenID Connect was defined as an evolution of the OpenID standard and it is based on the OAuth 2.0 protocol by adding an authorization layer. It relies on the JSON Web Token (JWT) standard for exchanging identity information or any other relevant information to the service provider called in OpenID context Relying Party (RP).

***Oauth*:** introduces an authorization layer that enables applications to obtain limited access to user accounts. It could be interfaced with kerberos and other authentications solution described above. The system may involve up to four actors : Ressource Owner (RO), application Client (C), Authorization Server (AS) and the resource server (RS). In general, a client C requests access to resources controlled by the RO and hosted by the RS. Instead of using the RO credentials to access the protected resources, C obtains an access token (A string indicating scope, lifetime, and other specific access attributes). Access tokens are assigned to third party clients (C) by an AS with the approval of the RO. The C uses the access token to access the protected resources hosted by the RO.

3 OUR CONTRIBUTION

In this section we will detail our contribution by first giving the main idea, then defining the PRE, and finally explaining the different phases of our protocol.

3.1 Main Idea

The proposed solution allows delegating authentication without the use of signatures and without the help of a Secure Socket Layer (SSL). Thus it is possible to authenticate the user on non-secure networks without the need to synchronize user credentials with services. Our system involves three actors including the IdP, the client and the SP. Nevertheless, a phase of enrollment must be pre-established between the client and the IdP as well as a phase of establishing a trust relationship between the IdP and the the SP. Theses phases mainly consist in exchanging identification in-

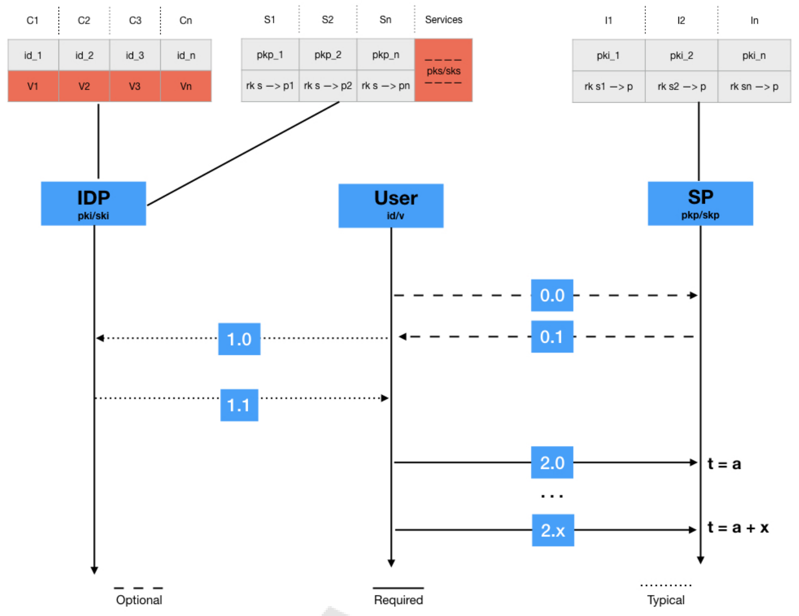

Figure 1: Main actors and interactions of our protocol.

formation as well as pre-shared keys. The Figure 1 shows the different actors, interactions and the different information necessary for our protocol. The secret data are presented in the tables in red background. We don't need to store any user-related information on the side of service providers.

- User ⟷ SP:
  - 0.0 : The user asks the list of IdPs trusted by the SP.
  - 0.1 : The SP returns the corresponding list.
- User ⟷ IDP:
  - 1.0 : The user chooses the IdP, creates an authentication request and sends it to the corresponding IdP.
  - 1.1 : The IdP verifies the validity of the request, authenticates the user and then responds with the authentication token.
- User ⟷ SP:
  - 2.0 : The user creates an ephemeral token, adds it to the received one and authenticates with the SP.
  - 2.x : At each connection the user resumes 2.0 with a slight modification.

Once enrollment and trust establishing phases are completed, we can authenticate the users to the SP without the need of direct communication between the IdP and the SP with the least amount of interaction possible (1). Also, it allows the user, once authenticated, to use any service that trusts this IdP (2) and to not need to store any information about the client on the SP side (3). In order to do so, we will need to use PRE as a core technology.

In the literature, it is hard to find an efficient solution that combines these three advantages. For example, OpenId needs a direct interaction between the IDP and the SP, it involves at least five interactions and uses JWT which are signed using a secret with the HMAC algorithm or a public / private key pair using RSA or ECDSA. The security model in any case relies on the random oracle model. A signature is by design based on Non-Interactive Zero Knowledge (NIZK) protocol. These protocols are either secure in the standard model but inefficient or practically efficient and usually built from the Fiat-Shamir heuristic which are secure in the random oracle model. Thus, we have decided to exclude the use of signatures and thereby create a protocol that could rely on cryptographic schemes that are proved to be secure in the standard model while maintaining a good level of efficiency. Of course, there is always a compromise between performance in terms of time, storage and level of security.

