BLAC Implementation on Different Use Cases in Healthcare Systems

by

Sarada Mantha

A Project Report Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science in Computer Science

Supervised by

Dr. Rajendra K. Raj

Department of Computer Science

B. Thomas Golisano College of Computing and Information Sciences Rochester Institute of Technology Rochester, New York

August 2016
Dedication

To whom I owe my everything - My parents
Acknowledgments

I am grateful to my advisor, Dr. Rajendra K. Raj for the guidance and wisdom he has provided throughout my MS program. His support and guidance have helped me in achieving my academic accomplishments.

I am also thankful to Dr. Hans-Peter Bischof for his constant encouragement.

Finally, I am grateful and thankful to my parents for believing in me.
Abstract

BLAC Implementation on Different Use Cases in Healthcare Systems

Sarada Mantha

Supervising Professor: Dr. Rajendra K. Raj

A healthcare system consists of high amount of confidential data. This data requires protection against several malicious attacks, insiders threats, and many other unknown threats. Sharing and maintaining healthcare information is one important task. With the proposal of BiLayer Access Control (BLAC) by Dr. Rajendra K. Raj, Dr. Sumita Mishra, and Suhair Alshehri, for healthcare systems to protect them against insider threats, it is important to ensure that the model is capable of handling all the different use-cases that make a system sustainable. This paper will be able to predict the effectiveness of a BiLayer Access Control model in a Healthcare system by implementing over different use cases. A healthcare system consists of five use cases which are also known as functions. These use cases are the core functions that any organization must incorporate. The performance of the model is highly dependent on the implementation of these use cases. Each use case will be developed over the existing data. A final analysis report will be constructed to predict the efficiency of the model.
# Contents

Dedication ................................................................. ii

Acknowledgments ....................................................... iii

Abstract ................................................................. iv

1 Introduction .......................................................... viii

2 Design and Implementation ........................................ xi

3 Analysis ............................................................... xviii
   3.1 Why BLAC in Healthcare? ..................................... xix

4 Conclusions .......................................................... xxi
   4.1 Current Status .................................................. xxi
   4.2 Future Work ..................................................... xxii
   4.3 Lessons Learned ................................................ xxiii

Bibliography ............................................................. xxv

A Code Listing .......................................................... xxvi
List of Tables

3.1 Comparison of RBAC, ABAC and BLAC . . . . . . . . . . . . . . . xix
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>The BLAC model representation as a 2-step procedure</td>
<td>xv</td>
</tr>
<tr>
<td>2.2</td>
<td>Data flow architectural design</td>
<td>xvi</td>
</tr>
<tr>
<td>2.3</td>
<td>Policy 1</td>
<td>xvi</td>
</tr>
<tr>
<td>2.4</td>
<td>Policy 2</td>
<td>xvi</td>
</tr>
<tr>
<td>2.5</td>
<td>Policy 3</td>
<td>xvii</td>
</tr>
<tr>
<td>2.6</td>
<td>Policy 4</td>
<td>xvii</td>
</tr>
<tr>
<td>3.1</td>
<td>Analysis of five use cases representing the time taken in ms</td>
<td>xx</td>
</tr>
<tr>
<td>4.1</td>
<td>Possibility of having a new improvised BLAC model</td>
<td>xxiii</td>
</tr>
<tr>
<td>A.1</td>
<td>Subject information sample</td>
<td>xxvi</td>
</tr>
<tr>
<td>A.2</td>
<td>This is the decisionMaker class which is responsible for the two-step pro-</td>
<td>xxvii</td>
</tr>
<tr>
<td></td>
<td>cedure of BLAC model (sample code)</td>
<td></td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

A healthcare system consists of high amount of confidential data. This data requires protection against several malicious attacks, insiders threats, and many other unknown threats. Sharing and maintaining healthcare information is one important task. With the proposal of BiLayer Access Control (BLAC) by Dr. Rajendra K. Raj, Dr. Sumita Mishra, and Suhair Alshehri, for healthcare systems to protect them against insider threats[2], it is important to ensure that the model is capable of handling all the different use-cases that make a system sustainable. This paper will be able to predict the effectiveness of a BiLayer Access Control model in a Healthcare system by implementing over different use cases.

