An Enhanced Linkable Anonymous Access Protocol of the Distributed Electronic Patient Records

Citation for published version (APA):
Addas, R., & Zhang, N. (2014). An Enhanced Linkable Anonymous Access Protocol of the Distributed Electronic Patient Records. In *The 9th International Conference on Availability, Reliability and Security (ARES 2014)* (pp. 146-151). IEEE.

Published in:
The 9th International Conference on Availability, Reliability and Security (ARES 2014)

An Enhanced Linkable Anonymous Access Protocol of the Distributed Electronic Patient Records

Rima Addas
School of Computer Science
University of Manchester
Manchester, UK
addasr@cs.man.ac.uk

Ning Zhang
School of Computer Science
University of Manchester
Manchester, UK
nzhang@cs.man.ac.uk

***Abstract*—This paper describes the growing concern of privacy and security in e-health applications. Sharing sensitive patient data in a distributed environment introduces security and privacy risks. Therefore, there are increasing demands to provide secure access to distributed Electronic Patient Records (EPRs) but without compromising performance. The aim of this paper is to respond to such demands and to support secure and efficient access to distributed EPRs. In this paper, we enhance the Linkable Anonymous Access Protocol while supporting security and performance. To achieve this, we have designed a secure protocol called the Enhanced Linkable Anonymous Access (ELAA) protocol. To show that the ELAA protocol is secure and efficient, (1) we formally verify and analyse it against security properties using the Casper/FDR2 verification tool. In addition, (2) we build a prototype using the Java technology to demonstrate the performance of the enhanced protocol. By doing this, we prove that the ELAA protocol maintains a good balance between security and performance while supporting distributed access to EPRs.**

Keywords-e-Health; electronic patient records; privacy; security; performance.

I. Introduction

The transition from traditional paper-based healthcare to electronic-based healthcare have led to improving the quality of patient health care. The latter has a number of advantages. Among them we note the greater convenience and speed to access health data, which translates into shorter treatment delays, less medical errors, better statistics, higher cost-efficiency, better fraud detection mechanisms, and shorter refund delays for patients covered by health insurance plans [1].

Despite all the above benefits, patients have shown reluctance towards new electronic healthcare systems. The reason for this reluctance is mainly due to the lack of assurances about the way patient data is handled. Privacy and security concerns are major barriers in ehealth applications. If they are not properly considered, patients will not be confident to share their sensitive data and health Service providers (HSPs) will face huge burden risks [2]. One of the main threats to privacy in an ehealth application occurs from the secondary use of health information. Secondary use concerns those scenarios where information revealed to one party for a particular purpose is subsequently used for other purposes, without the authorization of the data subject. The chances for privacy invasions due to secondary use by insiders are vast. Studies have confirmed that the most frequent breaches of patient information confidentiality do not come from unauthorized outsiders, but from uncontrolled secondary use, accidental disclosures, curiosity, and subordination by insiders [3].

To help reduce this lack of trust, ehealth protocols should be designed in a way that both security and privacy are considered. Due to the sensitive nature of health data, such protocols should be based on well established cryptographic techniques, and should provide rigid defences against possible attacks. Attackers can eavesdrop, modify and delete the health care messages in communication between HSPs and patients. In addition, if the identities of the patients are exposed to attackers who can record and track the patients privacy information, the attackers can sell this private information. In other words, both security and privacy breaches can cause serious legal and financial consequences [4].

In real-life situations, there are several scenarios, where authorized users have legitimate reasons to access patients' distributed EPRs. Based on the principle of least privilege, users should only be granted access rights that are just sufficient for them to carry out the tasks assigned to them. The minimum level of access privilege is allowing users to access anonymous or de-identified records. De-identification means that patients' identifiable information is removed from the records [5]. Pseudonymization is one of the de-identification methods. Pseudonymization is the process by which all personal identities within a data record are replaced by an artificial identifier. The artificial pseudonym allows tracking back of data to its origins from anonymized data where all person-related data has been removed [6]. In practice, there are times when, for legitimate reasons, multiple de-identified records of the same patient may need to be linked (e.g., when we need to study the history of a patient's medical condition) or an anonymised record needs to be re-identified at a later date. In such cases, a patient's pseudonym should be mapped or reversed to the patient's identity and two or more pseudonyms of the same patient should be linkable and these should be done in a controlled manner.

This paper is organized as follows. In Section 2 we present the related work. In Section 3, we describe, model and verify the ELAA protocol. Also, we set the security requirements that the ELAA protocol should meet. After that, we show the result of the verification. In Section 4, we present the implementation and performance analysis of the ELAA protocol. In Section 5, we conclude the paper and discuss future work.

