# A Novel 3-Level Access Control (3LAC) Framework for Data Access in a Healthcare Cloud Context

Gabriel Sanchez Bautista
and Ning Zhang

School of Computer Science
The University of Manchester
Manchester M13 9PL, United Kingdom
Email: {sanchezg, nzhang}@cs.man.ac.uk

*Abstract*—**While the use of Personal Health Records (PHRs) in a cloud computing environment brings benefits, it also raises concerns. One of the major concerns is how to prevent patients' data managed by a cloud provider (i.e., a third-party) from being revealed to unauthorised entities, including the cloud provider. One way to address this concern is to protect data by using an Attribute-Based Encryption (ABE) based solution, in which data is encrypted before it is uploaded to the cloud provider. As part of the solution, data is first encrypted by using a symmetric key, which is then protected by using a pair of keys: a public and a private key. The public key is used for encrypting the symmetric key, and the private key is used for decrypting the symmetric key. To access data, a user needs to acquire the private key. Existing work on controlling the access of PHRs in a cloud environment largely focuses on how to make the solutions more fine-grained or how to strike the balance between data access granularity and efficiency. However, there is little work on ensuring how to securely distribute a private key in an ABE based PHRs access control system. This paper addresses the issue by proposing a multi-level approach to private key distribution in a Ciphertext-Policy ABE (CP-ABE) based access control model. This multi-level approach is inspired by our observation that patients' data may not have the same level of sensitivity, and to optimise the trade-off between privacy protection and costs (i.e., computational and communication), the level of access control should be tailored based on the data sensitivity levels. We have implemented these ideas by designing and evaluating a Novel 3-Level Access Control Framework (3LAC) that combines the Shamir's Secret Sharing scheme with a CP-ABE based access control model, in which to access more sensitive data a user needs to acquire more shares, and for the acquisition of each share, there is an authentication process. The results of the evaluation have demonstrated that the 3LAC Framework balances the performance according to the data sensitivity levels as compared with a fixed-level approach.**

***Keywords*–Privacy; Security; Attribute-based encryption; Secret sharing; Access control; eHealth ;Multi-level.**

## I. Introduction

According to the Health Insurance Portability and Accountability Act (HIPAA) [1], Personal Health Records (PHRs) are described as "electronic records of an individual's health information by which the individual controls access to the information and may have the ability to manage, track, and participate in his or her own health care..." [2][3]. Similarly, the American Health Information Management Association (AHIMA) [4] describes PHRs as "an electronic resource of health information needed by individuals to make health decisions, in which individuals own and manage the information that comes from the healthcare providers and the individual" [5]. The definition that is given by AHIMA also specifies that PHRs should be maintained in a private and secure environment, with the individual specifying the access rights [6]. PHRs may be implemented over a cloud computing environment. If the patients' PHRs are stored in the cloud, it means those PHRs can be accessed anywhere provided there is a connection to the Internet [7][8]. However, in this case, the patients' PHRs are stored in an entity (a third-party) that is neither the patient (i.e., the data owner) nor the healthcare service providers. This raises a question as how a patient can ensure that only authorised users can access his or her PHRs [9]–[11]. Similarly, this raises an issue of how to prevent that the cloud service provider reads the patients' PHRs and uses that information for other purposes [12]. For this reason, there is a need to have a privacy-preserving access control solution. Role-Based Access Control (RBAC) [13], is one of the early proposed access control models. In this model, users are assigned roles and permissions are applied to those roles. In RBAC, a user may perform an operation only if that user has been assigned a role and permissions have been granted to that role. However, RBAC does not protect the PHRs against unauthorised access by the data-manager. In our context, RBAC only protects unauthorised access by users, but it does not protect against unauthorised access by the the cloud service provider. This means, there is a need that the PHRs uploaded to the cloud service provider should be first encrypted, so that the cloud service provider cannot read those PHRs although they are stored there. This brings the need for encryption. The Identity-Based Encryption (IBE) scheme [14] allows data to be encrypted by using the identity of the destined user. In this scheme, the public key of a user contains information about the identity of the user (e.g., id number). However, there are limitations with this approach as the sender always needs to know in advance the identity of the receiver. Also, IBE does not support a fine-grained description of a user, so IBE cannot support fine-grained access control. To support a fine-grained access control so that users can be assigned different attributes to provide a more detailed description of them, the Attribute-Based Encryption (ABE) scheme [15] was proposed. In this approach, users can be assigned with different attributes that specify their identities and the permissions to access a particular piece of data. With ABE there is no need to know in advance the identity of the destined user as encryption is performed based on attributes rather than the identity, and those attributes can be used to describe different users. However, in existing ABE based access control solutions, issues in relation to the distribution of the private keys (i.e., decryption keys) are not addressed. It is largely assumed that private keys are securely distributed to their intended users. Based on the discussion of the challenges described in this section, our motivation is to design an access control solution to answer the following research questions.

