A Novel Architecture of a Strong and Mutual Authentication Protocol for Distributed Systems

Zakariae Tbatou, Ahmed Asimi, Chawki El Balmany, Younes Asimi, and Azidine Guezzaz

***Abstract*—With the prompt growth of distributed systems architectures, in particularly cloud computing, the authentication policy has becomes a crucial element for distributed communication. To ensure a secure access to data, numerous schemes have been designed to prevent listening, dictionary and intrusion attacks into stored password lists. These approaches remain relatively weak in terms of computer security; thus, they have defects on mutual authentication and they try to overcome their existing vulnerability.**

Our goal in this paper, is to enhance security in distributed systems, without affecting its performance. For this reason, we propose a new secure mutual authentication architecture for distributed systems, based on secure cryptographic primitives at the three communication entities involved (client, authentication server and n-servers of services), a consistent analysis regarding the complexity of our approach has been demonstrated with the BAN logic. It's composed of three main consists phases namely: 1)registration phase for secure exchange of authentication parameters, 2)communication phase aims to ensure mutual authentication of the three actors, based on secure cryptographic primitives and function ($S2KExS$) for key generation and 3)renewal phase to update the authentication parameters.

***Index Terms*—Distributed-systems, authentication-policy, computer-security, cryptographic-primitives, mutual-authentication.**

I. INTRODUCTION

THE present invention relates to new architectures such as distributed systems and cloud computing which treats a big data mass and ensures transparency for its users [1]. It can be divided into three layers related to security requirements: confidentiality [2], authentication [3] and access control [4], [5]. In practice, the trend of the proposed security protocols migrates to authentication without sending the password [6], [7], [8], [9] in a client/server environment, it appears in several techniques like cryptographic primitives [7], [10], the use of shared encryption keys between different actors [11], authentication by physical primitives such as smart card [12], [13] or biometrics [14], [10] or Electronic Records [15]. The main objective is to ensure a trustworthy exchange of the authentication parameters for each client (often its ID and password) at the server side, this requires a higher measures for the adoption of a communication channel and the storage of these parameters [2], [5], [16] during the registration phase. All these results show us that the authentication protocols without sending the password face several attacks such as brute force or man in the middle [6], [17], [9] and expose the robustness of these protocol. We mention some protocols like SSO, Kerberos and Timely authentication which are still the most used in practice by several companies and functional, compared to the different existing protocols. In our approach, we propose an authentication protocol by integrating an authentication server, that guarantees on one hand the management exchange of the encryption keys and on the other hand, a strong mutual authentication. This protocol is structured of three phases, as follow: The registration phase to exchange the authentication parameters based on a pseudo-random regenerator [18] for each user and per session to reinforce the password, and on a hash function. The communication phase guarantees a strong mutual authentication by the authentication server in both the client and server of the services side; by using the keys generations function ($S2KExS$)[19], [8] and the dynamic salt regenerator(RGSCG)[18] to guarantee non-traceable keys; and to narrow the message lifespan by using tickets principle. The renewal phase firstly requires the client authentication then the settings update. This phase is recommended after the registration phase for more securing communication channel by the use of associated session encryption keys.

Manuscript received February 06, 2019; revised April 20, 2019.

Z. Tbatou Asimi is with the Department of Mathematics and Computer Science, Faculty of Sciences, University Ibn Zohr, Agadir, Morocco. (Email: Tbatou.zakariae@gmail.com)

A. Asimi is with the Department of Mathematics, Faculty of Sciences, University Ibn Zohr, Agadir, Morocco. (Email: Asimi-ahmed2008@gmail.com)

C. El Balmany is with the Department of Mathematics and Computer Science, Faculty of Sciences, University Ibn Zohr, Agadir, Morocco. (Email: Chawki.elbalmany@gmail.com)

Y. Asimi is with the Department of Computer Science, superior School of Technology, University Ibn Zohr, Guelmim, Morocco. (Email: asimi.younes@gmail.com)

A. Guezzaz is with the Department of Computer Science, superior School of Technology, University Cadi Ayyad Zohr, Essaouira, Morocco. (Email: a.guzzaz@gmail.com)

II. RELATED WORK

With the extended use of distributed computer networks, it has become common to provide various accesses to users in several network services offered by distributed services providers. From the IT security view, the challenge for distributed systems is not only to support several security policies [4], [20], [21], [22]; but also the interaction between those different policies[23], [21]. However, the protection of such systems is a complex issue: indeed, in which entities of the system can we have confidence, and given this confidence, how to ensure the protection of the global system? As a result, user authentication plays a crucial role in distributed computer networks [24], [11], it verifies if a user has legal access or not to the requested services [18], [25], [17], [26]. In this sense, several techniques have been proposed to automate the configuration tasks management [27], [11], [28].Many techniques focus on the planning and implementation phases, where requirements and abstract security policies are designed [12], [23]. Among the authentication models, we mention KERBEROS, one of the most used protocols in distributed environments, based on the principle of Trusted Third Parties and the management of authentication keys.

TABLE I
NOTATIONS

C	Client.
B	Browser.
U	User.
AS	Authentication server.
ID_c	Client identity.
$@X$	Address of X.
SSi	Server of services.
S_j	Service.
$SS_{(i,Sj)}$	The S_j service of server SSi .
PW	Client password.
$A_{x,y,z}$	Authenticator generated by x to authenticate y to z.
KT	Temporary key regenerated during the registration phase.
K_{base}	Basic key.
K_{ia}	Initial mutual authentication key per session between C and AS.
K_{as}	AS secret key.
K_c	Associated client Key.
$K_{x,y}$	Shared key between x and y.
K_{pu}	Public key.
K_{pr}	Private key.
H	Hash function.
$S2KexS$	String to key extends salt function.
$RotDy$	Dynamic rotation function.
$Max()$	Function returns the maximum of two numbers.
LCM	Least common multiple.
$E_{K()}$	Encryption function using the key K.
$D_{K()}$	Decryption function using the key K.
T_{ex}	Expiration time of a ticket.
T	Clock time.
TGT	Ticket Garanting Ticket.
TS or TS'	Ticket Garanting Service.
HTTP	HTTP protocole.
HTTPS	HTTPS protocole.
$==$	Comparaison.
=	Affectation.
$\|\|$	Concatenation.

This protocol has proved its robustness in several analyzes against many attacks like brute force and man in the middle etc [29]. But, it is still a questionable domain, due to it represents a model known by its computational complexity and the lack of mutual authentication [8]. In fact, several approaches have been proposed to improve the functionality of this protocol without touching the basic principle; we mention the use of the public key [30], [9] and smart cards [13], [31]. Another technique also proposed is the single sign on (SSO) [32]. In this technique, a client can access to several services using one session translated by a prior authentication into its own authentication server, where the client is registered, this technique gives more agility to clients to access multiple services without re-authentication. The major problem of this technique, is that, it has not experienced many changes and does not guarantee mutual authentication, in other words does not provide original source of data between the client and all servers of services.
In this paper, we propose mutual authentication architecture for a distributed environment. It represents the security trend in distributed environments divided into three phases that guarantee, between different communication entities, a secure and faithful exchange of encryption keys.

III. OUR APPROACH

Our architecture (Fig. 1) has three main actors: a client identified by a unique ID_c and a password PW, an authentication server aims to generate encryption keys and n-servers of services supposedly decentralized.
As a first step, each client must register into the authentication server before communicating with the requested services. This phase is the most important one, because it allows the exchange and the storage of client authentication parameters at the AS side, for this reason, the use of the HTTPS protocol was required to ensure a safe and faithful exchange of authentication parameters which improves more robustness of our protocol.