II. Related Work

In the literature, to de-identify (or anonymise) patients' records, two types of pseudonyms can be, namely, irreversible and reversible pseudonyms [7]. Irreversible pseudonyms are pseudonyms that cannot be reversed back to the patient's real identity. They are called one-way pseudonyms. Reversible pseudonyms are pseudonyms that can be reversed back to the original identity. That is a patient can be re-identified from his/her reversible pseudonyms. These pseudonyms are called two-way pseudonyms. Most pseudonym generation solutions used in supporting privacy preserving EPR access [8][9][10], consider preserving patient anonymity. They rely on irreversible pseudonyms to index de-identified records. This type of pseudonyms only supports anonymous data access. Though the pseudonym generation methods in [7][11], have considered the linkability requirement, they do not support the secondary use of patient information. That is, they do not allow linking of multiple pseudonyms of the single patient without revealing the patient's identity. A notable method that has addressed this limitation is LIPA [12]. Yet, LIPA supports this linkablity requirement, but assuming that patient records managed by different HSPs are stored in a single repository. The solution does not support distributed data access. To the authors' best knowledge, the works that are most related to ours are Deng's [13] and the PIPE [14] methods. Both methods aim to securely integrate primary and secondary use of distributed medical data without compromising the patient's identity privacy. We described an alternative method in [15] with the aim to reduce access delays. In other words, our method proved to be more efficient than Deng's and PIPE methods.

In detail, to facilitate the minimum access right management, we have proposed a new method called 3LI2Pv2 method to support controlled access to EPRs with three levels of identity privacy reservations [15]. In this method, we have identified three different user groups, each with a defined level of access. The first group of users (L3 users) are only given rights to access anonymised data. They are not allowed to identify the patient nor link multiple EPR objects of the same patient. The second group of users (L2 users) are allowed to access and link multiple objects of the same patient, but are not allowed to link the objects to their owner's (i.e., the patient's) identity. In other words, users in this group are allowed to access the multiple objects of the single patient without being able to identify the patient. Finally, the third group of users (L1 users) are allowed to access patients' records as well as identify the owners of the records. To summarise, we have identified in [15] three levels of patient identity privacy protection.

- *Level-1 (L1)- Linkable access:* At this level, multiple data objects of the same patient can be linked, and this set of objects can be linked to the patient's identity. L1 access should be limited to L1 users, i.e., users with linkable access privilege.
- *Level-2 (L2)- Linkable anonymous access:* At this level, multiple data objects of the same patient can be linked, but this set of objects cannot be linked to the patient's identity. L2 access should be limited to L1/L2 users, i.e., users with linkable anonymous access privilege.
- *Level-3 (L3)- Anonymous access:* At this level, multiple data objects of the same patient cannot be linked, nor the patient's identity be exposed. L3 access should be limited to L1/L2/L3 users, i.e., users with anonymous access privilege.

In this paper, we focus on the second level, the L2-Linkable anonymous access. We introduce an enhanced protocol to the one presented in [16]. We call it the Enhanced Linkable Anonymous Access (ELAA) protocol. In the enhanced protocol, we allow users not only to perform linkable anonymous access on a patient's objects managed by a single HSP (as was the case in the Linkable Anonymous Access (LAA) protocol [16]), but we expand their access to allow them to link distributed anonymised patient objects but managed my multiple HSPs.

In this paper, Casper/FDR2 verification tool [17][18], is also used to verify the ELAA protocol, as it has been used in verifying the LAA protocol in [16]. Casper/FDR2 has proven to be successful for modelling and verifying several security protocols [19]. Accordingly, we consider it also appropriate for the verification of the ELAA protocol. After completing the formal verification of the ELAA protocol using Casper/FDR2, we implement the protocol using the Java technology [20] to evaluate its performance.

III. Formal Verification of The ELAA Protocol

In this section, firstly, we describe and model the ELAA security protocol with Casper/FDR2 verification tool. Secondly, we identify essential security requirements that the ELAA protocol should fulfil. Finally, we discuss the verification result of the protocol and analyse its security requirements.