- Q.1. How to strengthen the privacy protection in patients' data when data is managed by an untrusted third-party while keeping the computational and communication costs as low as possible?
- Q.2. How to tailor the privacy protection given to data such that high sensitive data may have a strong level of protection but low sensitive data may not need a strong level of protection?

The rest of this paper is organised as follows: Section II presents the notations used in the design of the solution. Section III describes the 3-Level Access Control (3LAC) Framework in detail. Section IV describes the experiments setup. Section V discusses the experimental scenarios and settings. Section VI presents the experimental results and discussions. Section VII presents the conclusion and future work.

## II. Notations

The notations used in the design of the 3LAC Framework are given in Table I.

TABLE I. NOTATIONS.

| *Notation* | *Meaning* |
|---|---|
| $PrK_n^{CP-ABE}$ | User $n$'s CP-ABE private key |
| $PuK_n^{CP-ABE}$ | User $n$'s CP-ABE public key |
| $PrK_n^{RSA}$ | User $n$'s RSA private key to generate $n$'s signature |
| $PuK_n^{RSA}$ | User $n$'s RSA public key to verify $n$'s signature |
| $LK_n^e$ | User $n$'s level key for level $e$ , where $e \in \{1,2,3\}$ |
| $S_{LK_n^e}^w$ | Share $w$ of $LK_n^e$ , where $w \in \{1,2,3\}$ |
| $ns$ | The number of shares that $LK_n^e$ is split into |
| $k$ | The number of shares needed to reconstruct $LK_n^e$ |
| $SyK_i^{AES,j}$ | Symmetric key of data-object $j$ of patient $i$ |
| $Obj_i^j$ | Data-object $j$ of patient $i$ |
| $CT_{Obj_i^j}$ | Ciphertext of data-object $j$ of patient $i$ |
| $Sig_n$ | Digital signature of $n$ |
| $PKcert_n$ | RSA public key certificate of $n$ |
| $ATcert_n$ | Attribute certificate of $n$ |
| $G1$ | User-group 1 |
| $G2$ | User-group 2 |
| $G3$ | User-group 3 |
| $L1$ | Identifier of privilege level 1 (to access low sensitive data) |
| $L2$ | Identifier of privilege level 2 (to access medium sensitive data) |
| $L3$ | Identifier of privilege level 3 (to access high sensitive data) |
| $LS$ | Low sensitive data |
| $MS$ | Medium sensitive data |
| $HS$ | High sensitive data |
| $CA$ | Certification Authority |
| $AA$ | Attribute Authority |
| $PrKGA$ | Private Key Generation Authority |
| $RLKA$ | Root Level Key Authority |
| $LKA_1$ | Level Key Authority 1 |
| $LKA_2$ | Level Key Authority 2 |
| $LKA_3$ | Level Key Authority 3 |

$L1$, $L2$, and $L3$ are used to identify the level of access privilege granted to a user. A $LK$ is a symmetric key (i.e., AES), which is used to distribute a CP-ABE private key to the intended user.

## III. A Novel 3-Level Access Control (3LAC) Framework

The 3LAC Framework supports privacy protection in accordance with the data sensitivity levels. From analysing different data access scenarios, we have identified three sensitivity levels, i.e., low, medium, and high. For each sensitivity level, we propose an access privilege level, i.e., (L1) access to low sensitive data, (L2) access to medium sensitive data, and (L3) access to high sensitive data. To access more sensitive data, a user has to obtain more shares in order to reconstruct a Level Key (LK). A LK is used by a user to authenticate him/herself and acquire a CP-ABE private key (also known as Key Decryption Key, KDK). The LK is used to distribute securely a CP-ABE private key to the intended user. The user has to acquire the shares from different Level Key Authorities (LKAs) in order to reconstruct his or her LK. In addition, users are classified into different user-groups based on their levels of access privileges, i.e., G1, G2 and G3. Table II shows the relation among user-groups, levels of access privileges, and data sensitivity.

TABLE II. USER-GROUPS, LEVELS OF ACCESS PRIVILEGES AND DATA SENSITIVITY.

| *User-groups* | *Levels of access privileges granted* | *Data sensitivity* |
|---|---|---|
| $G3$ | $L3$, $L2$, $L1$ | $HS$, $MS$, $LS$ |
| $G2$ | $L2$, $L1$ | $MS$, $LS$ |
| $G1$ | $L1$ | $LS$ |

The 3LAC Framework consists of two architectures, i.e., AQ1: Architecture for Key Generation and Distribution, and AQ2: Architecture for Data Uploading and Access.