There are several solutions that reduce the storage and complexity of information management related to identification, such as accumulators. This cryptographic scheme was proposed as a decentralized alternative to digital signatures in the design of secure distributed protocols (Benaloh and De Mare, 1993). For verification of membership in the conventional method a trusted third party digitally signs each member's IDs and distributes its public key. To verify

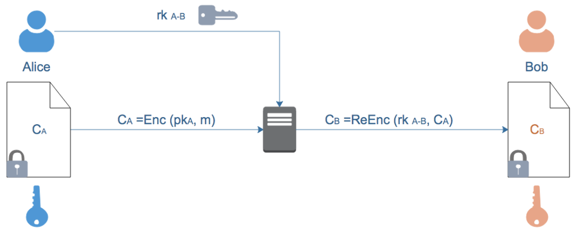

Figure 2: Main actors and interactions of a proxy re-encryption scheme.

a user's membership it is sufficient to provide their signed identity while the public key is accessible to everyone (even non-members can check it). While using an accumulator, it could be initialized and diffused by a member of a group and not necessarily by a trusted third party, the members exchange their identification information then each one calculates his witness and the accumulator. The latter may not be kept since it can be calculated. For static accumulators the group members cannot change which is restrictive, whereas with dynamic accumulators we can add and revoke members of the group. But the present development of the latter schemes does not allow an actual use (Tremel, 2013).

3.2 PRE Background

Proxy Re-Encryption (PRE) allows the delegation of the decryption rights on Alice's data only for the intended recipients (we will also refer to Alice as the delegator and Bob as the delegate). The first scheme was proposed by Blaze, Bleumer, and Strauss (Blaze et al., 1998). Their goal was to avoid that the data must be recovered, decrypted then encrypted with the delegate's key. And thus, relying on a semi-trusted proxy that converts the ciphers using re-encryption keys created by the delegator as illustrated in Figure 2.

The major disadvantage of their scheme is that Alice's delegation to Bob automatically allowed Bob's delegation to Alice. This will later be called bidirectional PRE. This property is due to the fact that re-encryption keys were created using the private keys of the two actors. Y.Dodis (Ivan and Dodis, 2003) formalizes the design of proxy re-encryption schemes by categorizing these systems in two types: unidirectional and bidirectional. Later, Ateniese (Ateniese et al., 2006) gives a more formal definition for PRE and defines concretely the properties. We define below the properties that we consider to be critical to the completion of our solution:

- *Unidirectional*: Delegation of decryption rights from Alice to Bob does not allow Alice to decrypt Bob's cipher.
- *Non-interactive*: The re-encryption key can be generated by Alice without interacting with Bob and thus using only Bob's public key.
- *Collusion-safe*: If the proxy and Bob collude, they should not get Alice's secret key.
- *Non-transitive*: The proxy can not re-delegate re-encryption rights. (e.g. from $Rk_{a \to b}$ and $Rk_{b \to c}$ the proxy can not calculate $Rk_{a \to c}$)
- *Non-transferable*: The proxy and delegates can not redefine decryption rights. (e.g. from $Rk_{a \to b}$ and Pk_c and Sk_b we can not calculate $Rk_{a \to c}$)
- *Key-private*: This means that by using the re-encryption key it is not possible to recover the public keys of both the delegate and the delegator.

In 2008 (Canetti and Hohenberger, 2007) propose the first bidirectional CCA secure PRE scheme where they prove the security of his scheme using the Universal Composability framework (Canetti, 2001). In (Deng et al., 2008), the authors deal with the open problem presented by Canetti (Canetti and Hohenberger, 2007) concerning the construction of a CCA secure PRE without pairing. Ateniese in (Ateniese et al., 2009) formalizes the notion of key privacy. Their construction is single-use CPA secure. (Chow et al., 2010) conduct a CCA attack on the Shao's system (Shao and Cao, 2009) and show how to fix the issue. They proposed their own scheme without using pairing and relying only on ElGamal and the Schnorr signature. This scheme was implemented in (Sbai et al., 2019) and used as a service for data sharing. (Selvi et al., 2017) find a flaw in the security proof of Chow's construction and propose to fix it. The system is unidirectional CCA secure in the random oracle model. (Zhang et al., 2013) proposed a unidirectional PRE scheme that has been claimed to be CCA secure without relying on pairing nor random oracles. In (Sbai et al., 2020) we first show that the proposal of (Zhang et al., 2013) is not CCA-secure, then we construct the first CCA secure PRE in the standard model that does not use pairing.

In (Nunez et al., 2012) authors propose a solution for privacy preserving in the context of OpenID AX that is based on PRE, where the attributes are stored encrypted. Thus, the user must create a re-encryption key for each RP that will allow the OP to re-encrypt the stored attributes for the RP. The same idea could be applied for Shibboleth or SAML. So it is an extra layer to guarantee the privacy.

3.3 The Proposed Protocol

The Figure 1 illustrates the different communications required to authenticate a user as well as the information to be stored within each entity that we will detail below.