A. Registration Phase

In a second step, the AS generates a single salt for each client. On the client side the browser must include

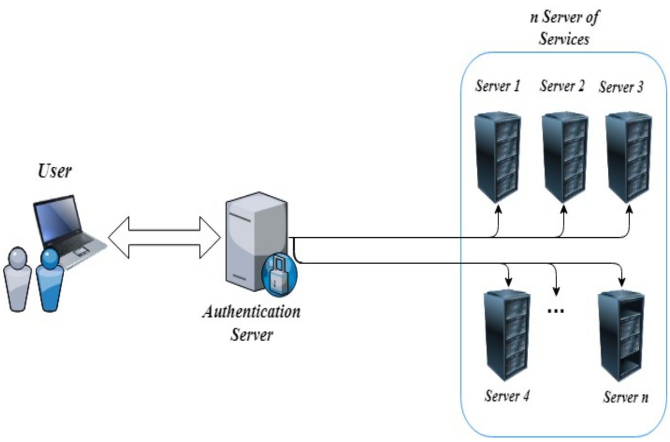

Fig. 1. Description of the distributed authentication architecture

cryptographic primitives and a pseudo random generator [18] for the salts regeneration in different phases. The registration phase (Fig. 2) is described as follows:

- The client sends his ID_c to AS.
- The AS checked client's ID_c:
  - If ID_c exists, it returns an error message.
  - Otherwise, the AS
    - ⋆ Generates a $salt\_AS$.
    - ⋆ Calculates a key KT which equals to the hash of ID_c concatenated with the $salt\_AS$.
    - ⋆ Sends the key KT to the client.
- The client:
  - Computes $M = H(RotDy(PW||H(PW)))$.
  - Encrypts M using the key KT: $C' = E_{KT}\{M\}$.
  - Sends C' to AS.
- Then, the AS:
  - Decrypts C' by KT to get M.
  - Stores ID_c and M.

B. Communication Phase

In this phase, each client registered at the AS must first authenticate to get access to the requested service. What's new in this phase, that the three players must authenticate to each other, which is to say in each communication, the client must authenticate his authentication server and vice versa. This translates the property of mutual authentication to define the source of data in our protocol, with a safe and faithful exchange of data during this phase. Indeed, the impact of mutual authentication is to reduce the probability of recovering the requests by a client and send falsified response requests, so the communication phase is subdivided into three parts:

1) Mutual authentication between client and AS; for the description, see Fig. 3.
2) Mutual authentication between AS and Server of services; for the description see Fig. 4.
3) Mutual authentication between client and Server of services; for description see Fig. 5.

1) Mutual Authentication between client and AS: The dialog in the first part of the communication phase between client and authentication server is described as follows:

- The client sends its ID_c to AS.
- The AS checks the existence of ID_c in the database:
  - If ID_c does not exist, the AS requests client registration.
  - Otherwise, it::
    - ⋆ Generates a new $Salt\_new$.
    - ⋆ Computes the key $K_{ia} = H((M||Salt\_new))$.
    - ⋆ Computes the basic key $K_{base} = S2KexS(K_{ia})$.
    - ⋆ Generates two keys $K_c = H(K_{base}||Salt\_new)$, $K_{as} = H(K_{base}||K_c)$.
    - ⋆ Encrypts K_{ia} and K_c using the key K_{ia} and TGT using K_{as}.
    - ⋆ Sends $E_{K_{ia}}\{K_c, K_{ia}\}, E_{K_{as}}\{TGT\}$ and $Salt\_new$ to client.
- The client:
  - Computes $M = H(RotDy(PW||H(PW)))$.
  - Computes $A = H(M||Salt\_new)$.
  - Decrypts $E_{K_{ia}}\{K_c, K_{ia}\}$ using A to obtain K_c and K_{ia}.
  - Compares K_{ia} with A :
    - ⋆ If $(K_{ia} == A)$, then the AS server is authenticated by the client, next it:
      - ∗ Generates an authenticator $A_{c,c,as}$ which contains the client authentication parameters and the requested service.
      - ∗ Sends $L = E_{K_c}\{A_{c,c,as}\}$ and $S = E_{K_{as}}\{TGT\}$) to AS.
    - ⋆ Otherwise, the client rejects the request and requests again the authentication of its AS server.
- The AS decrypts $S = E_{K_{as}}\{TGT\}$) using K_{as} and verifies TGT :
  - If TGT is expired, the AS requests the authentication of the client.
  - Otherwise, it:
    - ⋆ Decrypts $L = E_{K_c}\{A_{c,c,as}\}$ and recuperates $A_{c,c,as}$.
    - ⋆ Compares K_{ia} with A :
      - ∗ If $(K_{ia} == A)$, the mutual authentication is verified and the AS sends a broadcasts request for the service S_j to all servers of services.
      - ∗ Otherwise, it asks for re-authentication of the client.

2) Mutual authentication between AS and Server of Services: After the success of the first phase of mutual authentication between the client and the AS. The second part of the communication phase aims to ensure the mutual authentication property between the AS and all the servers of the services, that is, each time a client registered in the AS side, this latter requests a service, the AS authenticates one of the servers that contains this service, so the description of the communication phase between AS and service server (Fig. 4) is described as follows:

- The set of servers of services checks the availability of service S_j asked, if it is available, they send a response message.
- The AS selected @SSi address of the first response of services's server:
  - Generates $Salt - AS^*$.

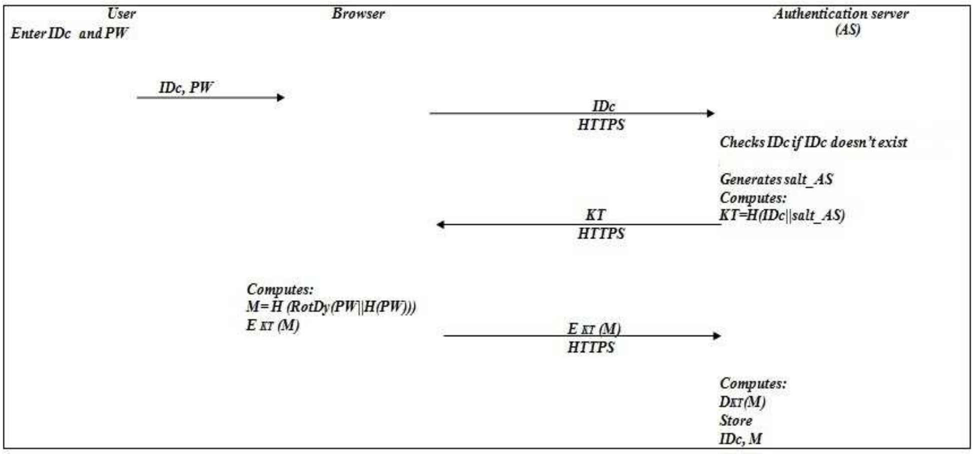

Fig. 2. Description of the registration phase

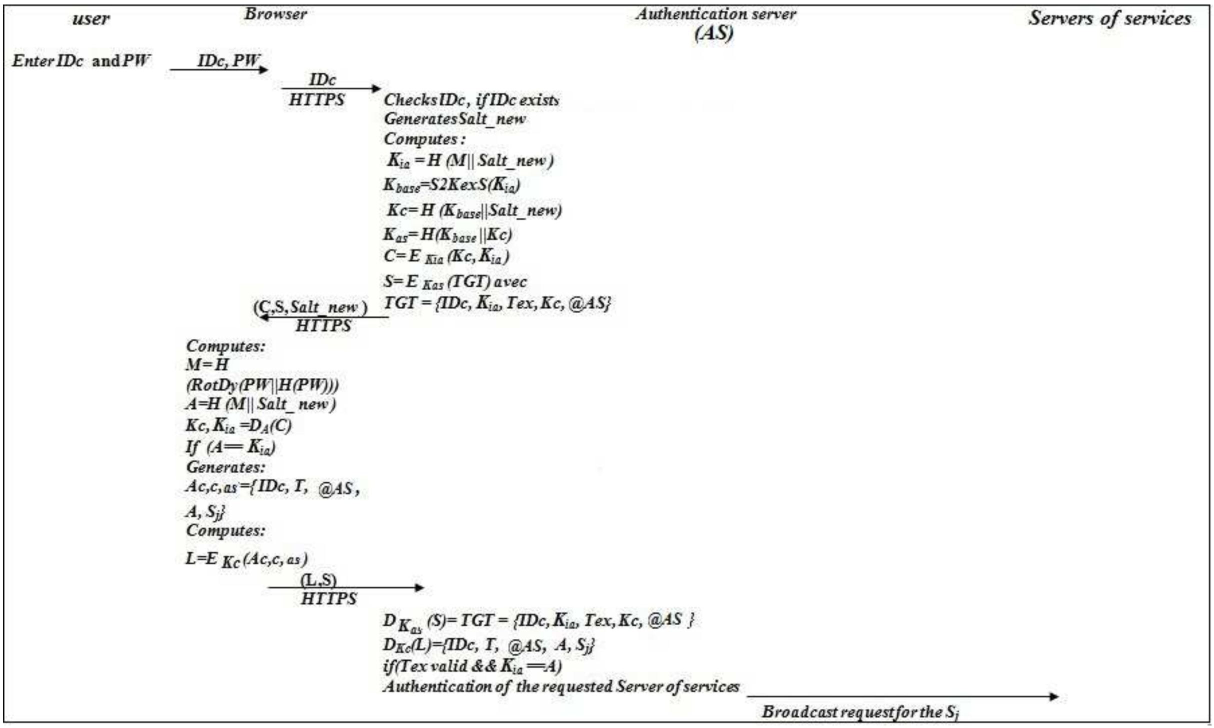

Fig. 3. Description of the communication phase between client and authentication server

- Computes N which equal to the sum of $Salt - AS^*$ bits.
- Computes M, which equal to the sum complementarily restricted to one of $Salt - AS^*$.
- Computes F which equal to the max between N and M.
- Encrypts $TS = \{@AS, Tex, @SSi, Salt - AS^*\}$ using K_{as}.
- Sends F and $E_{K_{as}}\{TS\}$ to the selected server of services SSi.