### A. AQ1: Architecture for Key Generation and Distribution

This architecture (AQ1) is responsible for generating the Level Keys (LKs). Also, for the distribution of the shares to their respective users. In addition, AQ1 is responsible for the acquisition of an attribute certificate and a RSA public key certificate by a user. AQ1 consists of the following entities and their functional components.

- Root Level Key Authority ($RLKA$): This is a trusted authority that is responsible for generating the Level Keys of users, and split each Level Key of L2 and L3 into shares, accordingly. $RLKA$ is also responsible for distributing the shares to the respective Non-Root Level Key Authorities (i.e., $LKA_1$, $LKA_2$, and $LKA_3$). The functional components of $RLKA$ are the generator, the dispatcher, and the database. The generator generates the Level Keys and splits them into shares (Level Keys of L2 and L3). The dispatcher communicates with the Non-Root Level Key Authorities to distribute the shares, and the database is where the shares and Level Keys are stored.
- Non-Root Level Key Authorities ($LKA_1$, $LKA_2$, and $LKA_3$): They are trusted authorities, which are responsible for distributing the respective shares to the users. In the case of $LKA_1$, it distributes a Level Key for a user that belongs to the G1 user-group. The functional components of each authority are an authentication point, a dispatcher, and a database. The authentication point is where a user is authenticated when requesting a share/Level Key. The dispatcher is

responsible for distributing a share/Level Key to the requesting user. A share/Level Key is retrieved from the database of the corresponding Non-Root Level Key Authority.

- Private Key Generation Authority ($PrKGA$): This is a trusted authority that is responsible for generating the CP-ABE public and private keys (i.e., KEKs and KDKs) for users. The functional components of $PrKGA$ are the authentication point, the generator, the database, and the dispatcher. The authentication point is where a user is authenticated when requesting the acquisition of his or her KDK. The generator generates the KEKs and KDKs. The database contains the data needed to generate the KEKs and KDKs. The dispatcher distributes a KDK to the requesting user.
- Certification Authority ($CA$): This is a trusted authority that is responsible for signing a user's $PKcert$ (i.e., a user's RSA public key certificate). A user's RSA public key is certified by this authority. The functional components of $CA$ are a certificate issuance and a database. The certificate issuance is used to sign the $PKcert$, and the database contains the certificate data.
- Attribute Authority ($AA$): This is a trusted authority that is responsible for generating an attribute certificate (i.e., $ATcert$) for a user. The functional components of $AA$ are an attribute aggregator and a database. The attribute aggregator gathers all the attributes of a user and generates an $ATcert$. The database contains the data needed for generating an $ATcert$.

Based on the functions, AQ1 is divided into three functional blocks, i.e., AQ1-FB1: Initialisation, AQ1-FB2: Shares Acquisition, and AQ1-FB3: Key Decryption Key Acquisition.

- AQ1-FB1: Initialisation. In this functional block, the $RLKA$ distributes the shares (and L1 Level Keys) to the Non-Root Level Key Authorities (i.e., $LKA_1$, $LKA_2$, and $LKA_3$). Also, a user makes a request to the $AA$ to obtain an $ATcert$, and a request to the $CA$ to obtain a $PKcert$.
- AQ1-FB2: Shares Acquisition. In this functional block, a user makes a request to the Non-Root Level Key Authorities to obtain the shares that are needed to reconstruct his or her $LK$. Users of user-group G1 need to make a request to $LKA_1$. Users of user-group G2 need to make a request to $LKA_1$ and $LKA_2$, respectively. Users of user-group G3 need to make a request to $LKA_1$, $LKA_2$ and $LKA_3$, respectively. For each share acquisition, there is an authentication process.
- AQ1-FB3: Key Decryption Key Acquisition. In this functional block, a user makes a request to the $PrKGA$ to acquire a KDK (i.e., a CP-ABE private key), which then can be used to recover a Data Encryption Key (DEK), provided that the user has the right attributes to recover it. A DEK is used to encrypt and decrypt a patient's data-object. Upon receiving the request, the $PrKGA$ will send a challenge to the user to authenticate the user. The challenge is encrypted by using the user's $LK$. The user will need to decrypt the challenge by using his or her $LK$. Once the user has recovered the challenge, it is sent to $PrKGA$ as a proof that the user knows the $LK$. After successful authentication, the $PrKGA$ generates a CP-ABE private key for the user and encrypts it by using the user's $LK$. In other words, in addition to authenticating the user, the $LK$ is also used to distribute the CP-ABE private key to the user. By using his or her $LK$, the user can recover the CP-ABE private key.