As mentioned before an enrollment phase is necessary where the IdP would have stored the client's identifier e.g. *id* and the secret to authenticate it e.g. *v* . We have taken as an example the identifier/secret for identification to illustrate our protocol (see Table 1) but other identification methods can be used such as biometrics. For services at the time of trust establishment the IdP should have generated a secret asymmetric key pair (pks/sks) in addition to its own asymmetric public/private key pair (pki/ski). The secret asymmetric key pair will be used to create an authentication token and re-encryption keys to the service concerned; the other key pair will be used to verify the validity of user's authentication request. Instead of using a pre-shared secret symmetric keys between the IDP and the SP in order to verify the origin of the tokens. We use an asymmetric secret pair key known only by the IDP, through which we generate a re-encryption key from IDP to the corresponding SP. As shown in Table 2 the IdP will then have stored for each service its public key as well as the re-encryption key generated for it. The use of a PRE scheme that is key private is then essential. This prevents the public key from being reconstructed through the re-encryption key and thus keeping the asymmetric key pair secret. On the service side it should have stored this re-encryption key as well as the public key of this IdP (see Table 3). The service in turn may have established trust with other IdPs.

Table 1: Users credentials stored by the IDP.

C1	C2	C3	...	Cn
id1	id2	id3	...	idn
v1	v2	v3	...	vn

Table 2: Public, re-encryption and secret asymmetric pair keys stored by the IDP.

S1	S2	...	Sn	Services
pkp_1 $rk_{s\to p1}$	pkp_2 $rk_{s\to p2}$	... ...	pkp_n $rk_{s\to pn}$	pks/sks

Table 3: Public and re-encryption keys information stored by the SP.

I1	I2	I3	...	In
pki_1	pki_2	pki_3	...	pki_n
$rk_{s1\to p}$	$rk_{s2\to p}$	$rk_{s3\to p}$	...	$rk_{sn\to p}$

We describe below the technical outlines of the protocol using Figure 3:

- 1.0: Pick $r \overset{\$}{\leftarrow} \{0,1\}^{l_0}$ then $Auth\_req = Enc(pki, id||v||r)$.
- 1.0 $\rightarrow$ 1.1: $Dec(ski, Auth\_req)$, and verify the validity of *id* & *v*.
- 1.1: $Auth\_token = Enc(pks, scope)$; $scope = (timestamp||expire\_date||r)$.
- 2.0: $Eph\_token = Enc(pkp, pki||r)$; $Auth\_resp = Auth\_token||Eph\_token$.
- 2.0 $\rightarrow$ 2.x: $Dec(skp, Re\text{-}Encrypt(rk_{s\to p}, Auth\_token))$; $Dec(skp, Eph\_token)$ then verify the validity of the *scope* and *r*.
- 2.x: $Auth\_token||Enc(pkp, pki||r+x)$.

First of all, the user asks the SP for the list of trusted IdPs (0.0). The SP then returns its list of known IdPs(0.1). The user chooses the corresponding IdP and sends an authentication request ($Auth\_req$) which contains information related to his identification and a random coin all encrypted with the IdP's public key(1.0) in order to collect the authentication token ($Auth\_token$). The IdP, upon receiving the $Auth\_req$, deciphers it and verifies the validity of the information. If it corresponds to some user, the latter creates an $Auth\_token$ by encrypting a scope containing primarily a timestamp, expiry date and the random coin created by the user(1.1). Other information related to authorization could be added. However, issues related to access control are out of the scope of this paper.

For the latter the IdP uses its own secret asymmetric encryption key intended for services. The user upon getting his $Auth\_token$, will encrypt this time his random coin with the public key of the SP which corresponds to the ephemeral token (Eph_token) , concatenate it with the $Auth\_token$ received and send it to the SP(2.0). The SP verifies the validity of the authentication response $Auth\_resp$. It starts by re-encrypting the first part of the response which could also be done by an independent semi-trusted party. Then the SP decrypts the resulting cipher and the second part of the $Auth\_resp$. If the random coin matches, the authentication is then successful. The SP stores the random coin until the next connection, the $Auth\_token$ during its term of validity and the number of connections used with this token. At each connection the user creates the Eph_token by encrypting the incrementation of the random coin (e.g. at the time of x connection (2.x) the value of this coin will correspond to $r+x$ and the $Auth\_token$ will be still the same.) The SP will only have to check the existence of this token, the number of connections related to it and the value of the random coin.