A. The ELAA Protocol Description

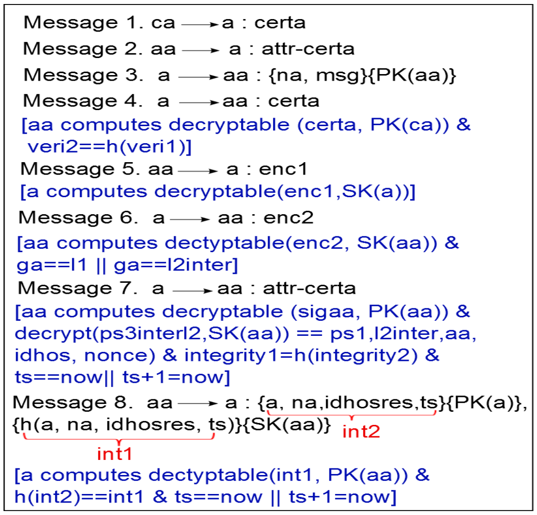
The purpose of the ELAA protocol is to link distributed data objects of the same patient managed by different HSPs, but this set of objects cannot be linked to the patient's real identity (e.g. NHS number). To facilitate this type of access, the enhanced protocol is performed on top of the LAA in [16]. So first, the requesting user will run the ELAA protocol to learn where the patient's objects are stored (i.e., get the HSPs's identity). Then, the requesting user will run the LAA protocol to get the patient's distributed objects from each HSP managing the patient's objects. Table I shows the notation used in the ELAA protocol. Fig 1 shows the message sequences of the ELAA protocol.

In the ELAA protocol, the communication channel is based on the Secure Socket Layer (SSL) protocol [21] to provide security for data transmission. For protocol analysis using Casper/FDR2, we assume the following. (A1) The underlying cryptographic algorithms used in SSL's public key and symmetric key ciphers are secure. (A2) All parties unconditionally trust the certification authority and public keys signed by it. The certification authority certifies the public key for clients. (A3) All parties unconditionally trust the attribute authority who issues the attribute certificates for clients. (A4) Patients' records have already been de-identified. That is their identity or NHS number has been replaced with a pseudonym.

TABLE I
THE ELAA PROTOCOL NOTATION AND DESCRIPTION

Notation	Description
a	An identifier of an initiator/client
ca	An identifier of a certification authority
aa	An identifier of a attribute authority
ga	A group membership of a
nx	A random nonce of x
encn	A challenge response of n
PKx	A public key of x
SKx	A secret Key of x
ts	A timestamp (an expiration time)
h	A hash function
msg	A message of data request
idhos	An identifier of a hospital identity
idhosres	An identifier of a hospital identity response
certa	A PK-certificate of client *a* generated by *ca*
attr-certa	An attribute certificate of client *a* generated by *aa*
veri1, veri2	An integrity verification of certa
ps3interl2	An L3 pseudonym Type-II
sigaa	A signature of *aa*
sigaa2	An identifier to verify the response's signature
integrity1,integrity2	An integrity verification of attr-certa
int1, int2	An integrity verification of the response

In the ELAA protocol, *ca* is the certification authority who issues public-key (PK) certificates, and *aa* is the attribute authority (e.g., a central trusted third party) who issues attribute certificates to legitimate users. *a* is the client or the initiator of the request. The PK-certificate includes two parts, {a, Pk(a), 12inter, ts} and {h(a, Pk(a), 12inter, ts)}{SK(ca)}. The first part, contains information about the client, such as, identity *a*, public key PK(a), group membership *l2inter* and timestamp *ts*. The second part, is the signature of the *ca*. Issuer *ca* signs subject *a*, public key *PK(a)*, a group membership *l2inter* and timestamp *ts* using its own private key *SK(ca)*, which is only known to the *ca*. Since the certificate is encrypted with the private key of ca, any other user cannot spoof it. The following describes the message sequence of the ELAA protocol depicted in Fig 1.



Message 1. ca → a : certa
Message 2. aa → a : attr-certa
Message 3. a → aa : {na, msg}{PK(aa)}
Message 4. a → aa : certa
[aa computes decryptable (certa, PK(ca)) &
veri2==h(veri1)]
Message 5. aa → a : enc1
[a computes decryptable(enc1,SK(a))]
Message 6. a → aa : enc2
[aa computes dectyptable(enc2, SK(aa)) &
ga==l1 || ga==l2inter]
Message 7. a → aa : attr-certa
[aa computes decryptable (sigaa, PK(aa)) &
decrypt(ps3interl2,SK(aa)) == ps1,l2inter,aa,
idhos, nonce) & integrity1=h(integrity2) &
ts==now|| ts+1=now]
Message 8. aa → a : $\{\underbrace{\text{a, na,idhosres,ts}}_{\text{int2}}\}$\{PK(a)\},
$\{\underbrace{\text{h(a, na, idhosres, ts)}}_{\text{int1}}\}$\{SK(aa)\}
[a computes dectyptable(int1, PK(aa)) &
h(int2)==int1 & ts==now || ts+1=now]

Fig. 1. The ELAA protocol description

Message 1: Certificate authority *ca* issues and sends the PK-certificate, *certa*, to client *a* in order to authenticate client a.

Message 2: Attribute Authority *aa* issues and sends the attribute certificate, *attr-certa*, to client *a*. This certificate includes the issuer's name (aa), the client's name (a), an L3 pseudonym (ps3interl2), a timestamp (ts) and the issuer's signature on the certificate. The L3 pseudonym (ps3interl2), contains another pseudonym, a lower-level one called, ps1, which can be used to link a patient's distributed objects managed by multiple HSPs. In other words, this type of L3 pseudonym is essential to facilitate the linkable anonymous access. Hence, it is given to the legitimate users as part of their access credentials to enable this type of access.