A healthcare system consists of five use cases which are also known as functions[1]. These use cases are the core functions that any organization must incorporate. The performance of the model is highly dependent on the implementation of these use cases. Each use case will be developed over the existing data. A final analysis report will be constructed to predict the efficiency of the model.

With several access control models being introduced and tested, it was found that RBAC and ABAC were determined to be the efficient access control models for several industries[4]. Each of the models has their respective pros and cons. RBAC makes use of roles but on healthcare industries [7], if a particular subject requires permission keeping role based access control it would pass the authority to several other subjects defined in the same role. Even though the access is limited at role level, it does give rise to insider threats. ABAC, on the other hand, has a better working mechanism where it implies the usage of only policies. But, a creation of policies involved high complexity. Every subject
was entitled to a 2n number of policies which was not a considerate number to look at in respect to commercial industries[2].

With further analysis and detailed research, it was learned that these access control models showcase limitations which otherwise can be handled. With growing data, it was found that maintaining security with certain drawbacks would not be a suitable option. NIST proposed BiLayer Access Control (BLAC) [7], which is a definite result of RBAC and Attribute based access control that is capable of handling the limitations of RBAC and ABAC at the same time. Health care systems contain an enormous amount of data. This data is highly confidential since it contains sensitive information about various patients, users, customers, etc[3]. It is important for healthcare providers to maintain their data privacy and at the same time ensure there is security against insider threats. Although there are many access control models implemented in various fields of industries, since privacy in healthcare systems is of high priority, BiLayer Access Control (BLAC) model was proposed as one of the suitable models.

RBAC and ABAC are two of the common access control models which have their implementation in various fields across different industries. RBAC is capable of controlling access based on roles[7]. However, a significant number of subjects get privilege at the same time. ABAC gives access based on attributes which associated with respective policies. This makes ABAC highly sophisticated. On the other hand, BLAC is designed to support attributes and policies along with maintaining advantages of RBAC. With the earlier work of implementation of BLAC in healthcare systems, it was regarded to be appropriate for healthcare providers as it was able to mitigate insider threats at a certain level[7]. Analyzing the efficiency of the model was also obtained as one of the primary objectives. This project aims to move provide further detailed analysis of BLAC and its effectiveness on different use cases in Healthcare systems. These use cases are considered to be the basic functions of any healthcare systems that would complete the life-cycle[2].

Implementation of access control in healthcare systems is essential to keep the security
of the system enact. BLAC implementation is still in the developing phase. There is continuous research going on BLAC[4]. An analysis report was generated where the studies have shown that BLAC was comparatively better in performance than RBAC, ABAC, and few other access control models[1]. One of RIT's Ph.D. student, Suhair Alshehri, worked on the implementation of BLAC in healthcare systems where she was able to derive the five use cases essential for a health system[1]. The important of this model can be predicted based on the performance and analysis obtained at the end. It is important to ensure that healthcare data is highly secured against insider and outsider threats at all time. Giving authority to subjects based on their roles and restricting their access based on policies was thoroughly implemented through BLAC model.

The hypothesis for my project is to perform a theoretical analysis based on a practical implementation of BLAC on different use cases in a healthcare system. Thus, being able to predict the efficiency of the model for a healthcare system. To do so, I will generate and develop new policies and pseudoroles as required for the respective use case. Generate necessary implementation code and develop the model to make it more efficient in its performance. These use cases are also known to be the primary functions of a healthcare system, that complete the life-cycle of the entire system. The following are the important use cases for a healthcare system providers:

- Subject requests access
- Subject permissions are modified
- Subject leaves the system
- Subjects join the system
- Organizational policies are modified
Chapter 2