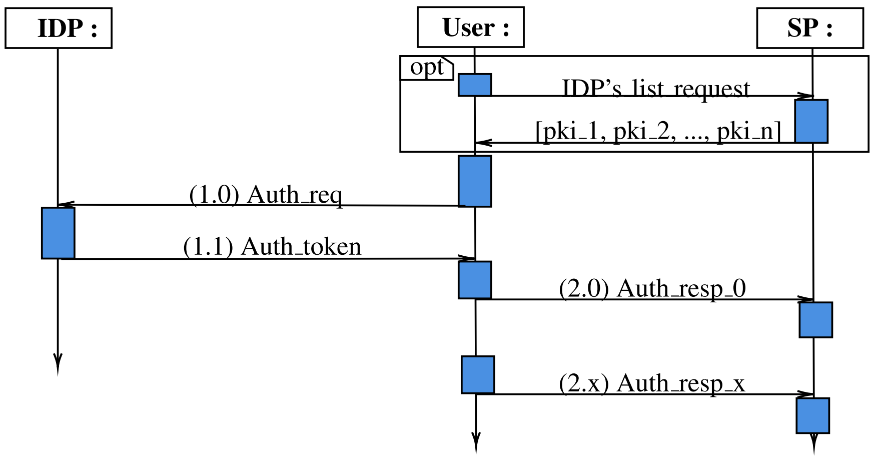

Figure 3: Proposed protocol sequence diagram.

4 SECURITY ANALYSIS

4.1 Behavioral Study

A secure delegated authentication protocol should have the following properties : completeness, security against forgery attack, security against replay attack and security against "man in the middle attack". Here we will introduce and define these properties and show that our proposal is secure.

- **Completeness.** The delegate (e.g. the SP) is always able to identify the source of the authentication response created by the delegator (e.g. the IdP) and the client (e.g. the user), if the different actors follow the protocol.
  Proof. When the SP receives the authentication response $Auth\_resp = Auth\_token||Eph\_token$, the re-encryption of $Auth\_token = Enc(pks, scope)$ will be valid only if it was created by the IdP. Indeed the IdP is the only one in possession of pks. The Eph_token should contains the same random coin found in the scope. Thus if both the user and IdP execute the protocol strictly, the SP will always identify the source of the authentication response.

- **Security of Forgery Attack.** When an attacker wants to forge a valid token and sends it to the intended SP, the protocol is able to withstand the forgery attack.
  Proof. To launch a forgery attack, the attacker must create a valid encrypted message corresponding to $Auth\_token = Enc(pks, scope)$. Then create the Eph_token which is $Enc(pkp, pki||r)$. The second part could be created easily by the attacker since pkp is known. However, the first part that correspond to the $Auth\_token$ could not be created unless the attacker knows the secret key pks.

- **Security of Replay Attack.** When a token transmission is maliciously repeated by an attacker, the intended SP deny the authentication request.
  Proof. An attacker can obtain a valid token $Auth\_resp$ (e.g. $Enc(pks, scope)$ || $Enc(pkp, pki||r)$) by eavesdropping. When the attacker wants to impersonate the user, the token obtained corresponds to the cipher of a random coin r chosen by the user, but the value of this coin is incremented after each connection. Therefore, re-transmisting the same token $Auth\_resp$ will not result to a successful authentication. The SP could detect the attack by storing the last value of the coin and compare it with the received one.

- **Security of Man in the Middle Attack.** An attacker cannot create a valid authentication request/response on behalf of the different actors.
  Proof. According to our protocol, if the attacker can create a valid authentication request on behalf of the user e.g. $Enc(pki, id||v||r) = Auth\_req$ he must know the authentication information related to the user. This is impossible because it implies that he has access to the secret key allowing to decrypt the $Auth\_req$. A second way to conduct an attack is to create a valid authentication response $Enc(pks, scope)||Enc(pkp, pki||r)$ on behalf of the user. This is also impossible since it implies that the attacker knows the value of the random coin chosen by the user. The only way for the attacker to know this value is to break the

PRE scheme or to find the random coin by a brute force attack. However, it is impossible to achieve this in the lifetime of the token because the length of the random coin is one of the security parameter of our protocol.

4.2 Correctness

At first sight we can see that all the communication is encrypted. An eavesdropper would not be able to interpret the packets into something human-readable. This is guaranteed by end to end encryption. Note that the registration and the trust establishment phases are done securely.

The use of random coins plays a major role in our solution. The token by itself cannot authenticate the client but can only confirm its origin; it does this by keeping secretly the IdP's asymmetric key pair pks/sks intended for services. Hence, if re-encryption is done correctly, we are sure that the token was created by the IdP. Nevertheless, anyone could intercept the token and send it to any SP to be authenticated. The use of random coin allows us to ensure that the user concerned is the entity that made the authentication request to the IdP. Its incrementation prevents replay attacks because the value of the coin will be different for each connection. The only way for an attacker to pretend to be the user is to guess the value of the random coin. In addition to that, it makes it possible to authenticate into SPs anonymously. Note that the PRE must have the properties explained in section 3.2, which are : {Unidirectional, Non-interactive, Collusion-safe, Non-transitive, Non-transferable, Key-private}

5 PROOF OF SECURITY

5.1 BAN Logic

Proposed in 1989 by Burrows, Abadi and Needham (Burrows et al., 1989), it is one of the first formal methods of proof for authentication protocols. It is based on the beliefs of the trusted parties involved in the protocols. It allows to study the evolution of these beliefs following the exchanges between the different entities. A verification with the BAN logic does not necessarily imply that no attack on the protocol is possible. Although, a proof with BAN logic is a good proof of accuracy for an authentication protocol based on well-defined assumptions. It allows many questions to be answered in a formal way and to exclude certain possible attacks and errors.