Message 3: Client *a* sends his/her nonce (na) along with a message of the request encrypted with *aa*'s pubic key.

Message 4: Client *a* sends his PK-certificate (certa) to *aa*. This certificate contains *veri1* and *veri2*. *veri1* contains the plain content of the certificate. *veri2* contains the deciphered *ca*'s signature on the certificate. Using *veri1* and *veri2* allows checking the integrity of the certificate to ensure that the certificate has not been modified during transmission. So first, verifier *aa* validates the *ca's* signature on the certificate and then, it verifies the certificate's integrity using *veri1* and *veri2*. This step is essential to ensure the correctness of the certificate.

Message 5: Verifier *aa* sends *enc1* to client *a* which contains the verifier's identity (aa), user's nonce (na) and the verifier's nonce (naa) encrypted with $PK(a)$. Client *a* checks if *enc1* is decryptable by $SK(a)$ and contains the right nonce na. This step is essential to allow client *a* to authenticate verfier *aa*.

Message 6: Client *a* sends *encr2* to recipient *aa*. Variable *encr2* contains the items a and naa encrypted with $PK(aa)$. Recipient *aa* checks if *enc2* is decryptable by $SK(aa)$ and contains the right nonce naa. This is to allow *aa* to authenticate *a*. Also in this step, *aa* checks a's group membership to ensure that the client belongs to the right group and legitimate for this type of access. In the ELAA protocol, only users belonging to L1 or L2inter user group can perform this type of access. So L2 users in the LAA are not legitimate to perform this type of access. They need higher privileges to do so.

Message 7: If user authentication was successful, *a* sends to *aa* his *attr-cert* to check his authorisation. Verifier *aa* checks the correctness of the certificate. It completes this by verifying the signature on the certificate and checks a's access credentials. That is to ensure that the certificate contains the right type of L3 pseudonym (ps3interl2). After that, it verifies the integrity of the lower-level pseudonym (ps1) to ensure it has not been altered during transmission.

Message 8: Finally, if user authorisation was successful, *aa* forwards to *a* the response in $int1$ and $int2$. *int2* contains the requested data in variable *idhosres*, which contains HSPs names, a timestamp (ts), user's nonce (na) and the user's identity (a) all encrypted with the user' public key. Variable *int1* contains same items as in *int2* but singed by *aa*. Finally, user *a* performs the final checks. (1) Checking the aa' signature on *int1* and verifying the integrity of the data using *int1* and *int2*. (2) Checking the timestamp to ensure data freshness.

B. Modelling the ELAA protocol Using Casper/FDR2

Based on the ELAA protocol's notation in Table I, we model the ELAA protocol in Casper's script, as shown below.

```
#Protocol description
--ca issues and sends PK-certificate to client a
0. ca -> a : {{a,PK(a),{l2inter}%ga,ts}%veri1,
{{h(a,PK(a), {l2inter}%ga,ts)}%veri2}{SK(ca)%skca}
%certa}{PK(a)}
--a wants to contact aa
1. -> a : aa
--a sends his original request message with a nonce
2a. a -> aa : {msg, na}{PK(aa)}
--a sends his PK-certificate to be verified by aa
2b. a -> aa :{veri1%{a,PK(a),ga%{l2inter},ts},{certa
%{veri2%{h(a,PK(a),ga%{l2inter},ts)}}}{SK(ca)}}{PK(aa)}
[decryptable(certa, PK(ca)) and veri2== h(veri1) and
ts==now or ts+1==now]
--Mutual authentication and checking user membership
3. aa -> a : {aa, na, naa}{PK(a)} %enc1
[decryptable (enc1, SK(a))]
4. a -> aa :{a, naa}{PK(aa)} %enc2
[decryptable(enc2,SK(aa))and ga==l2inter or ga==l1]
--aa issues and sends attribute certificate to a
5a. aa -> a :{aa,a,{{ps1,l2inter, aa, idhos,
nonce}%integrity2, {h(ps1, l2inter, aa, idhos,
nonce)}% integrity1}%ps3interl2,ts}{PK(a)}
5b. aa -> a : {h(aa,a,ps3interl2,ts)}{SK(aa)}%sigaa
[ts==now or ts+1==now]
--a sends to aa his attribute certificate for
authorisation verification
6a. a -> aa :{aa,a, ps3interl2
%{integrity2%{ps1,l2inter,aa,idhos,nonce},integrity1%
{h(ps1,l2inter,aa,idhos,nonce)}}, ts}
6b. a -> aa:sigaa%{h(aa,a,ps3interl2,ts)}{skaa%SK(aa)}
[decryptable(sigaa,PK(aa)) and integrity1==
h(integrity2) and decrypt(ps3interl2, SK(aa))==(ps1,
l2inter,aa,idhos,nonce) and ts==now or ts+1==now]
--aa sends the response to a
7. aa -> a : {{a, na, idhosres, ts} %int2,
{h(a,na,idhosres,ts)%int1}{SK(aa)}%sigaa2}{PK(a)}
[decryptable(sigaa2,PK(aa)) and int1== h(int2) and
ts==now or ts+1==now]
```