- Server of services SSi
  - Computes two keys: a public key K_{pu} and private key K_{pr} from F.
  - Computes $Z = E_{K_{pr}}\{@SSi, @AS, Service - name\}$
  - Sends K_{pu}, $E_{K_{as}}\{TS\}$ and Z to the authentication server AS.
- Authentication server AS:
  - Decrypts $E_{K_{as}}\{TS\}$ using K_{as}.
  - Checks the expiration time of the TS and the $Salt - AS^*$:
    - ⋆ If the time or $Salt - AS^*$ are invalid, the AS requests SSi to authenticate again.
    - ⋆ Otherwise, it :
      - ∗ Decrypts $Z = E_{K_{pr}}\{@SSi, @AS, Service - name\}$ using K_{pu} and get $@SSi$,$@AS$, Service-name.
      - ∗ If $@AS$, $@SSi$ and the service-name are correct, it:
        - ▷ Computes $K_{c,ssi} = H(K_{base}||K_{as})$ the key to share between the service server and the client.
        - ▷ Generates an authenticator $A_{as,C,SSi}$
        - ▷ Encrypts the key $K_{c,ssi}$ using the key

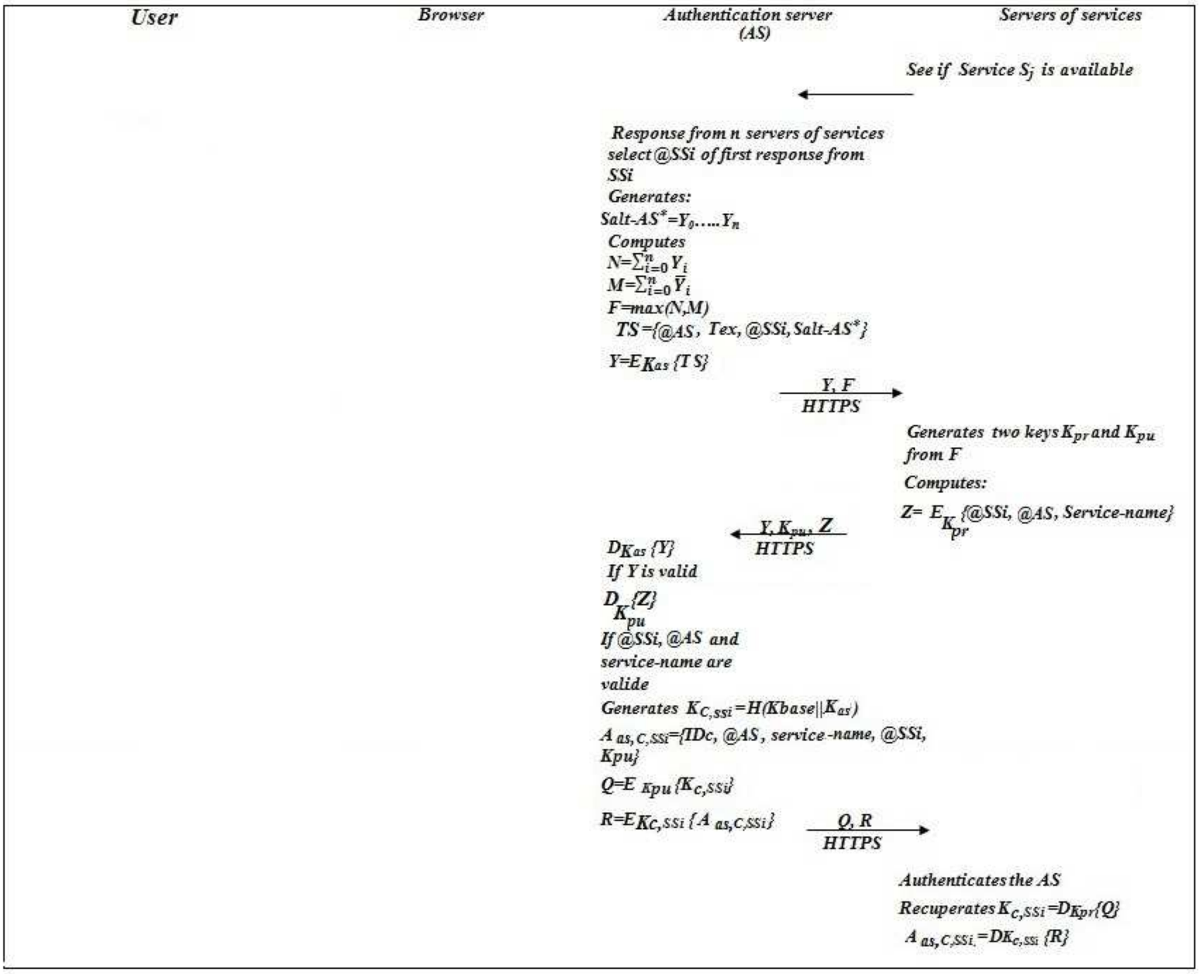

Fig. 4. Description of the communication phase between AS and server of services SSi

K_{pu} and encrypts the client authenticator $A_{as,C,SSi}$ using the key $K_{c,ssi}$.

▷ Sends $E_{K_{pu}}\{K_{c,ssi}\}$ and $E_{K_{c,ssi}}\{A_{as,C,SSi}\}$ to the server of services SSi.

∗ Otherwise, it requests SSi to authenticate again.

- Server of services SSi:
  - Decrypts $E_{K_{pu}}\{K_{c,ssi}\}$ using K_{pr} and obtains $K_{c,ssi}$.
  - Decrypts $E_{K_{c,ssi}}\{A_{as,C,SSi}\}$ using $K_{c,ssi}$ and recovers $A_{as,C,SSi}$.
  - Verifies if $@AS$ and K_{pu} of $A_{as,C,SSi}$ are valide. If this check is successful, then mutual authentication is guaranteed between SSi and AS, and SSi identifies the client via AS.

3) Mutual authentication between Client and Server of Services: At this level, the authentication server has shared the encryption keys at the client side and the servers of services, all that is remaining to the client is to send his authentication parameters and the requested service to the server of services so he can access to his desired services. The communication phase between the client and the service server means that the mutual authentication is satisfied between AS and the server of services (SSi), the communication phase between the client C and SSi (Fig. 5) is described as follows:

- The authentication server (AS):
  - Generates a ticket $TS' = \{@AS, Tex, @SSi, Salt-AS^*\}$ to manage access to services.
  - Encrypts the ticket TS' using the public key K_{pu}.
  - Sends $E_{K_c}\{K_{pu}, @SSi, K_{c,ssi}\}$ and $E_{K_{pu}}\{TS'\}$ to the client.
- Thee client:
  - Decrypts $E_{K_c}\{K_{pu}, @SSi, K_{c,ssi}\}$ using K_c and obtains $K_{c,ssi}$ and K_{pu}.
  - Encrypts its authenticator $A_{C,C,SSi}$ using $K_{c,ssi}$.
  - Sends $E_{K_{c,ssi}}\{A_{C,C,SSi}\}$ and $E_{K_{pu}}\{TS'\}$ to SSi.
- Server of services SSi:
  - Decrypts $E_{K_{pu}}\{TS'\}$ using its private key K_{pr} to obtain TS' .
  - Verifies $@AS$ and K_{pu} of TS'.
    - ⋆ If $@AS$ or K_{pu} are incorrect, it returns error message.
    - ⋆ Otherwise, it:
      - ∗ Decrypts $E_{K_{c,ssi}}\{A_{C,C,SSi}\}$ using $K_{c,ssi}$ to obtain $A_{C,C,SSi}$.
      - ∗ Compares $A_{C,C,SSi}$ and $A_{as,C,SSi}$. If $A_{C,C,SSi} == A_{as,C,SSi}$, it:
        - ▷ Computes $E_{K_{c,ssi}}\{S_j\}$.
        - ▷ Sends to client $E_{K_{c,ssi}}\{S_j\}$.