### B. AQ2: Architecture for Data Uploading and Access

This architecture (AQ2) supports data uploading by patients, and data access by users. AQ2 consists of the following entities and their functional components.

- Cloud Service Provider ($CSP$): This is the third-party that manages patients' data-objects. The functional components are a data-objects agent, an authentication point, and a database. The data-objects agent receives the requests of data uploading by patients and the requests of data access by users. Authentication (i.e., challenge response authentication) is performed in the authentication point, and the database is where the encrypted data-objects are stored.
- Patients: Patients are data owners to whom data-objects belong to. Each patient has a set of data-objects. Each patient is responsible for encrypting his or her own data-objects and uploading them to the $CSP$. A patient specifies an access policy to govern who can recover a DEK, which will be used to recover a data-object.
- Users: A user is who requests access to a patient's data-objects. A user belongs to a user-group (G1, G2, or G3).

Based on the functions, AQ2 is divided into two functional blocks, i.e., AQ2-FB1: Uploading of a Data-Object by a Patient, and AQ2-FB2: A User Requesting Access to a Data-Object.

- AQ2-FB1: Uploading of a Data-Object by a Patient. In this functional block, a patient requests the uploading of a data-object to the $CSP$. Before the request is made, the patient first encrypts the data-object to protect it from unauthorised access by the $CSP$.
- AQ2-FB2: A User Requesting Access to a Data-Object. In this functional block, a user requests access to a patient's data-object. A data-object granted to a user is encrypted (i.e., ciphertext). This means the user needs to acquire a DEK to decrypt the data-object.

## IV. Experiments Setup

In this section, we describe the programming language, the database, the hardware platform and the configuration used to prototype the 3LAC Framework.

### A. Programming Language

The programming language used to prototype the 3LAC Framework is Java 2 Platform Standard Edition (J2SE) [16]. Java is chosen because it includes the Java Cryptographic Architecture (JCA) and the Java Cryptographic Extension (JCE). JCA and JCE provide the implementation of different cryptographic primitives and key management services that are required to prototype the 3LAC Framework. The key management services include a message digest function, X.509 digital certification facility, a secure random number generator and block ciphers, such as AES and RSA.

### *B. Database*

The database used in the 3LAC Framework is created using MySQL Workbench 6.3 [17]. This is a database design tool that integrates database design, creation and maintenance. MySQL uses a standard form of the well-known SQL data language. MySQL can be used with other languages, including Java. MySQL is considered an efficient tool to create and maintain patients' PHRs.

### *C. Hardware Platform and Configuration*

To prototype the 3LAC Framework, we used two desktop computers, machine_1 (M1) and machine_2 (M2). M1 is used as the client and M2 as the server machine. M1 and M2 have the following specifications: Windows 7 Enterprise, Service Pack 1, 64-bit operating system, 3.20 GHz, Intel Core i3, with 8GB of RAM memory. Hard Drive Disk: M1 has 195 GB, and M2 has 237 GB. We have decided to prototype the 3LAC Framework and run it under a two-machine set-up. The reason is to test, compare and evaluate the performance of the 3LAC Framework when the patients' data-objects are stored in a third-party's machine (e.g., a $CSP$), which is not the machine of the patient.

## V. Experimental Scenarios and Settings

We run experiments of the 3LAC Framework under three different access scenarios, which cover the different levels of protection that may be given to data. The scenarios are described as follows.

- **Scenario_A: Fixed-1-Acquisition**: All the data-objects are assumed to have the same sensitivity level, and a weak protection is applied. In other words, there is no distinction between data sensitivity levels. The Level Key of a user is not split into shares and a user only needs to perform one Level Key acquisition per lifetime of the key or until the user is revoked.
- **Scenario_B: Fixed-3-Acquisitions**: Data-objects are also assumed to have the same sensitivity level but a strong level of protection is applied. Each Level Key is split into 3 shares, and each user needs to acquire 3 shares in order to reconstruct his or her Level Key.
- **Scenario_C: The 3LAC Framework**: Data-objects are classified into three groups, each with a distinctive sensitivity level and different protection levels are applied. For data-objects with the highest sensitivity level, a Level Key is split into 3 shares. Similarly, to access data-objects with medium sensitivity level, a Level Key is split into 2 shares, and to access data-objects with the lowest sensitivity level, the Level Key is not split. In other words, the number of shares in which a Level Key is split depends on the level of access privileges granted to a user. The 3LAC Framework supports access to data-objects with three levels of protection. The fundamental difference among Scenario_A, Scenario_B, and Scenario_C is that Scenario_C is flexible and it can adjust the level of protection in adaptation with the data sensitivity levels.