5.1.1 BAN Logic Terms

A protocol is formalized using the following terms:

- Participants: Each entity or actor involved in the execution of the protocol is considered as a participant and noted by a capital letter.
- Keys: Any symmetric or asymmetric keys used are taken into account for the proofs of security. Asymmetric key pair is denoted by (K, K^{-1}).
- Messages: Any cleartext or ciphertext used or transmitted are taken into account.

5.1.2 BAN Logic Notations

Based on the terms defined above, the protocol is translated using the following notations and formulas:

- P believes in X: $P \mid\equiv X$
- P sees X: $P \triangleleft X$
- P once said X: $P \mid\sim X$
- P controls X: $P \Rightarrow X$
- The formula X is fresh: $\#(X)$
- P and Q shared a secret key K: $P \overset{K}{\leftrightarrow} Q$
- P holds the public key K: $\overset{K}{\mapsto} P$
- P and Q shared a secret X: $P \overset{X}{\rightleftharpoons} Q$

5.1.3 BAN Logic Rules

Now that we have defined the terms and the different existing notations, we can follow certain laws, also called logical postulate in order to generate new statements.

- Checking the origin of the message:
  - For Shared Keys: If P believes that he has shared a secret key K with Q and sees an encrypted message X under this secret key, then P believes Q once said X.

$$\frac{P \mid\equiv Q \overset{K}{\leftrightarrow} P, P \triangleleft \{X\}_K}{P \mid\equiv Q \mid\sim X}$$

  - For Public Keys: If P believes that K is the public key of Q and sees a message X signed with the corresponding private key $-K$, then P believes Q once said X.

$$\frac{P \mid\equiv \overset{K}{\mapsto} Q, P \triangleleft \{X\}_{K^{-1}}}{P \mid\equiv Q \mid\sim X}$$

- For Shared Secrets: If P believes that he has shared a secret Y with Q and sees a message X joint with this secret, then P believes Q once said X.

$$\frac{P \mid\equiv Q \overset{Y}{\rightleftharpoons} P, P \triangleleft \langle X \rangle_Y}{P \mid\equiv Q \mid\sim X}$$

- Checking Message Freshness: If P believes that a message X is fresh and believes that Q once said X, then P believes that Q believes X .

$$\frac{P \mid\equiv \#(X), P \mid\equiv Q \mid\sim X}{P \mid\equiv Q \mid\equiv X}$$

- Checking the Reliability of the Origin: If P believes that Q controls X and that Q believes X, then P believes X.

$$\frac{P \mid\equiv Q \Rightarrow X, P \mid\equiv Q \mid\equiv X}{P \mid\equiv X}$$

- Other rules:
  - If P sees X and Y then he sees X.

$$\frac{P \triangleleft (X,Y)}{P \triangleleft X}$$

  - If P sees X joint with a secret Y then he sees X.

$$\frac{P \triangleleft \langle X \rangle_Y}{P \triangleleft X}$$

  - If P believes that he has shared a secret key K with Q and sees an encrypted message X under this secret key, then P sees X.

$$\frac{P \mid\equiv Q \overset{K}{\leftrightarrow} P, P \triangleleft \{X\}_K}{P \mid\equiv X}$$

  - If P believes that K is his public key and sees a message X encrypted with K, then P sees X.

$$\frac{P \mid\equiv \overset{K}{\mapsto} P, P \triangleleft \{X\}_K}{P \triangleleft X}$$

  - If P believes that K is the public key of Q and sees a message X signed with the corresponding private key $-K$, then P sees X.

$$\frac{P \mid\equiv \overset{K}{\mapsto} Q, P \triangleleft \{X\}_{K^{-1}}}{P \triangleleft X}$$

Note that this list is not exhaustive. However the flexibility of the BAN logic allows to add appropriate constructions and rules. There are several extensions of the BAN method such as GNY (Gong et al., 1990) which adds the notion of possession and non provenance. This method constructs new logical postulates different from the BAN and can be applied to other cryptographic protocols outside the authentication system. In the context of our proof, we analyze a system of delegated authentication. Also, the notions of possessions and non-origin have no effect on our protocol, so we use the logical BAN with a small modification. Some notions that do not exist in the two logics BAN and GNY have been definedwhich integrate elements allowing the interpretation of the functions of PRE in the BAN and implicitly for the GNY method too.

5.2 BAN Logic Adaptation

In order to be able to use the logical BAN for the proof of our protocol, we propose to extend it. Some notions are not formalized and must be included in the proof such as re-encryption keys and re-encryption. The new terms we will use to formalize these notions will be :

- Re-encryption Key: $K_{A \to B}$
- Re-encryption: $\{\{X\}_{K_A}\}_{K_{A \to B}}$

We also define new rules for making inferences from new assumptions using the terms we have just defined.