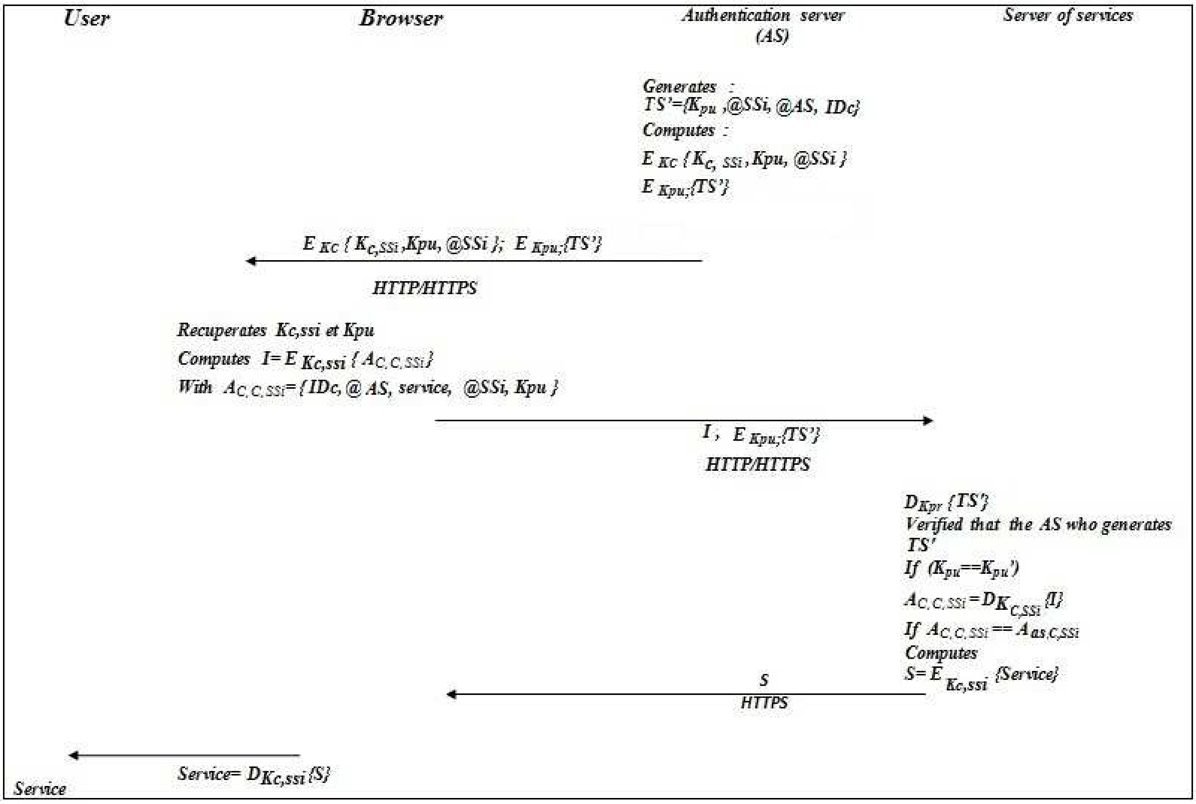

Fig. 5. Description of the communication phase between client and server of services

    * Otherwise, it sends a re-authentication request to the client.

- The client:
  - Decrypts $E_{K_{c,ssi}}\{S_j\}$ using $K_{c,ssi}$.
  - Obtains the requested service S_j.

4) Renewal phase: In this phase, the client updated its authentication parameters stored at the AS side; to note, that after the first communication and after the client authentication, we require this phase in our architecture to make the parameters stored at the AS side more safe. In this phase, each client must have a session opened in advance as described in the first part of the communication phase, after having exchanged the key shared between the client and the AS. The description of the renewal phase (Fig. 6) is described as follows:

- The client sends its ID_c to AS.
- The AS verifies if ID_c exists, then it:
  - Generates $Salt\_new$.
  - Computes the initial authentication key $K_{ia} = H(M||Salt\_new)$.
  - Computes the basic key $K_{base} = S2KexS(K_{ia})$.
  - Computes $K_c = H(K_{base}||Salt\_new)$.
  - Computes $K_{as} = H(K_{base}||K_c)$.
  - Encrypts K_c and K_{ia} using K_{ia}.
  - Generates a ticket TGT.
  - Encrypts TGT using K_{as}.
  - Sends $E_{K_{ia}}\{K_c, K_{ia}\}, E_{K_{as}}\{TGT\}$ and $Salt\_new$ to the client.
- Otherwise, the AS sends a re-authentication request.
- The client
  - Computes $M = H(RotDy(PW||H(PW)))$.
  - Computes $A = H(M||salt\_new)$.
  - Decrypts $E_{K_{ia}}\{K_c, K_{ia}\}$ using A and obtains K_c and K_{ia}.
  - Compares A with K_{ia}.
    - ⋆ If $(A == K_{ia})$, it :
      * Enters his new password PW_{new}.
      * Computes the new value $M' = H(RotDy(PW_{new}||H(PW_{new})))$.
      * Encrypts M' using K_c.
      * Sends $E_{K_c}\{M'\}$ to AS.
    - ⋆ Otherwise, it returns the error message.
- The authentication server (AS):
  - Decrypts $E_{K_{as}}\{TGT\}$ using K_{as}. If TGT is validated, it:
    - ⋆ Decrypts $E_{K_c}\{M'\}$ using K_c.
    - ⋆ Stores the new value of M'.
  - Otherwise, it sends an error message.

In our architecture, we aim to satisfy the mutual authentication property between different actors during a session, this makes our protocol more secure and reliable against different attacks that will be proved by the behavioral study of the keys management used, also, with a consistent analysis of complexity based on BAN logic to prove more the robustness of our scheme.

IV. BEHAVIORAL STUDY

In this part we will treat the keys used between the different entities to check the robustness of these keys. Each browser must support the cryptographic primitives used in our approach such as hash function and dynamic rotation function $RotDy$ [18]. For the servers of services side, they

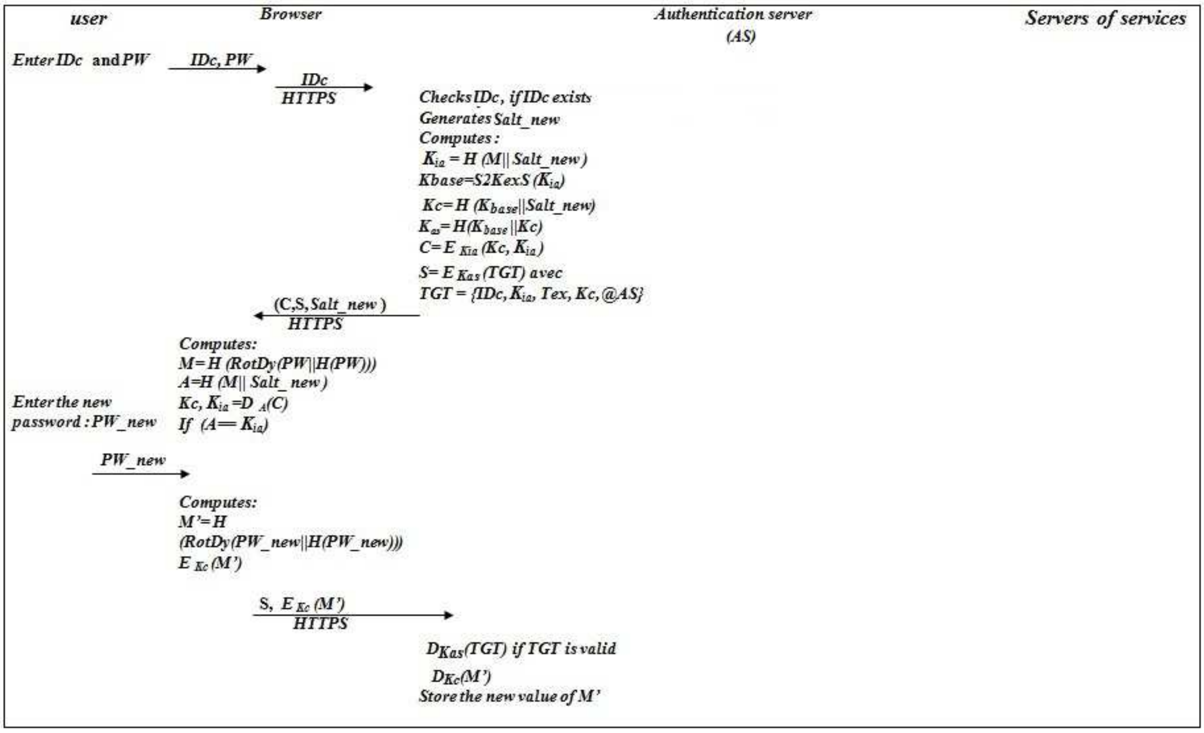

Fig. 6. Description of the renewal phase between client and authentication server

must adopt the authentication server extension that defines symmetric and asymmetric encryption primitives and the generation of public and private keys. In our case, we set up the management of the different keys used using PHP 5 to program the different functions as follow:

- DES CBC mode as a symmetric encryption algorithm.
- RSA as a asymmetric encryption algorithm.
- $SHA256$ as a hash function.
- RGSCS [18] as a dynamic salt regenerator.
- $RotDy$ function [18] as a dynamic rotation function.
- $S2KexS$ function [19], [8] for the generation of the basic key.

The hardware used in our experiments is a processor $1.3GHz$ CPU and $4GB$ of memory.

A. KEY MANAGEMENT

In order to verify the results of our approach, we have implemented it for three sessions and a given password and processed the results obtained as described in Fig. 7: We analyze all the keys regenerated by the AS server to see the robustness of these keys and we deduce the following results:

- The salt's impact per session makes the password untraceable.
- Using the $RotDy$ function makes the initial information opaque.
- The use of tickets enhances mutual authentication between the three entities and limit the dictionary attack by session expiration time
- The keys used are dynamic and per session.

1) NORMALIZED HAMMING DISTANCE: In our approach, session keys are dynamic, per session and have a variable size. Furthermore, in order to prove the robustness of the keys used in our approach, we admit the definition (4.1) and property (4.1):

Definition 4.1: Let S and S' be two binary sequences having successively the periods K and K' not necessarily equal. The Normalized Hamming Distance [33], named D, defined as follows:

$$D(S,S') = \frac{\sum_{i=0}^{k-1}((S(i\ mod\ K) + S'(i\ mod\ K'))\ mod\ 2)}{k} \tag{1}$$

with $k = LCM(K, K')$.

In order to prove the non-correlation between the keys used in our approach, Asimi et al [18] demonstrated that for S and S' two periodic binary strings are weakly correlated if $D(S, S') \simeq 0.5$; so we adopt the following property:

Proposition 4.1: Let S and S' be two periodic binary strings, we say that S and S' are weakly correlated if $D(S, S') \simeq 0.5$.

According to the figure, the definition (4.1) and the property (4.1), the results in figure Fig. 8 accumulate in the neighborhood of 0.5; which translate the non-correlation of the keys used during the communication phase even with a minimum perturbation for a given password. This means that the keys used even if they are derived from the same password, in several different sessions are totally independent of each other.

B. BAN LOGIC

BAN logic [34] was proposed in 1989 as a formal method for analyzing authentication protocols. As to describe a formal system, we first present the notations, then the deduction rules of the BAN logic. The logics of the BAN logic describe the concepts in the cryptographic protocols. The objective of this logic in our case is used to prove the robustness of mutual authentication and its impact on the entire protocol during the communication phase.

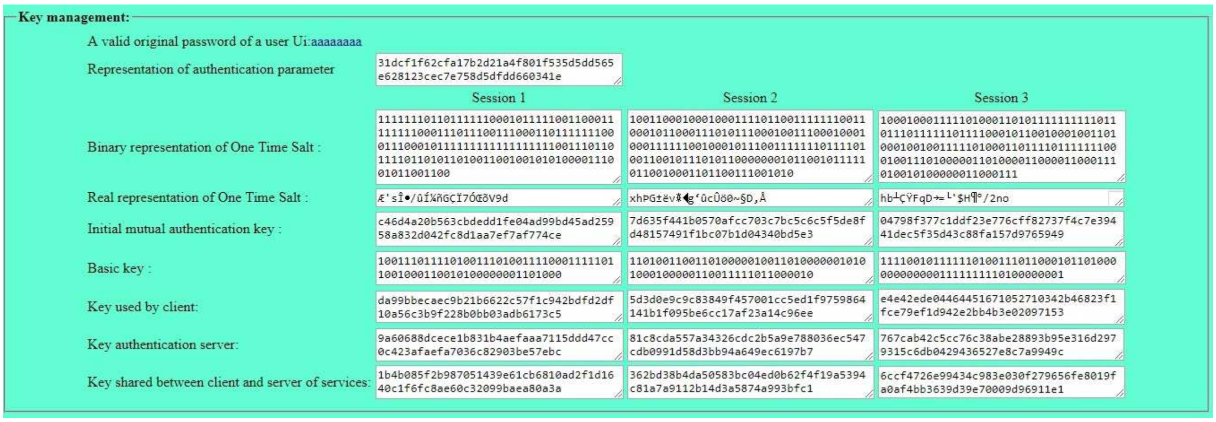

Fig. 7. Key management for three sessions with same password and different salts

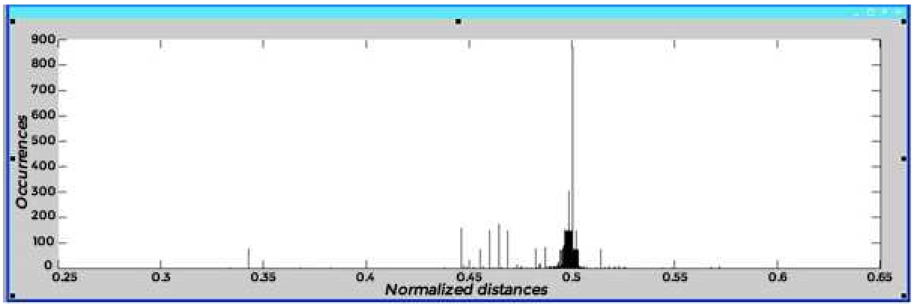

Fig. 8. Study of the non-correlation of the keys used in the communication phase

Notations of BAN logic:

$P\| \equiv X$	:	P believes in X.
$P \lhd X$	:	P sees X.
$P\| \sim X$	:	P once said X.
$\#(X)$	:	The formula X is fresh.
$P \overset{K}{\leftrightarrow} Q$	:	P and Q share the secret key K.
$\overset{K}{\rightarrow} P$	:	P has a public key K.
X_K	:	X is encrypted by the key K.
$X_{K^{-1}}$	:	X is encrypted by the public key K.
$< X >_Y$	:	X combined with Y, $X \parallel Y$.

Rule Definitions:

- Rule 1: $\dfrac{P| \equiv Q \overset{K}{\leftrightarrow} P, P \lhd \{X\}_K}{P| \equiv Q| \sim X}$ or $\dfrac{P| \equiv \overset{K}{\rightarrow} Q, P \lhd \{X\}_{K^{-1}}}{P| \equiv Q| \sim X}$
- Rule 2: $\dfrac{P \lhd (X,Y)}{P \lhd X}$ or $\dfrac{P \lhd < X >_Y}{P \lhd X}$
- Rule 3: $\dfrac{P| \equiv X, P| \equiv Y}{P| \equiv (X,Y)}$

Our goal is to prove the sharing of the key $K_{c,ssi}$ between B and SSi which translates mutual authentication implicitly between the three actors, which is translated by the following goals:

- Goal 1 : $B| \equiv SS_i \overset{K_{C,SSi}}{\rightleftharpoons} B$.
- Goal 2 : $SS_i| \equiv SS_i \overset{K_{C,SSi}}{\rightleftharpoons} B$.

In our protocol we divide the communication phase into three parts. To prove the mutual authentication and agreement of the session key we enumerate the verification objectives then we enumerate the idealized form transformed from the proposed scheme (Fig. 9):

Part one between browser and AS:

- Message 1 : $B \longrightarrow AS : ID_c$.
- Message 2 : $AS \longrightarrow B : (E_{K_{ia}}\{K_C, K_{ia}\}, E_{K_{as}}\{TGT\}, Salt\_new)$.
- Message 3 : $B \longrightarrow AS : (E_{K_C}\{A_{c,c,as}\}, E_{K_{as}}\{TGT\})$.