These three access scenarios are defined to reflect the three access control protection levels. Scenario_A has the least protection level but also the least time-consuming case. Scenario_B has the highest protection level among the three scenarios but also the most time-consuming case. Scenario_C captures the 3LAC Framework, which adjusts the level of protection according with the data sensitivity levels. In this evaluation, our intention is to investigate the 3LAC Framework in terms of performance costs and scalability through different experiments. The experiments are run using three settings with a distribution of users belonging to different user-groups, as shown in Table III.

TABLE III. USER-GROUPS.

| *Settings (SE)* | *User-groups (G1, G2, or G3)* |
|---|---|
| SE1 | (60% users in G1) (30% users in G2) (10% users in G3) |
| SE2 | (30% users in G1) (60% users in G2) (10% users in G3) |
| SE3 | (10% users in G1) (30% users in G2) (60% users in G3) |

In Table III, we can see a different distribution among the user-groups of users. These settings are based on real-life scenarios, where we may have more users belonging to a particular user-group than another. To cover different possibilities, we give a 60 % for the biggest user-group in each setting, then a 30% for the second biggest user-group, and a 10% for the smallest user-group. We chose this distribution to make the percentage of each user-group representative such that even when adding the 30% + 10% user-groups, the 60 % user-group remains as the biggest group. We use SE1, SE2, and SE3 to observe how efficient is the 3LAC Framework for each user-group.

## VI. Experimental Results and Discussions

This section evaluates the 3LAC Framework. The experiments were run under settings SE1, SE2, and SE3, and with three different access scenarios (i.e., Scenario_A, Scenario_B, and Scenario_C). The aim of the experiments is to investigate the performance costs in supporting the three levels of access privileges and to see how the 3LAC Framework performs in terms of scalability.

### *A. Exp-1: Share Acquisition Time Imposed on Users*

This experiment investigates the share acquisition time imposed on users based on the different access scenarios and with a different distribution of their user-groups (i.e., some users need to request more shares than others). Then, we investigate the share acquisition time per user for each user-group. In this experiment, we use settings SE1, SE2 and SE3. The reason for using these settings is to investigate the share acquisition time imposed on users based on the number of shares needed and based on the different users' user-groups. We present the results of Exp-1 in Figure 1 for SE1, Figure 2 for SE2, and Figure 3 for SE3. The Share Acquisition Time Imposed on Users is measured in milliseconds (ms). In these figures, the x-axis of the graphs indicates the number of users requesting shares with values ranging from 0 to 50, with an increase in each scale of 10. The y-axis of the graphs indicates the share acquisition time measured in milliseconds, with values ranging from 0 to 160000, with an increase in each scale of 20000.

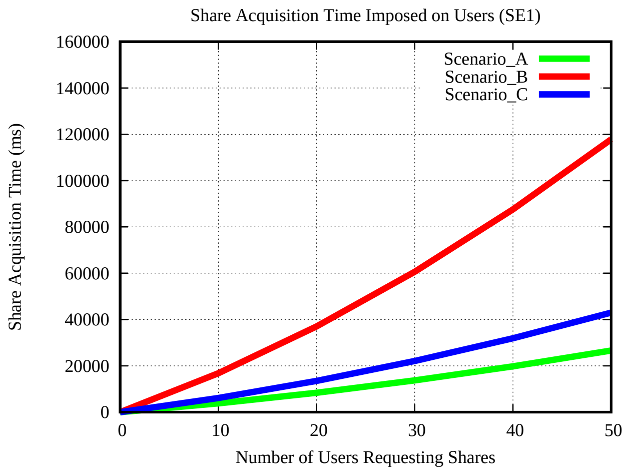

Figure 1. Share Acquisition Time Imposed on Users (SE1).

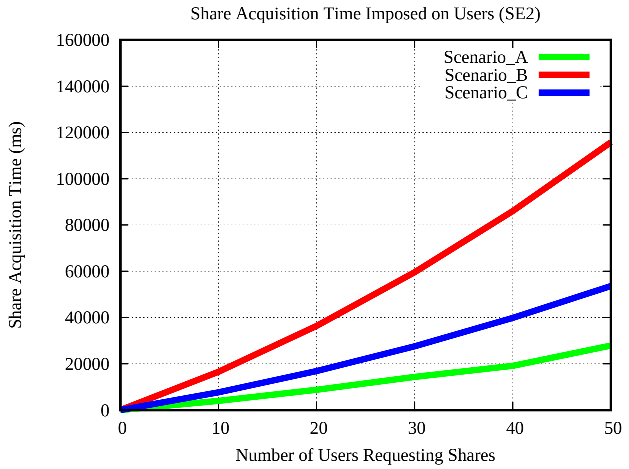

Figure 2. Share Acquisition Time Imposed on Users (SE2).