C. ELAA Protocol Security Requirements

In this section, we identify the ELAA protocol security requirements. These requirements are essential to preserve patient privacy in e-health applications.

* **(R1) Data Confidentiality:** Confidentiality is an important requirement that provides security and privacy in e-health applications. It offers protection against attacks such as forgery and spoofing. An unauthorised party should not be able to learn anything about any communication between two entities by observing or even tampering the communication lines.

* **(R2) Integrity Protection:** A strong integrity protection mechanism should be deployed to protect against data tampering. The ELAA protocol should detect any unauthorised alteration to data being transmitted over the channel.

* **(R3) Ensuring Accountability:** The protocol should obtain an undeniable response from entities participating in the protocol. That is, to ensure that the originator of a communication cannot deny it later.

* **(R4) Mutual Authentication:** Also known as two-way authentication, refers to both entities of the protocol should authenticate each other to permit the exchange of information there-between.

* **(R5) Certificate Manipulation Protection:** It should be guaranteed that the certificates (i.e., PK-certificates) used in the protocol are valid and have not been corrupted or modified during transmission.

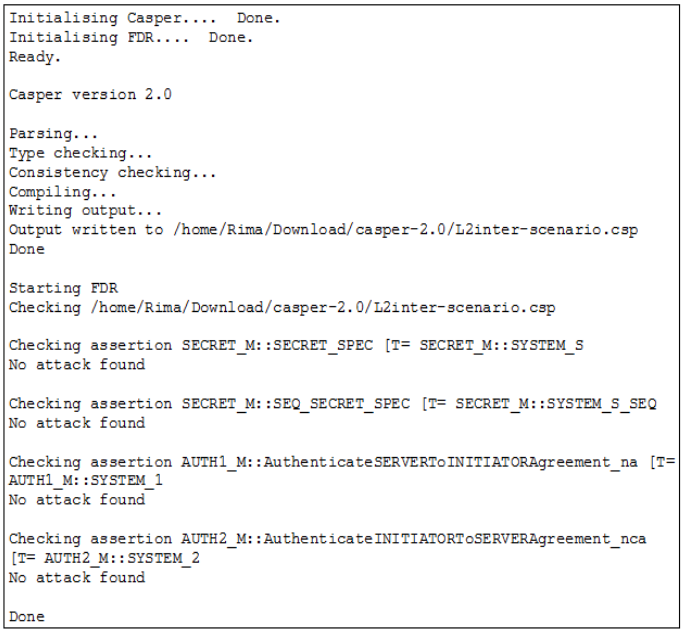
* **(R6) Credential Forgery Protection:** It should be assured that users' credentials are not stolen or forged, as it can lead to the elevation of privileges attack. This attack occurs when a user with limited privileges assumes the identity of a user with higher privileges to gain access to patient confidential data.

* **(R7) Data Freshness:** There should be a proof that nonces, generated during protocols, are fresh.

* **(R8) Anonymous Linkability:** A user with L2inter access credentials should be able to link distributed anonymous objects of the same patient managed by different HSPs but should not be able to link them to the patient's real identity.

D. Verification Result and Security Analysis of The ELAA Protocol

The verification result using the Casper/FDR2 model checking tool shows that the ELAA protocol has met all the security requirements identified in Section III-C. The result of the verification is shown in Fig 2.



```
Initialising Casper....  Done.
Initialising FDR....  Done.
Ready.

Casper version 2.0

Parsing...
Type checking...
Consistency checking...
Compiling...
Writing output...
Output written to /home/Rima/Download/casper-2.0/L2inter-scenario.csp
Done

Starting FDR
Checking /home/Rima/Download/casper-2.0/L2inter-scenario.csp

Checking assertion SECRET_M::SECRET_SPEC [T= SECRET_M::SYSTEM_S
No attack found

Checking assertion SECRET_M::SEQ_SECRET_SPEC [T= SECRET_M::SYSTEM_S_SEQ
No attack found

Checking assertion AUTH1_M::AuthenticateSERVERToINITIATORAgreement_na [T=
AUTH1_M::SYSTEM_1
No attack found

Checking assertion AUTH2_M::AuthenticateINITIATORToSERVERAgreement_nca
[T= AUTH2_M::SYSTEM_2
No attack found

Done
```

Fig. 2. Verification result of the ELAA protocol using Casper/FDR2

* **(R1) Data Confidentiality:** was achieved by deploying cryptographic techniques such as symmetric cryptoystem, asymmetric cryptoystem, and hash functions.