The re-encryption of a message X encrypted with the public key from A via a re-encryption key from A to B corresponds to the encryption of message X with the public key from B under the condition that the entity performing the re-encryption knows the re-encryption key:

$$\frac{P \mid\equiv \xmapsto{K_{Q \to P}} Q, P \triangleleft \{\{X\}_{K_Q}\}_{K_{Q \to P}}}{P \triangleleft \{X\}_{K_P}}$$

The second rule concerns an unusual case that we used in our protocol where the encryption key will be secret. We note the secret asymmetric key pair as $+(K, K^{-1})$. This allows us to verify the authenticity of encrypted messages for well-defined entities. These entities must have received a re-encryption key with which they will be able to re-encrypt and then decrypt the message. If these two processes are valid, then the encrypted message has been created by the holder of the secret key. We include this concept in the following rule:

$$\frac{P \mid\equiv Q \Rightarrow +K, P \triangleleft \{X\}_K, P \triangleleft X}{P \mid\equiv Q \mid\sim X}$$

Thus, we assume that the jurisdiction of participant Q over his secret asymmetric key pair $+(K, K^{-1})$ is that P can read the encrypted and plaintext message allows us to deduce that P believes that Q has said this message.

6 PROOF

6.1 Definition of Goals and Sub-goals of the Protocol

The first part of the authentication process is the validation of the user's identity by his IdP. This is done by encrypting the random value generated by user C with the secret SP key held by the IdP. Since the identity information is not disclosed to the SP and the random value can be intercepted, a second part is required for the validation of authentication process to the SP. It consists in encrypting the same random value but this time by the user to the SP. This can result in a secret message sharing r. The first validation consists in the fact that the IdP believes that C has shared a secret with him. The second validation allows the SP to believe that the same secret has been shared with him by the IdP.

- Goals:
  1 . $SP \mid\equiv IdP \overset{r}{\leftrightharpoons} SP$
  2 . $IdP \mid\equiv C \overset{r}{\leftrightharpoons} IdP$

These goals make it possible to check whether the IdP believes he has exchanged a secret with user C (Goal 2.) and if the SP has really exchanged the same secret with the IdP (Goal 1.).

- Sub-goals:
  1.1. $IdP \mid\equiv r$
  1.2. $SP \mid\equiv IdP \mid\equiv r$
  2.1. $C \mid\equiv r$
  2.2. $IdP \mid\equiv C \mid\equiv r$

In order to SP believes that it has shared a secret with the IdP, the IdP must believe in the validity of the shared secret and SP must believe that the IdP believes in the validity of the shared secret. The same reasoning applies to the secret exchange between the IdP and C.

6.2 Definition of Assumptions

As explained previously, our protocol assumes that a registration and trust relationship phase is pre-established. It results in the exchange of a secret message between the IdP and the user, which corresponds to (id, v). Regarding the trust relationship between the IdP and the SP, it results in the exchange of the re-encryption key. Thus we assume that the SP holds the re-encryption key and believes that the IdP controls the secret asymmetric key through which the re-encryption key was generated. We also assume that the "timestamps" are considered valid by the different entities. The last hypothesis assumes that the IdP believes that C has control over the randomness it generates. This hypothesis cannot be applied to the SP since it has no pre-established relationship with the user. On the other hand, at the IdP level, the received random values are always stated with the identifier of the user in question. We can thus make the following assumptions :

1. $IdP \mid\equiv \xmapsto{K_I} IdP$
2. $SP \mid\equiv \xmapsto{K_P} SP$
3. $SP \mid\equiv \xmapsto{K_{S\to P}} SP$
4. $SP \mid\equiv IdP \Rightarrow +K_S$
5. $IdP \mid\equiv C \overset{(id,v)}{\leftrightharpoons} IdP$
6. $IdP \mid\equiv \#(Ts)$; $SP \mid\equiv \#(Ts)$
7. $IdP \mid\equiv C \Rightarrow r$

6.3 Proof

- C chooses $r \overset{\$}{\leftarrow} \{0,1\}^{l_0}$:

1. $C \mid\equiv \#(r)$
2. $C \mid\equiv r$ **(sub-goal 2.1.)**

- C sends $Auth\_req = Enc(pki, id||v||r)$ to IdP :

3. $IdP \triangleleft \{(id, v, r)\}_{K_I}$
   Assumption 1 and Formula 3 allow us to deduce the postulate : $\frac{IdP|\equiv \xmapsto{K_I} IdP, IdP \triangleleft \{X\}_{K_I}}{IdP \triangleleft X}$
4. $IdP \triangleleft (id, v, r)$
5. $IdP \triangleleft \langle r \rangle_{(id,v)}$
   Assumption5 and Formula 5 allow us to deduce the postulate : : $\frac{IdP|\equiv C \overset{Y}{\leftrightharpoons} IdP, IdP \triangleleft \langle X \rangle_Y}{IdP|\equiv C|\sim X}$
6. $IdP \mid\equiv C \mid\sim r$
7. $IdP \mid\equiv \#(r)$ *
   Formulas 6 and 7 are used to deduce the following postulate : $\frac{IdP\#(r), IdP|\equiv C|\equiv r}{IdP|\equiv r}$
8. $IdP \mid\equiv C \mid\equiv r$ **(sub-goal 2.2.)**
   Assumption 7 and Formula 8 allow us to deduce the postulate : $\frac{IdP|\equiv C \Rightarrow r, IdP|\equiv C|\sim r}{IdP|\equiv C|\equiv r}$
9. $IdP \mid\equiv r$ **(sub-goal 1.1.)**

- IdP sends $Auth\_token$ to C :

10. $C \triangleleft \{r, T_s\}_{K_S}$

- C chooses $k \overset{\$}{\leftarrow} \{0,1\}^{l_1}$.