Design and Implementation

BLAC model is a two-step procedure as can be understood from figure 2.1. And every step requires certain attribute checking to be done. The implementation is broken down into several stages. Since this project concentrates on handling the five different use cases, it is important to understand each one of them respectively[1]. Having learned and understood about BLAC, it is important to know that getting healthcare data is highly restricted. Although attempts were made to generate and store the data in a database for the project, it was not quite successful. So, based on the information gathered in the past, data has been generated and stored in the form of XML files, which are then parsed for access through Java program. Figure 2.2 explains the data flow design of the implementation where there are four distinct types of data files in the form of XML files. These files are parsed to a Java code where the execution performs the necessary operations.

The implementation of a basic BLAC model was successful generated, where new policies and pseudoroles were developed based on the algorithm proposed by Suhair Alshehri[1]. Every use case had respective criteria that had to be met, and therefore a particular policy was generated to support the use-case. For the given five use cases as stated above, there were four different policies generated to support the functioning of the system.

Having learned and understood about BLAC, it is important to know that getting healthcare data is highly restricted. Although attempts were made to generate a database for the project, it was not quite satisfied[2]. So, based on the information gathered in the past, data has been generated and stored in the form of XML files, which are then parsed for access through Java program. Figure 2 explains the data flow design of the implementation where
there are four distinct types of data files in the form of XML files. These files are parsed to a Java code where the execution performs the necessary operations.

The implementation of a basic BLAC model was successful generated, where new policies and pseudoroles were developed based on the algorithm proposed by Suhair Alshehri[3][1]. Every use case had several criteria that had to be met, and therefore a particular policy was generated to support the use-case. For the given five use cases as stated above, there were four different policies produced to support the functioning of the system. BLAC has two important relationships. First, pseudoroles generated are associated with subjects. Second, policies made are associated with objects respectively. Definition of a subject for this particular project is defined as anything that requires the permission to access data or create data, etc. And the object in a healthcare system represents any resource that contains information or data, and it requires special permissions to access this information.

Every time a subject or an object join the healthcare system, respective attributes and information are created by adding the necessary information to the system. An object ID gets associated with a respective subject identifier, in other words, every patient is associated with one particular physician in the initial phase. Sometimes, these patients may require superior physicians or diagnosis from a different department, that is when further policies are generated to handle the respective situations. The main reason for these policies is to restrict authorization of random members from accessing information of patients. RBAC, in this case, was proven to be less protective in security aspects. Since, when a particular role is given the authorization, every subject having the respective role gets access to information or data. This does result in possible insider threats. This was then handled in a different way when NIST proposed ABAC[1]. Where, permissions are granted based on attributes but required policies to be generated which further increased the complexity respectively.

The main purpose of these use cases is to show that BLAC model can perform the necessary and basic functions of an organizational system that would complete the lifecycle of any access control model[1].
Subject joins the system: This use case requires the generation of new attributes and pseudoroles when a new employee is joining the organization. Every new member gets an ID generated and the associated object ID. The initial entry of data into any system certainly requires restrictions and thus policy 1 explains the rules clearly which can be observed from figure 2.3.

Subject requests access: On the implementation, this appeared as a default case where the inputs are received by the user through the terminal, and the respective operation is performed based on the request being granted or not. In certain cases, where information needs to be shared based on the request proposed, policy 2 states the exact set of permissions based on which any access must be granted. Figure 2.4 explains policy 2 in a healthcare system.

Organizational policies are modified: This requires special administrative permissions to pass certain policy changes. In the implementation provided, an option is given to the user to determine if an organizational policy is to be implemented. If yes, then based on the default settings, changes will be made and the code executed will prompt for an overall modification of policy. Although handling of dynamic attribute changes is difficult to incorporate, efforts were made to understand the use-case. The changing of policy for a certain scenario can be understood better from figure 2.5.

Subjects permissions are modified: Read/Write/ReadWrite are the only granted permissions to every subject which are modified upon request by the user. Based on the subject ID, Java along with XML parsing libraries allows commands to modify a respective attribute based on the search pattern. Figure 2.6 is one such example of the policy as derived by Suhair [1].