* **(R2) Integrity Protection:** was met by incorporating digital signatures and hash functions, which can detect any data alteration during transmission.

* **(R3) Ensuring Accountability:** was fulfilled by using digital signatures of both entities, the sender and receiver.

* **(R4) Mutual Authentication:** was accomplished by integrating the challenge response protocol.

* **(R5) Certificate Manipulation Protection:** this requirement has been abided by including a timestamp in the certificate, which can detect any manipulation by the intruder.

* **(R6) Credential Forgery Protection:** was met by including the credential holder identity in both types of certificates, the PK-certificate and the attribute certificate. So by checking that both certificates contain the same credential holder identity, we can ensure that both credentials have not been forged.
* **(R7) Data Freshness:** was achieved by including a freshly random nonce with the transmitted data.
* **(R8) Anonymous Linkability:** was fulfilled by integrating the L3 pseudonym-TypeII in the L2inter user's access credential. This allows linkable anonymous access to patient data as it contains a lower-level pseudonym that can be used to link the distributed patient's objects managed by several HSPs.

IV. Implementation and Performance Evaluation

In this section, we focus on the implementation and performance evaluation of the ELAA protocol. To achieve this, we have built a prototype using the Java 2 platform (standard edition), as it is suitable for e-health applications. It offers implementations for several cryptographic primitives and key management services needed for our solution.

We measure performance by two metrics, minimising access delay and minimising server computation time. These metrics are essential to evaluate our protocol's performance. An access delay is defined as the time elapsed from submitting an access request to the time when the response to the access request is received. A server computation time is the time needed for the server to complete the necessary operations, verifications and checks from receiving the request to the time when the response to the request is sent. Both metrics should be kept as low as possible.

We have measured the time taken to execute the access protocol based upon the prototype under two scenarios. In the first scenario, we run the LAA protocol introduced in [16]. This scenario allows linking a patient's different objects managed by a single HSP. This protocol has been described in detail in [16]. In the second scenario, we run the ELAA protocol, which has been introduced in this paper. This scenario is similar to the one above, however, it allows linking a patient's object managed by multiple HSPs. The measurements are taken for 10 execution rounds for each scenario, and the averages are calculated. The results are shown in Fig 4 and Fig 3.

A. Implementation Platform

To prototype the ELAA protocol, we have used a desktop computer running Windows 8 with a 2.30 GHz Intel Core i3 and 8GB of RAM. The software used to implement the ELAA protocol is Java 2 Platform, Standard Edition (J2SE), which was also used to implement the LAA in [16]. JAVA is chosen because it supports a set of standard security interfaces. Examples of these interfaces include the hash functions such as SHA-256 [22] and MD-5 [23], the symmetric encryption algorithms such as AES [24] and 3DES [25] and the asymmetric encryption algorithms such as RSA [26] and DSA [27].

B. Performance Evaluation Parameters and Target

The performance evaluation parameters we rely on in this paper are (1) the patient's records are distributed in different databases, which are managed by different HSP (e.g., hospitals). That is we run the simulation on a distributed manner and test its performance. (2) Running the simulation where the database size of each HSP increases, patient wise and record wise. We first, run the simulation with the parameter 10 objects by 1000 patients, and then we increase the object's size by ten and the patients' number by 1000. (3) We focus on a single patient data access.

The target of the performance evaluation is to show that the ELAA protocol offers a better and secure access to distributed EPRs than the LAA protocol and with a linear increase in performance. In other words, the ELAA protocol aims to balance between security and performance while supporting distributed data access but without adding a massive amount of overhead into the solution.

C. Performance Evaluation Result and Analysis

It can be seen from Fig 3 that the time (Access delay) taken to execute the ELAA protocol is 432 milliseconds in its peak, which is approximately 80% more than the time taken in the LAA protocol, which is 78.5 milliseconds shown in Fig 4. The server computation time in ELAA protocol is 425 milliseconds, which is approximately 80% more than that in the LAA protocol, which is 77.7 milliseconds.