11. $C \mid\equiv \#(k)$

- C sends $Auth\_resp = Auth\_token||Eph\_token$ to SP :

12. $SP \triangleleft (\{r, T_s\}_{K_S}, \{r, k\}_{K_P})$
13. $SP \triangleleft \{r, k\}_{K_P}$

    Assumption 2 and Formula 13 allow us to deduce the postulate : $\frac{SP|\equiv \xrightarrow{K_P} SP, SP \triangleleft \{X\}_{K_P}}{SP \triangleleft X}$
14. $SP \triangleleft (r, k)$
15. $SP \triangleleft \{r, T_s\}_{K_S}$

- The SP compute $Dec(K^{-1}{}_P, Re\text{-}Encrypt(K_{S \to P}, Auth\_token))$;

    Assumption 3 and Formula 15 allow us to deduce the postulate : $\frac{SP|\equiv \xrightarrow{K_{S\to P}} SP, SP \triangleleft \{\{X\}_{K_S}\}_{K_{S\to P}}}{SP \triangleleft \{X\}_{K_P}}$

16. $SP \triangleleft \{r, T_s\}_{K_P}$

    Assumption 2 and Formula 16 allow us to deduce the postulate : $\frac{SP|\equiv \xrightarrow{K_P} SP, SP \triangleleft \{X\}_{K_P}}{P \triangleleft X}$
17. $SP \triangleleft (r, T_s)$

    Assumption 6 allows us to deduce the postulate : : $\frac{SP|\equiv \#(T_s)}{SP|\equiv \#(r, T_s)}$
18. $SP \mid\equiv \#(r, T_s)$

    Assumption 4 and Formulas 15 and 17 allow us to deduce the postulate : $\frac{SP|\equiv IdP \Rightarrow +K_S, SP \triangleleft \{(r, T_s)\}_{K_S}, SP \triangleleft (r, T_s)}{SP|\equiv IdP|\sim (r, T_s)}$
19. $SP \mid\equiv IdP \mid\sim (r, T_s)$
20. $SP \mid\equiv IdP \mid\equiv (r, T_s)$
21. $SP \mid\equiv IdP \mid\equiv r$ **(sub-goal 1.2.)**

Thus, we find that the goals we defined were achieved since the sub-goals derived from it were all achieved.

6.4 Discussion

The main purpose of our protocol is to authenticate a user to a service provider. However, the BAN model does not allow us to present this objective directly. In order to interpret it through the BAN logic, we defined the elements that allow the user to be authenticated. The main element is the random value generated by the user. If the value received from the IdP by the user is the same that the one exchanged between the user and the IdP, we consider that the user has been identified by the IdP, that his authentication request has been validated by the IdP and is valid for the SP. This random value is never revealed in clear text but sent encrypted to the IdP and to the SP. This value can then be seen as a secret shared between C and the IdP and between the IdP and the SP. The same secret is not considered to be shared directly between C and SP since only the IdP can verify the identity of the user. Also, only the IdP can validate C's authentication request to the SP, then the definition of these two goals is: $SP \mid\equiv IdP \overset{r}{\leftrightharpoons} SP$ & $IdP \mid\equiv C \overset{r}{\leftrightharpoons} IdP$.

In order to facilitate the sequence of proof, the definition of sub-goals makes it possible to simplify the proof by reducing the number of deductions to be made, and by deriving the sub-objectives through the previously defined objectives. For example, in order to the SP believes that the exchange of the secret is true between him and the IdP, it is necessary that the IdP itself believes in the validity of the secret and that finally the SP believes that the IdP believes in this secret. Similarly, we have derived the sub-goals from the second objective.

After the goals and sub-goals were defined, we presented our assumptions. The two first concern the ownership of public keys by the different entities, namely the IdP and the SP. We also present the ownership of the re-encryption key by the SP, and the encryption key of the IdP for authentication messages to the SP. We also assume the existence of a secret shared between the user and the IdP. All our assumptions are directly related to the registration phase and the establishment of a trust relationship. Nevertheless, other hypotheses have been presented such as the trust of different entities to "timestamps", specifically for IdPs and SPs. For the last hypothesis, we assume that the value generated by the user is controlled by the user. Since this value is always associated with identifiers verifiable by the IdP then he believes that this value is really controlled by the user.

Now that the preliminary phases of the logical BAN have been established then, all that remains is to follow the evolution of the protocol. Using the different rules we have defined we can deduce the different sub-goals and thus verify that the protocol achieves the defined goals.