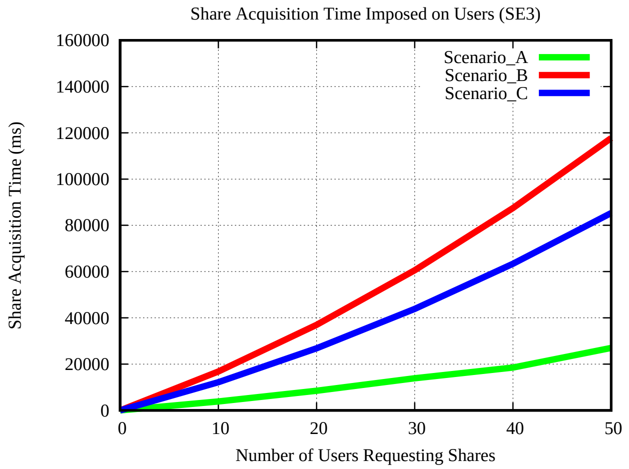

Figure 3. Share Acquisition Time Imposed on Users (SE3).

It can be observed that the results for all the three scenarios increase steadily as the number of users requesting shares increases, though the increase for Scenario_B is steeper in the three figures. In Figure 1, we can observe the results of Exp-1 using SE1. In this setting, most users belong to G1 user-group. The results of Scenario_C (i.e., the 3LAC Framework) are closer to Scenario_A. The reason is that Scenario_A is when 1 acquisition is required, and Scenario_C contains a majority number of G1 users. The cause of the difference between Scenario_A and Scenario_C is due to the $30\%$ of the G2 users and $10\%$ of the G3 users. In other words, $60\%$ of the requests performed in Scenario_C are similar to the requests performed in Scenario_A in which only 1 acquisition is required. In Figure 2, we can observe the results of Exp-1 when using SE2. In SE2, most users are in G2 user-group. The trend in this graph is similar to that in Figure 1, with an exception that the results of Scenario_C in this case are not as close to Scenario_A as they were in Figure 1. The reason is that in SE2 most users belong to G2 user-group, which means that for each of these users, two requests are needed. This increases the acquisition time for Scenario_C in SE2 as compared to the acquisition time of Scenario_C in SE1. However, the acquisition time in Scenario_C is markedly smaller than in Scenario_B, where three shares acquisitions are always needed. In Figure 3, we can observe the results of Exp-1 when using SE3. In SE3, most users belong to G3 user-group. We can observe that the trend in this graph is similar to that in Figure 1 and Figure 2, respectively, with an exception that in this case Scenario_C is steeper and also the results of Scenario_C are closer to the results of Scenario_B. The reason is that in SE3 most users are in G3 user-group, which means they need three shares. This is similar to Scenario_B in which three shares are always needed. In addition, we further investigated the share acquisition time imposed on a user vs. the number of users requesting shares in Scenario_C, as displayed in Figure 4.

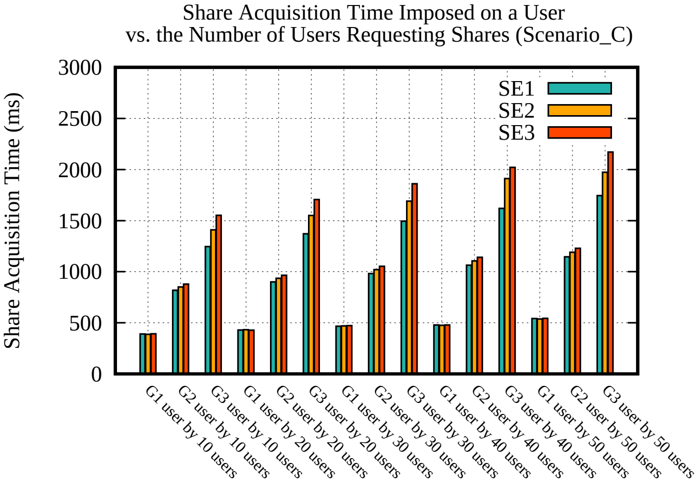

Figure 4. Share Acquisition Time Imposed on a User vs. the Number of Users Requesting Shares (Scenario_C).