Subjects leave the system: This is one of the easiest use case that was handle by the given implementation.
The complete execution of BLAC inclusive of all possible combinations was not a hassle but there were several hurdles. Collection of data, combining several information, generating pseudoroles and policies according to the respective user information was quite challenging. Peudorole algorithm as proposed by Suhair was very helpful in this procedure. Based on the use cases and their respective rules, four significant policies were created which are associated with the healthcare systems. These were carefully analyzed and studied as they were proposed by Suhair[1].
Figure 2.1: The BLAC model representation as a 2-step procedure
Figure 2.2: Data flow architectural design

POLICY 1

```xml
<Policy>
  <PseudoRole>
    (subject.provider="physician" V subject.provider="nurse") \ 
    subject.provider="primaryCare" \ 
    subject.hospital = "abca">
  </PseudoRole>
  <Rule>
    <Subject>"any"</Subject>
    <Object>object.providerID=subject.ID</Object>
    <Operation><action.type="read" V action.type="modify"></Operation>
    <Environment><environment. AccessIP="192.123.***"></Environment>
  </Rule>
</policy>
```

Figure 2.3: Policy 1

POLICY 2

```xml
<Policy>
  <PseudoRole>
    (subject.provider="physician" V subject.provider="nurse") \ 
    (subject.department="dermatology" V subject.provider="primaryCare") \ 
    subject.hospital = "abca">
  </PseudoRole>
  <Rule>
    <Subject>"any"</Subject>
    <Object>object.providerID=subject.ID</Object>
    <Operation><action.type="read" V action.type="modify"></Operation>
    <Environment><environment. AccessIP="192.123.***"></Environment>
  </Rule>
</policy>
```

Figure 2.4: Policy 2
Figure 2.5: Policy 3

POLICY 3

<Policy>
  <PseudoRole/>
  <(subject.provider = "physician" V subject.provider = "nurse") ∧
   subject.department = "any" ∧
   subject.hospital = "abca">
  </PseudoRole>
  <Rule>
   <Subject>"any"</Subject>
   <Object>"any"</Object>
   <Operation><action.type = "read" V action.type = "modify"></Operation>
   <Environment><environment.AccessIP = "192.123.4.*"></Environment>
  </Rule>
</Policy>

Figure 2.6: Policy 4

POLICY 4

<Policy>
  <PseudoRole/>
  <(subject.provider = "physician" V subject.provider = "nurse") ∧
   (subject.department = "dermatology" V subject.department = "primaryCare") ∧
   subject.hospital = "abca">
  </PseudoRole>
  <Rule>
   <Subject>subject.provider = "physician"</Subject>
   <Object><object.providerID = subject.ID></Object>
   <Operation><action.type = "read" V action.type = "modify"></Operation>
   <Environment><environment.AccessIP = "any"></Environment>
  </Rule>
  <Rule>
   <Subject>"any"</Subject>
   <Object><object.providerID = subject.ID></Object>
   <Operation><action.type = "read" V action.type = "modify"></Operation>
   <Environment><environment.AccessIP = "192.123.4.*"></Environment>
  </Rule>
</Policy>
Chapter 3

Analysis

The main disadvantage of RBAC is that, when permission for access is granted to a particular role, an entire group of subjects is given privilege, irrespective of whether they are directly or indirectly related to that case. This may cause irregular usage of data in case of sensitive information. When Healthcare systems are to be considered sharing sensitive patients information with such group of people can cause security concerns.

ABAC is regarded as a superior access control model in comparison to RBAC, but it also has its tradeoffs. Considering that for every n number of attributes in a database, an ABAC policy would require at least 2n number of policies which would be highly complex for big data to handle.