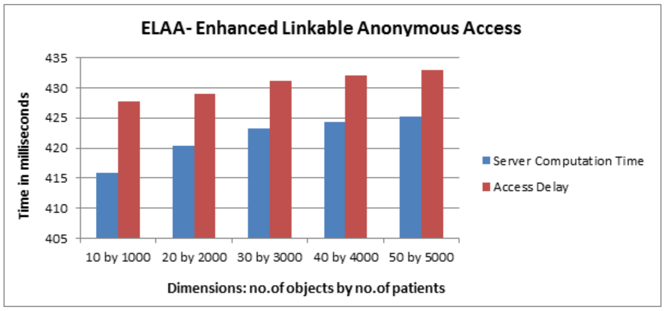


Fig. 3. Performance evaluation result of the ELAA protocol

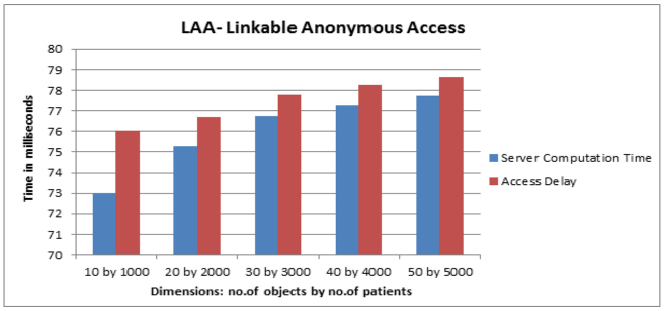


Fig. 4. Performance evaluation result of the LAA protocol

The extra cost in the ELAA Scenario is caused by several reasons, (1) the ELAA protocol contains an additional security layer, which was added on top of the LAA protocol. (2) The extra communications between the client and the verifier. (3) The extra computations in signature verifications by both the client and the verifier. (4) The extra computation in the attribute certificate verification by the verifier. (5) The extra computation in checking the timestamp in the attribute certificate. (6) The extra computation in validating the pseudonym (PS3interl2) included in the attribute certificate. (7) The extra integrity check of the lower-level pseudonym (PS1) included in PS3interl2. (8) The extra computation in signing the requested data or the response before sending it to the client. (9) Finally, the extra computation in checking the integrity of the response and the time stamp included in the response.

V. Conclusion and Future Work

In this paper, we have introduced an enhanced protocol (ELAA protocol) to the LAA protocol presented in [16]. The idea behind the enhanced protocol is that we have allowed users to access distributed EPRs that are under multiple HSPs's management and not only under a single HSP management as the case in the LAA protocol. We showed that the ELAA protocol is also secure while maintaining a linear increase in performance. To achieve this, firstly, we have formally verified and analysed the ELAA protocol using the Casper/FDR2 verification tool. Secondly, we have implemented the ELAA to test its performance by building a prototype using the Java technology. The result from the verification using the Casper/FDR2 tool showed that the ELAA protocol has met important security requirements. It supports linkable anonymous access to a patient's distributed EPRs by integrating significant cryptographic techniques. It ensures confidentiality of patient sensitive data. It provides data freshness by relying on timestamps and nonces. It is protected from certificate manipulation and credential forgery attacks. It ensures accountability by deploying digital signatures. Mutual authentication is also provided to obtain unforgeable proof of other participant's authenticity before it engages in the protocol with that participant. In addition to fulfilling important security requirements, the result from the ELAA protocol implementation showed that the ELAA protocol had successfully balanced between security and performance. That is the increase in performance was linear with the increase of security. So our analysis proved that the ELAA protocol is secure and efficient. It allows a client and a server to exchange some sensitive distributed patient data in a secure manner and within a reasonable amount of time. Our future work is concerned with extending our analysis of the ELAA protocol to other security protocols and specifically, e-health protocols, taking into account security and performance as major criteria.

Acknowledgement

This work is financially sponsored by the Ministry of Higher Education in Saudi Arabia.

References

[1] J. G. Hodge, "Health information privacy and public health," *The Journal of Law, Medicine & Ethics*, vol. 31, no. 4, pp. 663–671, 2003.

[2] P. S. Appelbaum, "Privacy in psychiatric treatment: threats and responses," *FOCUS: The Journal of Lifelong Learning in Psychiatry*, vol. 1, no. 4, pp. 396–406, 2003.

[3] J. D. Halamka, P. Szolovits, D. Rind, and C. Safran, "A www implementation of national recommendations for protecting electronic health information," *Journal of the American Medical Informatics Association*, vol. 4, no. 6, pp. 458–464, 1997.

[4] A. Appari, M. E. Johnson, and D. L. Anthony, "Hipaa compliance: An institutional theory perspective." in *AMCIS*, 2009, p. 252.

[5] A. Pfitzmann and M. Hansen, "Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management - a consolidated proposal for terminology," Tech. Rep., February 2008.

[6] B. Alhaqbani and C. Fidge, "Privacy-preserving electronic health record linkage using pseudonym identifiers," July 2008, pp. 108–117.