We can observe in Figure 4 that the share acquisition time imposed on a user vs. the number of users requesting shares for G1 users is similar in SE1, SE2, and SE3 (when having the same number of users). The reason is that a G1 user sends a request to $LKA_1$ only, and $LKA_1$ receives one request from each user regardless of what setting is used. In other words, even when using SE2 (most users in G2) or SE3 (most users in G3), the total number of requests received by $LKA_1$ is the same. For G2 users, the share acquisition time imposed on a user increases by 4% from SE1 to SE2, and by 3% from SE2 to SE3. The reason is that a share acquisition time imposed on a G2 user relies on the service time of $LKA_1$ and $LKA_2$. In SE2, $LKA_2$ receives more requests than in SE1, and in SE3, $LKA_2$ receives more requests than in SE2. Share acquisition time imposed on a G3 user increases by 12% from SE1 to SE3, and by 10% from SE2 to SE3. The reason is that the share acquisition time imposed on a G3 user relies on the service time of $LKA_1$, $LKA_2$ and $LKA_3$. In SE2, $LKA_2$ receives more requests than in SE1, and in SE3, $LKA_3$ receives more requests than in SE2. When comparing G1, G2, and G3 users in Scenario_C against users in Scenario_A and Scenario_B, we found that the share acquisition time imposed on a user in Scenario_B is 35% more than the share acquisition time imposed on a G3 user in Scenario_C. The reason is that in Scenario_B all users need to request three shares and $LKA_3$ receives a request from each user. However, in Scenario_C only the G3 users need to send a request to $LKA_3$. For this reason, the service time imposed on $LKA_3$ is more in Scenario_B than in Scenario_C. We also found that the share acquisition time imposed on a G1 user in Scenario_C is similar to the share acquisition time imposed on a user in Scenario_A. The reason is that in Scenario_A each user sends a request to $LKA_1$ only, and in Scenario_C, a G1 user sends a request to $LKA_1$ only. Figure 5 shows the peak values for a G1, G2 and G3 user in Scenario_C, and the peak values for a user in Scenario_A and Scenario_B, respectively.

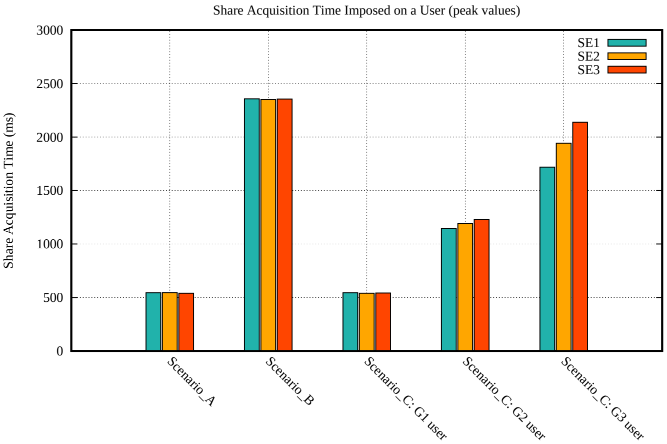

Figure 5. Share Acquisition Time Imposed on a User (peak values).

In SE1, Scenario_C (i.e., the 3LAC Framework) is 93% more efficient than Scenario_B. In SE2, Scenario_C is 73% more efficient than Scenario_B. In SE3, Scenario_C is 32% more efficient than Scenario_B. To summarise, the extra acquisition time when SE3 is used as compared to the acquisition time when using SE1 and SE2 is due to the following reasons:

- The extra communications between the user and the $LKAs$. When SE3 is used, a user needs to send more requests to $LKA_1$, $LKA_2$ and $LKA_3$ in comparison against SE2, in which the number of requests to $LKA_3$ decreases as the acquisition of a third share is not needed for most users in SE2. Similarly, when using SE1, the number of requests to $LKA_2$ and $LKA_3$ decreases as the acquisition of a second and third share is not needed for most users in SE1.
- The extra computations in the nonce verifications by both the user and the $LKAs$. As in SE3, most users need to obtain a third share, it means an extra nonce verification is required by $LKA_3$ as compared against SE2 and SE1, in which a nonce verification performed by $LKA_3$ is not required when a third share is not needed. Also, as the user communicates with more $LKAs$, it also involves more nonce verifications on the user's side to verify the nonce received by a $LKA$.
- The extra computations in the digital certificate verifications by both the user and the $LKAs$. As in SE3 more $LKAs$ are involved, this also means a digital certificate verification performed by each $LKA$ that receives a request, and the user that receives a share.
- The extra computations in the signature verifications by both the user and the $LKAs$. In SE3 more $LKAs$ are involved, then more signature verifications are performed as each $LKA$ that receives a request has to perform this verification. Also, on the user's side, the user has to perform a signature verification of the $LKA$ that the user is communicating with.