BLAC as proposed by Alshehri and Dr. Raj[1], is a fair result of ABAC and RBAC combined in a way where the access control model generated produces a secure model, given that ABAC and RBAC both have their limitations. NIST tried to add attributes to RBAC to provide a better access control while maintaining the advantages of RBAC [5]. Thus, tried to connect RBAC with ABAC. While maintaining the benefits of both the access controls individually, a set of three different mechanisms were developed [7]

- Dynamic roles
- Attribute centric
- Role centric

It was observed that all the three mechanisms had their own drawbacks which motivated the development of BLAC.
3.1 Why BLAC in Healthcare?

From the paper written together by Suhair Alshehri and Dr. Rajendra K. Raj titled Secure Access Control for Health Information Sharing Systems[3], a thorough analysis has been made on five access control models, RBAC, ABAC, ABP-RBAC, RABAC al. and BLAC, with respect to different characteristics[7]. These characteristics have been carefully selected by analyzing the feature of every access control to its best of performance. I would like to refer to the same table as in the paper for my personal inferences and also from my independent study report[6].

<table>
<thead>
<tr>
<th>Characteristics</th>
<th>RBAC</th>
<th>ABAC</th>
<th>BLAC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flexibility</td>
<td>Low</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>Granularity</td>
<td>Low</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>Authorization Complexity</td>
<td>Static</td>
<td>Dynamic</td>
<td>Dynamic</td>
</tr>
<tr>
<td>Privilege Modifiability</td>
<td>Simple</td>
<td>Complex</td>
<td>Complex</td>
</tr>
<tr>
<td>Permission Modifiability</td>
<td>Simple</td>
<td>Complex</td>
<td>Complex</td>
</tr>
<tr>
<td>Revocability</td>
<td>Simple</td>
<td>Complex</td>
<td>Simple</td>
</tr>
<tr>
<td>Permission Reviewability</td>
<td>Simple</td>
<td>Complex</td>
<td>Simple</td>
</tr>
<tr>
<td>Setup Complexity</td>
<td>Complex</td>
<td>Simple</td>
<td>Simple</td>
</tr>
</tbody>
</table>

With the studies and research conducted it is understandable that BLAC is a better model for healthcare systems. Based on the work put together BLAC model implemented was able to handle insider threats. Implementation of BLAC on the five use cases was challenging.

A systematic approach followed the analysis of the implementation. Every use case was executed at least five times before the average time for execution was noted. During the execution of the model, certain errors were encountered due to mishandling of attribute information. With careful debugging, exceptions and errors were handled, and the code was able to execute.

It was found that one of the use cases, ”Organizational policies are modified” was quite a challenge. The function was trying to implement required dynamic operation of attributes
which was considered to be difficult due to certain constraints. With time and space constraints, I tried to run the implementation on a dummy file and was able to note the time stamps. Since this implementation is void of any database for storage of data, it would be extremely time-consuming to redo the data if there were any mishaps of data loss during the execution.

With the given executions I have obtained a graphical representation of the same to provide better understanding of the theoretical analysis. Figure 3.1 clearly outlines the time taken by each use to execute.

![Bar Chart](image)

**Figure 3.1: Analysis of five use cases representing the time taken in ms**
Chapter 4

Conclusions

BLAC model used for the implementation based on theoretical analysis and discussions is considered to be a better model in comparison to RBAC and ABAC. Use of BLAC in healthcare data systems is shown to be effective since it is capable of handling insider threats. With the limitations learned about RBAC and ABAC, it is suggested to implement BLAC as the suitable access control into every healthcare provider systems. Attempts were made to prove that BLAC can be a successful implementation due to the performance along with the additional handling of use cases in health systems. Considering all necessary implementations and security facilities must be in accordance to HIPAA.

Currently, the implementation does not support any database storage facilities. With the minimum available data available regarding XML files, the model was able to execute. Although all use cases were not developed successfully with theoretical analysis, they can be achieved in the future as well.