Part Two between AS and SSi:

- Message 4 : $AS \longrightarrow^{*} SS_i : (S_j)$.
- Message 5 : $SS_i \longrightarrow AS : \{@SS_i\}$.
- Message 6 : $AS \longrightarrow SS_i : (E_{K_{as}}\{TGT\}, F)$.
- Message 7 : $SS_i \longrightarrow AS : (K_{pu}, E_{K_{as}}\{TGT\}, E_{K_{pu}}\{@AS, @SSi, nom-service\})$.
- Message 8 : $AS \longrightarrow SS_i : (E_{K_{pu}}\{K_{C,SSi}\}, E_{K_{C,SSi}}\{A_{as,C,SSi}\})$.

Part three between B and SSi:

- Message 9 : $AS \longrightarrow B : (E_{K_C}\{K_{C,SSi}\}, E_{K_{pu}}\{TS'\})$.
- Message 10: $B \longrightarrow SSi : (E_{K_{C,SSi}}\{A_{C,C,SSi}\}, E_{K_{pu}}\{TS'\})$.
- Message 11: $SS_i \longrightarrow B : (E_{K_{C,SSi}}\{S_j\})$.

In our protocol, we have several exchanges of keys and tickets in the different phases to verify mutual authentication between different entities. In a session, the following assumptions are assumed to be true to improve our system if all hypotheses are verified. Therefore, the mutual authentication is satisfied.

Assumptions:

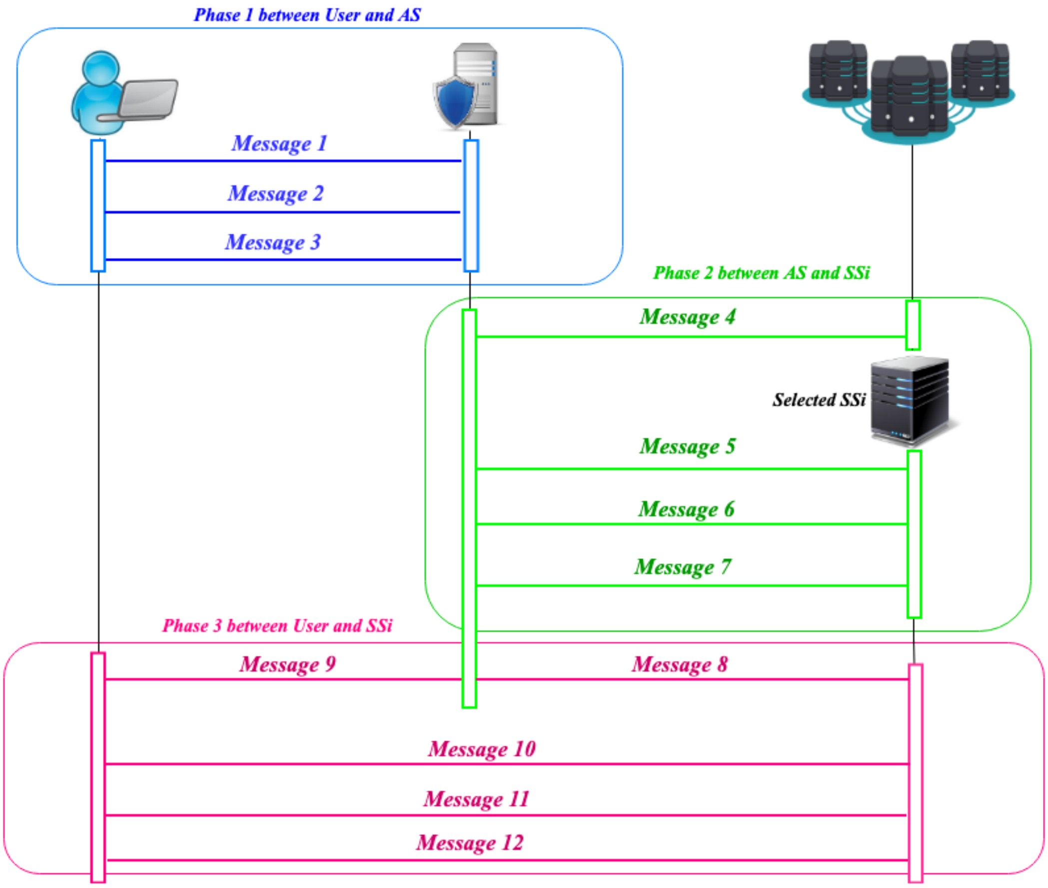

Fig. 9. Prototype for different phases

- $H1 : B| \equiv \#(Salt\_new)$.
- $H2 : AS| \equiv \#(TGT)$.
- $H3 : AS| \equiv \#(T)$.
- $H4 : \frac{AS| \equiv B \lhd K_c}{AS| \equiv B| \sim S_j}$.
- $H5 : \frac{AS| \equiv (TS)}{AS| \equiv \xrightarrow{K_{pu}} SSi}$.
- $H6 : \frac{SSi \lhd \{X\}_{K_{C,SSi}}}{SSi| \equiv AS| \sim X}$.

Verification: **Part One: Communication between client and AS**

• From message 2, we have $B \lhd (E_{K_{ia}}\{K_C, K_{ia}\}, E_{K_{as}}\{TGT\}, Salt\_new)$ According to rule 2, $B \lhd Salt\_new$ and $B| \sim PW$. According to $H1$ $B| \equiv \#(Salt\_new)$: B computes $A = H(M||Salt\_new)$, According to rule 2, we get $B \lhd (E_{K_{ia}}\{K_C, K_{ia}\})$, B decrypts $E_{K_{ia}}\{K_C, K_{ia}\}$ using A.
If $A == K_{ia}$ therefore: $B| \equiv AS| \sim (K_C, K_{ia})$, we deduce:

$$\textbf{Rule1}: \begin{cases} B| \equiv AS| \sim K_{ia} \\ B| \equiv AS| \sim K_C \end{cases}$$

According to the message 3 and rule 2 :

$$\begin{cases} AS \lhd E_{K_{as}}\{TGT\} \\ AS \lhd E_{K_c}\{A_{C,C,as}\} \end{cases}$$

• According to $H2$ $AS| \equiv \#(TGT)$. According to AS decrypts $E_{K_c}\{A_{C,C,as}\}$ using K_c. From rule 2 $AS \lhd A_{C,C,as} = (ID_c, T, @AS, A, S_j)$. According to $H3$ and rule 2, $AS \lhd A$.

• If $A == K_{ia}$, we deduce:

$$\textbf{Rule2} : \begin{cases} AS| \equiv B \lhd K_C \\ AS| \equiv B| \sim K_{ia}\} \end{cases}$$

As a result, from $Rule1$ and $Rule2$ we deduce the rule modeling mutual authentication between the client and the authentication server:

TABLE II
MUTUAL AUTHENTICATION (RULE1)

$AS| \equiv B| \sim K_{ia}, B| \equiv AS| \sim K_{ia}$

$$\textbf{Therefore, we conclude:} \begin{cases} AS| \equiv AS \overset{K_C}{\leftrightarrow} B \\ B| \equiv AS \overset{K_C}{\leftrightarrow} B \end{cases}$$

Part two: Communication between AS and server of services SSi

• According to message 5, $AS \lhd @SS_i$. According to message 6, $SSi \lhd (E_{K_{sa}}\{TS\}, F)$. According to rule 2, $SS_i \lhd F$, therefore, $SS_i| \sim K_{pu}$ and $SS_i| \sim K_{pr}$.
• According to message 7 and rule 2, $AS \lhd K_{pu}$, $AS \lhd E_{K_{as}}\{TS\}$ and
$AS \lhd E_{K_{pr}}\{@SSi, @AS, service - name\}$.
• AS decrypts $K_{pr}\{@SSi, @AS, service - name\}$ using K_{pu}.
• If all parameters are valid we deduce according to $H5$:

R'1 : $AS \rightarrow E_{K_{pu}}\{SSi\}$
• According to message 8, $SS_i \lhd (E_{K_{pu}}\{K_{C,SSi}\}, E_{K_{C,SSi}}\{A_{as,C,SSi}\})$. According to rule 1, $SS_i \lhd E_{K_{pu}}\{K_{C,SSi}\}$ and $SS_i \lhd E_{K_{C,SSi}}\{A_{as,C,SSi}\}$.
• According to rule 1, $SS_i \lhd K_{C,SSi}$ therefore, $SS_i \lhd A_{as,C,SSi} = (ID_c, @AS, service-name, @SSi, K_{pu})$. Then, According to rule 2, $SS_i \lhd K_{pu}$.
• According to rule 2 and $H6$, we deduce:

R'2 : $\dfrac{SS_i \lhd \{A_{as,C,SSi}\}_{K_{C,SSi}}}{SS_i| \equiv AS| \sim A_{as,C,SSi}}$

From **R'1** and **R'2**, the mutual authentication is verified and the rule modeling mutual authentication between the authentication server and the server of services is as follows:

TABLE III
MUTUAL AUTHENTICATION (RULE2)

$SS_i| \equiv AS \overset{K_{C,SSi}}{\longleftrightarrow} SS_i$

Part three: Communication between client and SS_i
• According to message 9, $B \lhd (E_{K_C}\{K_{C,SSi}\}, E_{K_{pu}}\{TS'\})$, thus according to rule 2 $B \lhd E_{K_C}\{K_{C,SSi}, K_{pu}, @SSi\}$ and $B \lhd E_{K_{pu}}\{TS'\}$. According to rule 1 we note $B \lhd E_{K_{c,ssi}}$
• From Rule 1 we deduce $B| \equiv AS| \sim K_{C,SSi}$
• According to the hypothesis $H5$ $B| \equiv AS \overset{K_{C,SSi}}{\longleftrightarrow} B$, and according to **M.A.Rule**1 et **M.A.Rule2** $B| \equiv AS \overset{K_{C,SSi}}{\longleftrightarrow} SSi$
• According to message 10, $SS_i \lhd (E_{K_{C,SSi}}\{A_{C,C,SSi}\}, E_{K_{pu}}\{TS'\})$, According to rule 2, $SSi \lhd E_{K_{C,SSi}}\{A_{C,C,SSi}\}$ and $SS_i \lhd E_{K_{pu}}\{TS'\}$
• We have $SS_i \lhd K_{C,SSi}$ according to Rule 2, therefore, according to rule 1 and $H4$ $SS_i \lhd A_{C,C,SSi} = \{ID_c, @AS, S_j, @SS_i, K_{pu}\}$. therefore, according to rule 2, $SS_i \lhd K_{pu}$

Since Rule 3, and "Mutual Authentication (Rule 2)" III, we deduce the mutual authentication rule between the server of services and the authentication server:

TABLE IV
AUTHENTIFICATION MUTUELLE (RULE3)

$SSi| \equiv AS| \sim TS', SSi| \equiv B| \sim K_{pu}$

• Then $SS_i| \equiv B| \sim S_j$ and $SS_i| \equiv AS \overset{K_{C,SSi}}{\longleftrightarrow} B$.
• According to the message 11, $B \lhd E_{K_{C,SSi}}\{S_j\}$, and rule 1, $B \lhd S_j$.

According to rules (II, III and IV) which define the mutual authentication between the three actors in different communication phases we conclude from the demonstration of the BAN logic that the mutual authentication aims to prove the source of the data exchanged in each step as well, the effectiveness of the protocol at the keys management level by treating each request sent to the network. The obtained results based on well-defined rules and solid assumptions prove the utility of mutual authentication in order to guarantee a secure exchange. Indeed, the addition of bi-directional authentication features with a limited expiration time of ticket in a session ensures the identification of each actor in different communication steps. in addition, the impact of dynamic and per session salts makes the whole protocol dynamic and per session, which limits the chance that information such as passwords imprint or even the requested service will be guessed by the exchanged information during a session, therefore according to these studies we grant the following remarks which are based on (support) the results proved previously:

- Mutual authentication is an obligation in distributed environments.
- The policy package must be targeted before adding the mutual authentication mechanism, without perturbing the full functionality of the used protocols.
- Despite the complexity of the architecture of distributed systems, mutual authentication improves the robustness of all the policies used.
- The use of a dynamic protocol per session reduces the chance of perceiving the exchanged entities, such as the identification parameters.

V. SECURITY ANALYSIS

Typically, a client process relies on an authentication protocol before grant his access. Most authentication protocols are based on passwords or information derived from it. In most architecture that define distributed systems, the set of systems must guarantee the transparency and integrity of data sent to the client, we mention in this case, the use of protocols such as SSO [32] and Kerberos [29]. Whereas, with the development of those architectures which becomes more and more complicated, mutual authentication becomes a criterion required to check the source of the data [12]. In our approach, we have targeted this requirement by keeping the notions of transparency and integrity of the data sent and dealing with different attacks that still remain a real threat and affect the functionality of this type of system.

A. Salt impact

Most IT systems that have one or more processes are accessible by client. A process authenticates the client by comparing information derived from his password [6], [17] that are exchanged during the registration phase. Adding an often static salt increases the chance of guessing the password that represents the core of the authentication process [12]. In our approach we used RGSCS, wich is a dynamic and cryptographically safe salt generator [18]. The point is to perturb the original password, and diminishes the chance of finding original information especially by dictionary attack, because the salt is renewed at each session and regenerated for each user.

B. Ticket impact

The three main types of authentication in a distributed computer system are message content authentication, message origin authentication, and identity authentication of message transmitter. The use of tickets principle makes our protocol safer. It represents the entity generated by the authentication server and encrypted with its own key. It contains the authentication parameters with a limited expiration time in purpose to reduce the chance of falsification of these settings. Additionally, it provides mutual authentication in all three parts of the communication phase.

C. Robustness against man in the middle

In this technique, the attacker should be able to observe and intercept (Sniffing) the encrypted data exchanged between two entities during a communication in a valid time. In our protocol, to cope with this attack, we define the mutual authentication property, one side by using symmetric cryptographic primitives with robust and dynamic keys. The other side by adding tickets to define the source of the messages. Moreover, limiting life of the data transmitted at the session time reduces the chance to find out the origin of the message.

D. Robustness against Dictionary Attack

This type of attacks is very effective in case of authentication systems based on breakable hash functions. This attack represents a real threat to the distributed architectures cited in several analyses [35], [29]. In our system, the cryptographic quality of passwords is strongly related to user-appropriate salts, dynamic rotation, and irreversible hash function. For an attacker, the chance to find the password from the messages exchanged during a session is too low especially with the impact of the tickets which reduces the lifetime of the exchanged messages.

E. Robustness against brute force

This attack depends, in general, on the resistance of the passwords also to their complexity. The principle is to launch software to test the possible combinations of the passwords. In our approach, the attacker does not only need to find the hash of the password, but he must guess the original password from the $M = H(RotDy(PW||H(PW)))$, it is very difficult considering the limited session time, moreover, the dynamic rotation applied on the password guarantees the non-correlation between the original passwords. So we confirmed the unpredictable nature of passwords makes our architecture secure against brute force attacks.

F. Comparison between Our Protocol and other authentication protocols

Our protocol, designed for distributed systems, aims to ensure the confidential exchange and mutual authentication between clients, authentication servers and services server. For these reasons, our approach is based on tickets, key management and other functions namely: $S2KexS$ function, $RotDy$ function. The table (V) gives a comparison between our approach and other protocols used in distributed architecture such as Kerberos $V5$ [13], SSO [32], two factors authentication protocols [25] and timely authentication protocols [25]:

VI. CONCLUSION

With the development of computer systems, the distributed architectures have proven their functionality to make the system decentralized to the level of data and services; while keeping the notion of transparency. Despite to the development of computers, attacks have become more and more efficient [16], [36] which requires a secure and faithful authentication protocol [37], [38]. Several works have discussed the requirement of authentication such as the use of SSO [33] or certificates [7], but in this type of complicated architecture, it is necessary to prove the source of the data exchanged between the different entities to reduce the chances of affecting the confidentiality and integrity of the stored and exchanged data. In our work, we have implemented the problem of mutual authentication in distributed systems with the proposal of a protocol that aims to achieve a mutual authentication between different entities; the addition of tickets reduces the chance of breaking the functionality of the protocol by the expiration time and by the robustness of the keys used in different phases, which is proven in the behavioral study and BAN logic.

REFERENCES

[1] I. Odun-Ayo, C. Okereke, and H. Orovwode, “Cloud computing and internet of things: Issues and developments,” in *Proceedings of the World Congress on Engineering*, vol. 1, 2018.