## VII. Conclusion and Future Work

In this paper, we have presented the design of a Novel 3-Level Access Control Framework (3LAC). The 3LAC Framework supports multiple levels of access privileges, i.e., (L1) access to low sensitive data, (L2) access to medium sensitive data, and (L3) access to high sensitive data. Also, this paper has described the architecture used in the design of the 3LAC Framework and its functional components. The 3LAC architecture is divided into AQ1: Architecture for Key Generation and Distribution, and AQ2: Architecture for Data Uploading and Access. The experiments conducted have shown that the 3LAC Framework balances the level of protection given to data in response with the different data sensitivity levels. Future work includes the consideration of contextual information about a user, such as access history and location. These may be used as factors to estimate the level of risk involved in an access request, in which a high level of risk may involve a more rigorous authentication process. Future work also includes the assessment of network delays and the evaluation of how they affect the performance of the 3LAC Framework.

## Acknowledgment

This research is supported by the National Council of Science and Technology Mexico (CONACYT).

## References

[1] Y. B. Choi, K. E. Capitan, J. S. Krause, and M. M. Streeper, "Challenges associated with privacy in health care industry: implementation of hipaa and the security rules," Journal of Medical Systems, vol. 30, no. 1, 2006, pp. 57–64.

[2] P. C. Tang, J. S. Ash, D. W. Bates, J. M. Overhage, and D. Z. Sands, "Personal health records: definitions, benefits, and strategies for overcoming barriers to adoption," Journal of the American Medical Informatics Association, vol. 13, no. 2, 2006, pp. 121–126.

[3] I. Carrión, J. L. F. Alemán, and A. Toval, "Assessing the hipaa standard in practice: phr privacy policies," in Engineering in Medicine and Biology Society, EMBC, 2011 Annual International Conference of the IEEE. IEEE, 2011, pp. 2380–2383.

[4] M. Rouse, "American health information management association (ahima)," http://searchhealthit.techtarget.com/definition/American-Health-Information-Management-Association-AHIMA, 2017 [retrieved: 08, 2018].

[5] J. Wolter and B. Friedman, “Health records for the people: touting the benefits of the consumer-based personal health record,” Health Records for the People: Touting the Benefits of the Consumer-based Personal Health Record/AHIMA, American Health Information Management Association, 2005.

[6] D. Wiljer et al., “Patient accessible electronic health records: exploring recommendations for successful implementation strategies,” http://www.jmir.org/2008/4/e34/, 2008 [retrieved: 08, 2018].

[7] N. M. Shrestha, A. Alsadoon, P. W. C. Prasad, L. Hourany, and A. Elchouemi, “Enhanced e-health framework for security and privacy in healthcare system,” in 2016 Sixth International Conference on Digital Information Processing and Communications (ICDIPC), April 2016, pp. 75–79.

[8] C. Techapanupreeda and R. Chokngamwong, “Accountability for electronic-health systems,” in 2016 IEEE Region 10 Conference (TENCON), Nov 2016, pp. 2503–2506.

[9] P. Thummavet and S. Vasupongayya, “A novel personal health record system for handling emergency situations,” in 2013 International Computer Science and Engineering Conference (ICSEC), Sept 2013, pp. 266–271.

[10] P. Ray and J. Wimalasiri, “The need for technical solutions for maintaining the privacy of ehr,” in 2006 International Conference of the IEEE Engineering in Medicine and Biology Society, Aug 2006, pp. 4686–4689.

[11] A. Alabdulatif, I. Khalil, and V. Mai, “Protection of electronic health records (ehrs) in cloud,” in 2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), July 2013, pp. 4191–4194.

[12] R. Zhang and L. Liu, “Security models and requirements for healthcare application clouds,” in 2010 IEEE 3rd International Conference on Cloud Computing, July 2010, pp. 268–275.

[13] D. Ferraiolo, J. Cugini, and D. R. Kuhn, “Role-based access control (rbac): features and motivations,” in Proceedings of 11th annual computer security application conference, 1995, pp. 241–48.

[14] D. Boneh and M. K. Franklin, “Identity-based encryption from the weil pairing,” Proceedings of CRYPTO 2001 on Advances in cryptology, vol. 32, no. 3, 2001, pp. 586–615.

[15] A. Sahai and B. Waters, “Fuzzy identity-based encryption.” in Eurocrypt, vol. 3494. Springer, 2005, pp. 457–473.

[16] Oracle, “Java enterprise edition,” Journal of Technology, vol. 2, 2012, pp. 1–18.

[17] J. Letkowski, “Doing database design with mysql,” Journal of Technology Research, vol. 6, 2014, pp. 1–15.