4.1 Current Status

The current state of this project is very crucial to determine a particular statement, since it efforts were made and it was capable of handling insider threats, four use cases were treated, and the execution was found to be successful. Out of five use cases, only four were successfully developed and were put to the test. The use case that required more concentration was the change of organizational policies which required a lot of efforts in implementation and a necessary database to store the required data carefully. Since the
current project is not equipped with a data bank, it is risky to implement the changes at the cost of the entire data. This use case makes use of dynamic generation of attributes, which is not yet being implemented and thus was not a complete success. Although it has been found that the implementation of the use case can make advance changes to BLAC overall.

4.2 Future Work

BLAC being in the development mode creates a whole new space for research and development. Various aspects can be investigated, and work can be progressed accordingly. For this particular BLAC model and based on the current status, a healthcare database can be generated. The current data consists of a subject and patient data which can be utilized for the database. However, it must be understood that obtaining health care data is highly challenging considering that it contains sensitive and personal information about patients. This information can either be anonymized and obtained or modified based on certain results. This will certainly help future projects highly concentrating in healthcare facilities. Building a database of the required information can be a definite task to achieve in the future.

The current BLAC model is a two-step procedure. Where the first is to verify for respective pseudoroles and the second step involves checking of respective policies. With the present model, insider threats have been handled. There can still be scope for an additional layer of checking involved in the procedure to make the entire process highly complex for trespassers and certainly provides the system with an improvised version of BLAC. A modified version of BLAC by adding checking step can be included to improve the performance. One of the use case - Organizational policies are modified, is found to be difficult. Work can be done in this direction for the future. For instance adding a single point access check as shown in figure 4.1, after the second step is processed can be effective in detecting unauthorized access at all times to all aspects of threats.
4.3 Lessons Learned

During the project phase, several mistakes were made and new lessons were learned as well. The importance of any implementation consisting of huge data and not having any database was one challenging task. Although the execution was not a hassle but there were several hurdles. Collection of data, combining several information, generating pseudoroles and policies according to the respective user information was quite challenging. Peudorole algorithm as proposed by Suhair was very helpful in this procedure.
While developing the model for the working of various use cases, it was learned that trying to make it real-time is difficult. Modification of subjects permissions was manageable, but developing organizational policy changes was a difficult task. Technical difficulties faced was on the memory storage and time complexity. Modification of policies involved a high number of complications. The user’s input was not able to be matched, and the respective changes did not take place. At this stage, this use case was not able to be a successful one. However, with time and memory constraints, this can be looked for the future projects. Handling of use cases gave rise to several unknown exceptions, which were handled later during the debugging phase.
Bibliography


Appendix A

Code Listing

The following figures give a glance into the BLAC implementation.

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<hospital>
  <subject>
    <subjectId>1665081326995</subjectId>
    <firstname>Addison</firstname>
    <lastname>Weeks</lastname>
    <provider>physician</provider>
    <department>OB/GYN</department>
    <location>A</location>
  </subject>
</hospital>
```

Figure A.1: Subject information sample
package decisionMaker;
import readPatientsData.readFile[

public class decisionMaker {

    public static void main(String args[]) {

        Scanner subjectID = new Scanner(System.in);
        System.out.println("Enter the subject id");
        String sid = subjectID.next();
        Scanner objectID = new Scanner(System.in);
        System.out.println("Enter the object id");
        String oid = objectID.next();
        Scanner actionRequest = new Scanner(System.in);
        System.out.println("Enter the action you want to perform. r/w/rw");
        String action = actionRequest.next();
        SubjectPolicies sp = new SubjectPolicies();
        String pseudorole = sp.fetchPseudorole(sid);
        System.out.println("The pseudorole is: " + pseudorole);

        // first stage check for pseudorole
        long startTime = System.currentTimeMillis();
        firstStageCheck fc = new firstStageCheck();
        boolean firstCheck = fc.check(pseudorole);
        System.out.println("First check ? " + firstCheck);
        boolean finalCheck = false;

        if(firstCheck == true){
            secondStageCheck ssc = new secondStageCheck();
        }
    }
}