[7] B. S. Elger, J. Iavindrasana, L. Lo Iacono, H. Müller, N. Roduit, P. Summers, and J. Wright, "Strategies for health data exchange for secondary, cross-institutional clinical research." *Computer methods and programs in biomedicine*, vol. 99, no. 3, pp. 230–251, September 2010.

[8] K. Pommerening and M. Reng, "Secondary use of the ehr via pseudonymisation." *Studies in health technology and informatics*, vol. 103, pp. 441–446, 2004.

[9] D. Slamanig and C. Stingl, "Privacy aspects of ehealth," in *ARES '08: Proceedings of the 2008 Third International Conference on Availability, Reliability and Security*. Washington, DC, USA: IEEE Computer Society, March 2008, pp. 1226–1233.

[10] L. Iacono, "Tmulti-centric universal pseudonymisation for secondary use of the ehr," *In: Proc. of HealthGrid:*, pp. 239–247, 2007.

[11] P. Schartner and M. Schaffer, "Unique user-generated digital pseudonyms," in *Computer Network Security*, ser. Lecture Notes in Computer Science, V. Gorodetsky, I. Kotenko, and V. Skormin, Eds. Berlin, Heidelberg: Springer Berlin / Heidelberg, 2005, vol. 3685, ch. 15, pp. 194–205.

[12] N. Zhang, A. Rector, I. Buchan, Q. Shi, D. Kalra, J. Rogers, C. Goble, S. Walker, D. Ingram, and P. Singleton, "A linkable identity privacy algorithm for healthgrid," in *From Grid to Healthgrid: Proceedings of Healthgrid 2005*, 2005, pp. 234–245.

[13] M. Deng, D. DeCock, and B. Preneel, "Towards a cross-context identity management framework in e-health," *Online Information Review*, vol. 33, no. 3, pp. 422–442, 2009.

[14] T. Neubauer and J. Heurix, "A methodology for the pseudonymization of medical data," *International Journal of Medical Informatics*, vol. 80, no. 3, pp. 190–204, Mar. 2011.

[15] R. Addas and N. Zhang, "An enhanced approach to supporting controlled access to eprs with three levels of identity privacy preservations," in *Information Quality in e-Health*, ser. Lecture Notes in Computer Science, A. Holzinger and K.-M. Simonic, Eds., vol. 7058. Springer Berlin / Heidelberg, 2011, pp. 547–561.

[16] R. Addas and N. Zhang, "Formal security analysis and performance evaluation of the linkable anonymous access protocol," to appear in Proceedings of the 2014 Asian Conference on Availability, Reliability and Security (AsiaARES 2014) in conjunction with ICT-EurAsia 2014, ser. Lecture Notes in Computer Science of Springer, L. et al. (Eds.), Ed. LNCS 8407.

[17] G. Lowe, "Casper: a compiler for the analysis of security protocols," pp. 18–30, Jun. 1997.

[18] F. S. E. LTD., "Failure-divergences refinement fdr2 manual," 2010.

[19] I.-G. Kim, H.-S. Kim, J.-Y. Lee, and J.-Y. Choi, "Analysis and modification of ask mobile security protocol," in *Mobile Commerce and Services, 2005. WMCS '05. The Second IEEE International Workshop on*, 2005, pp. 79–83.

[20] P. Chan, R. Lee, and D. Kramer, *The Java Class Libraries, Volume 1: Supplement for the Java 2 Platform, Standard Edition, V 1.2*. Addison-Wesley Professional, 1999, vol. 1.

[21] D. Wagner and B. Schneier, "Analysis of the ssl 3.0 protocol," in *In proceedings of the second Unix Workshop on electronic commerce*. USENIX Association, 1996, pp. 29–40.

[22] H. Gilbert and H. Handschuh, "Security Analysis of SHA-256 and Sisters Selected Areas in Cryptography," *Selected Areas in Cryptography*, vol. 3006, pp. 175–193, 2004.

[23] B. Kaliski and M. Robshaw, "Message authentication with md5," *CryptoBytes (RSA Labs Technical Newsletter)*, vol. 1, no. 1, 1995.

[24] J. Blömer and J.-P. Seifert, "Fault based cryptanalysis of the advanced encryption standard (aes)," in *Financial Cryptography*, R. N. Wright, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, vol. 2742, ch. 12, pp. 162–181.

[25] Y. Fuping, S. Liyuan, and J. Yuanming, "Design and implementation of 3des encryption system based on dsp [j]," *Computer Measurement & Control*, vol. 7, p. 051, 2009.

[26] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Commun. ACM*, vol. 21, pp. 120–126, February 1978.

[27] L. Harn, M. Mehta, and W.-J. Hsin, "Integrating diffie-hellman key exchange into the digital signature algorithm (dsa)," *Communications Letters, IEEE*, vol. 8, no. 3, pp. 198–200, 2004.