[2] F. Bao, “Security analysis of a password authenticated key exchange protocol,” in *International Conference on Information Security*. Springer, 2003, pp. 208–217.

[3] S. Gritzalis, D. Spinellis, and P. Georgiadis, “Security protocols over open networks and distributed systems: Formal methods for their analysis, design, and verification,” *Computer Communications*, vol. 22, no. 8, pp. 697–709, 1999.

[4] D. Hindawi, O. Hindawi, L. Lippincott, and P. Lincroft, “System, security and network management using self-organizing communication orbits in distributed networks,” Jan. 26 2016, uS Patent 9,246,977.

[5] C.-C. Lee, C.-H. Liu, and M.-S. Hwang, “Guessing attacks on strong-password authentication protocol.” *IJ Network Security*, vol. 15, no. 1, pp. 64–67, 2013.

[6] M. Abdalla, P.-A. Fouque, and D. Pointcheval, “Password-based authenticated key exchange in the three-party setting,” *IEE Proceedings-Information Security*, vol. 153, no. 1, pp. 27–39, 2006.

[7] B. Lampson, M. Abadi, M. Burrows, and E. Wobber, “Authentication in distributed systems: Theory and practice,” *ACM Transactions on Computer Systems (TOCS)*, vol. 10, no. 4, pp. 265–310, 1992.

[8] Z. Tbatou, A. Asimi, Y. Asimi, Y. Sadqi, and A. Guezzaz, “A new mutuel kerberos authentication protocol for distributed systems.” *IJ Network Security*, vol. 19, no. 6, pp. 889–898, 2017.

[9] R. Tso, “Security analysis and improvements of a communication-efficient three-party password authenticated key exchange protocol,” *The Journal of Supercomputing*, vol. 66, no. 2, pp. 863–874, 2013.

[10] H. R. Talkhaby and R. Parsamehr, “Cloud computing authentication using biometric-kerberos scheme based on strong diffi-hellman-dsa key exchange,” in *Control, Instrumentation, Communication and Computational Technologies (ICCICCT), 2016 International Conference on*. IEEE, 2016, pp. 104–110.

[11] C. Lin and V. Varadharajan, “Trust based risk management for distributed system security-a new approach,” in *null*. IEEE, 2006, pp. 6–13.

[12] X. Huang, Y. Xiang, A. Chonka, J. Zhou, and R. H. Deng, “A generic framework for three-factor authentication: Preserving security and privacy in distributed systems,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 8, pp. 1390–1397, 2011.

[13] N. Mavrogiannopoulos, A. Pashalidis, and B. Preneel, “Toward a secure kerberos key exchange with smart cards,” *International journal of information security*, vol. 13, no. 3, pp. 217–228, 2014.

[14] S. Sarah and P. Pamela, “Secure transmission by employing multi-biometric authentication in cloud computing,” *International journal of rend in Research and Development (IJRD)*, vol. 3.

[15] W. R. Simpson and K. E. Foltz, “Digital key management for access control of electronic records.” *IAENG International Journal of Computer Science*, vol. 43, no. 4, pp. 411–426, 2016.

[16] G. Notoatmodjo and C. Thomborson, “Passwords and perceptions,” in *Proceedings of the Seventh Australasian Conference on Information Security-Volume 98*. Australian Computer Society, Inc., 2009, pp. 71–78.

[17] Y. Sadqi, A. Asimi, and Y. Asimi, “A cryptographic mutual authentication scheme for web applications,” *arXiv preprint arXiv:1412.2908*, 2014.

TABLE V
COMPARISON BETWEEN OUR PROTOCOL AND OTHER PROTOCOLS

	Our protocol	Kerberos V5	SSO	Two factors Authentication	Timely authentication
Key management	OK	OK	OK	OK	OK
Synchronized system	OK	OK	-	-	-
Dynamic Key	OK	Based on password	Based on certificates	OK	Based on password
Dynamic salt per session	OK	-	-	-	-
mutual authentication	OK	-	-	-	-
strong authentication	OK	-	-	OK	-

[18] Y. Asimi, A. Amghar, A. Asimi, and Y. Sadqi, "New random generator of a safe cryptographic salt per session." *IJ Network Security*, vol. 18, no. 3, pp. 445–453, 2016.

[19] Z. Tbatou, A. Asimi, Y. Asimi, and Y. Sadqi, "Kerberos v5: Vulnerabilities and perspectives," in *Complex Systems (WCCS), 2015 Third World Conference on*. IEEE, 2015, pp. 1–5.

[20] M. Moriconi and S. Qian, "System and method for maintaining security in a distributed computer network," Dec. 5 2000, uS Patent 6,158,010.

[21] M. Satyanarayanan, "Integrating security in a large distributed system," *ACM Transactions on Computer Systems (TOCS)*, vol. 7, no. 3, pp. 247–280, 1989.

[22] W. R. Simpson and K. E. Foltz, "Ports and protocols extended control for security." *IAENG International Journal of Computer Science*, vol. 44, no. 2, pp. 227–240, 2017.

[23] D. M. Nessett, "Factors affecting distributed system security," *IEEE Transactions on Software Engineering*, no. 2, pp. 233–248, 1987.

[24] M. Lorch, S. Proctor, R. Lepro, D. Kafura, and S. Shah, "First experiences using xacml for access control in distributed systems," in *Proceedings of the 2003 ACM workshop on XML security*. ACM, 2003, pp. 25–37.

[25] M. Pourzandi, D. Gordon, W. Yurcik, and G. A. Koenig, "Clusters and security: distributed security for distributed systems," in *Cluster Computing and the Grid, 2005. CCGrid 2005. IEEE International Symposium on*, vol. 1. IEEE, 2005, pp. 96–104.

[26] V. Varadharajan, P. Allen, and S. Black, "An analysis of the proxy problem in distributed systems," in *Research in Security and Privacy, 1991. Proceedings., 1991 IEEE Computer Society Symposium on*. IEEE, 1991, pp. 255–275.

[27] W. G. Doyen, T. K. Ryan, T. Harper, K. W. Traub, C. Rausch, and K. Hawver, "Managing personnel access employing a distributed access control system with security enhancements for improved user awareness to aid in decision making," Apr. 26 2016, uS Patent 9,324,205.

[28] Q. Liu, Z. Wang, X. He, and D. Zhou, "A survey of event-based strategies on control and estimation," *Systems Science & Control Engineering: An Open Access Journal*, vol. 2, no. 1, pp. 90–97, 2014.

[29] T. D. Wu, "A real-world analysis of kerberos password security." in *Ndss*, 1999.

[30] H. Li, X. Lin, H. Yang, X. Liang, R. Lu, and X. Shen, "Eppdr: an efficient privacy-preserving demand response scheme with adaptive key evolution in smart grid," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 8, pp. 2053–2064, 2014.

[31] R. Song, "Advanced smart card based password authentication protocol," *Computer Standards & Interfaces*, vol. 32, no. 5-6, pp. 321–325, 2010.

[32] J. De Clercq, "Single sign-on architectures," in *Infrastructure Security*. Springer, 2002, pp. 40–58.

[33] Y. Asimi, A. Amghar, A. Asimi, and Y. Sadqi, "Strong zero-knowledge authentication based on the session keys (sask)," *International Journal of Network Security & Its Applications*, vol. 7, no. 1, p. 51, 2015.

[34] A. Bleeker and L. Meertens, "A semantics for ban logic," in *Proceedings of the DIMACS Workshop on Design and Formal Verification of Security Protocols*, 1997.

[35] S. M. Bellovin and M. Merritt, "Encrypted key exchange: Password-based protocols secure against dictionary attacks," in *Research in Security and Privacy, 1992. Proceedings., 1992 IEEE Computer Society Symposium on*. IEEE, 1992, pp. 72–84.

[36] I. Odun-Ayo, O. Ajayi, and S. Misra, "Cloud computing security: Issues and developments," in *Proceedings of the World Congress on Engineering*, vol. 1, 2018.

[37] W. R. Simpson and K. E. Foltz, "Insider threat metrics in enterprise level security." *IAENG International Journal of Computer Science*, vol. 45, no. 4, pp. 610–622, 2018.

[38] M. Spremić and A. Šimunic, "Cyber security challenges in digital economy," in *Proceedings of the World Congress on Engineering*, vol. 1, 2018, pp. 341–346.