7 CONCLUSION

In this paper we give a brief review of some SSO and delegated authentication solutions. Then we introduced a new delegated authentication protocol based on the use of proxy re-encryption. It is considered as a single sign-on method. Thanks to this protocol a user can authenticate once, receive the *Auth_token* from the IdP and use any service that trust the user's IdP during the token's lifetime. Our method is simple and asynchronous. For the proof of security, we used the logical BAN method. This allowed us to model the protocol in the form of logical postulates and to prove that our protocol achieves the objectives that we have defined for it. Thus, we used the predefined rules of the logical BAN but also new rules that we integrated to adapt them to the use of the PRE. Authorization issues have not been taken into account in this work but have been left for futur study. Futur work will involve the examination of (Nunez et al., 2012) solution, which include an authorization layer to propose a complete authentication and access control framework. We will also implement the solution and instantiate it with several PREs in order to compare results.

REFERENCES

Alizadeh, M., Abolfazli, S., Zamani, M., Baharun, S., and Sakurai, K. (2016). Authentication in mobile cloud computing: A survey. *Journal of Network and Computer Applications*, 61:59–80.

Ateniese, G., Benson, K., and Hohenberger, S. (2009). Key-private proxy re-encryption. In *Cryptographers' Track at the RSA Conference*, pages 279–294. Springer.

Ateniese, G., Fu, K., Green, M., and Hohenberger, S. (2006). Improved proxy re-encryption schemes with applications to secure distributed storage. *ACM Transactions on Information and System Security (TISSEC)*, 9(1):1–30.

Benaloh, J. and De Mare, M. (1993). One-way accumulators: A decentralized alternative to digital signatures. In *Workshop on the Theory and Application of of Cryptographic Techniques*, pages 274–285. Springer.

Blaze, M., Bleumer, G., and Strauss, M. (1998). Divertible protocols and atomic proxy cryptography. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 127–144. Springer.

Burrows, M., Abadi, M., and Needham, R. M. (1989). A logic of authentication. *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, 426(1871):233–271.

Canetti, R. (2001). Universally composable security: A new paradigm for cryptographic protocols. In *Proceedings 42nd IEEE Symposium on Foundations of Computer Science*, pages 136–145. IEEE.

Canetti, R. and Hohenberger, S. (2007). Chosen-ciphertext secure proxy re-encryption. In *Proceedings of the 14th ACM conference on Computer and communications security*, pages 185–194. ACM.

Chaidos, P. and Couteau, G. (2018). Efficient designated-verifier non-interactive zero-knowledge proofs of knowledge. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 193–221. Springer.

Chow, S. S., Weng, J., Yang, Y., and Deng, R. H. (2010). Efficient unidirectional proxy re-encryption. In *International Conference on Cryptology in Africa*, pages 316–332. Springer.

Deng, R. H., Weng, J., Liu, S., and Chen, K. (2008). Chosen-ciphertext secure proxy re-encryption without pairings. In *International Conference on Cryptology and Network Security*, pages 1–17. Springer.

Gong, L., Needham, R. M., and Yahalom, R. (1990). Reasoning about belief in cryptographic protocols. In *IEEE Symposium on Security and Privacy*, pages 234–248. Citeseer.

Hardt, D., Bufu, J., and Hoyt, J. (2007). Openid attribute exchange 1.0-final. *at, Dec*, 5:11.

Ivan, A.-A. and Dodis, Y. (2003). Proxy cryptography revisited. In *NDSS*.

Nunez, D., Agudo, I., and Lopez, J. (2012). Integrating openid with proxy re-encryption to enhance privacy in cloud-based identity services. In *4th IEEE International Conference on Cloud Computing Technology and Science Proceedings*, pages 241–248. IEEE.

Sbai, A., Drocourt, C., and Dequen, G. (2019). Pre as a service within smart grid city. In *Proceedings of the 16th International Joint Conference on e-Business and Telecommunications - Volume 2: SECRYPT,*, pages 394–401. INSTICC, SciTePress.

Sbai, A., Drocourt, C., and Dequen, G. (2020). CCA Secure Unidirectional PRE with Key Pair in the Standard Model without Pairings. In *6th International Conference on Information Systems Security and Privacy*, pages 440–447, Valletta, Malta. SCITEPRESS - Science and Technology Publications.

Selvi, S. S. D., Paul, A., and Pandurangan, C. (2017). A provably-secure unidirectional proxy re-encryption scheme without pairing in the random oracle model. In *International Conference on Cryptology and Network Security*, pages 459–469. Springer.

Shao, J. and Cao, Z. (2009). Cca-secure proxy re-encryption without pairings. In *International Workshop on Public Key Cryptography*, pages 357–376. Springer.

Tremel, E. (2013). Real-world performance of cryptographic accumulators. *Undergraduate Honors Thesis, Brown University*.

Zhang, M., Wang, X. A., Li, W., and Yang, X. (2013). Cca secure publicly verifiable public key encryption without pairings nor random oracle and its applications. *JCP*, 8(8):1987–1994.