ACCESS AND INFORMATION FLOW CONTROL TO SECURE MOBILE WEB SERVICE COMPOSITIONS IN RESOURCE CONSTRAINED ENVIRONMENTS

Lwazi E Maziya

A dissertation submitted

In partial fulfillment of the

Requirements for the degree of

Master of Science in

Computer Science

Department of Computer Science

Faculty of Science

University of Cape Town

Supervised by

Dr. Anne Kayem

January 2015

Abstract

The growing use of mobile web services such as electronic health records systems and applications like twitter, Facebook has increased interest in robust mechanisms for ensuring security for such information sharing services. Common security mechanisms such as access control and information flow control are either restrictive or weak in that they prevent applications from sharing data usefully, and/or allow private information leaks when used independently. Typically, when services are composed there is a resource that some or all of the services involved in the composition need to share. However, during service composition security problems arise because the resulting service is made up of different services from different security domains. A key issue that arises and that we address in this thesis is that of enforcing secure information flow control during service composition to prevent illegal access and propagation of information between the participating services. This thesis describes a model that combines access control and information flow control in one framework. We specifically consider a case study of an e-health service application, and consider how constraints like location and context dependencies impact on authentication and authorization. Furthermore, we consider how data sharing applications such as the e-health service application handle issues of unauthorized users and insecure propagation of information in resource constrained environments[1]. Our framework addresses this issue of illegitimate information access and propagation by making use of the concept of program dependence graphs (PDGs). Program dependence graphs use path conditions as necessary conditions for secure information flow control. The advantage of this approach to securing information sharing is that, information is only propagated if the criteria for data sharing are verified.

Our solution proposes or offers good performance, fast authentication taking into account bandwidth limitations. A security analysis shows the theoretical improvements our scheme offers. Results obtained confirm that the framework accommodates the CIA-triad (which is the confidentiality, integrity and availability model designed to guide policies of information security) of our work and can be used to motivate further research work in this field.

[1] Resource constrained environments are characterized by infrastructural limitations that impact negatively on computation efficiency and effectiveness. Examples arise in disaster management scenarios, war-torn zones, and rural or remote areas.

Information Flow Control to

Secure Mobile Web Service Composition

by

Lwazi Maziya

Plagiarism Declaration

I know the meaning of plagiarism and declare that all the work in the document, save for which is properly acknowledged, is my own.

Signed by candidate

Lwazi Maziya

Friday January 30, 2015

(Date)

****

Acknowledgements

First and foremost I would like to extend my appreciation and sincere gratitude to my supervisor Dr. Anne Kayem for her constant support, encouragement, and patience throughout this entire Masters process. I simply could not have reached this point without her and I will forever carry an unwavering appreciation and debt of gratitude for her efforts on my behalf. Thank you for the opportunity you forwarded to me to work with you, for pushing me and helping me through every gate. If I'm a better researcher and writer today, it is all because of your unwavering insightful guidance and support.

I would also like to thank A/Professor Andrew Hutchison for being a wonderful inspiration through his work and teachings. Thank you for your kind support and advice during the earlier months of the study. When I make it to the top, know your contribution has been insurmountable. Thanks to all my fellow graduate students who helped me in my research and prototyping efforts, I know I wouldn't have made it thus far without your inputs.

The Information Security research group, thank you for the time, moments and experiences we've shared together in our meetings and social gatherings. Your valuable suggestions, constructive criticism and encouragement have not gone unnoticed along the development of my work. This one is for all the insights and ideas that were borne out of those meetings to make this work a remarkable end product.

Finally I thank my family for sacrificing in different ways for me to be able to achieve this dream. Please know I tried my best to lessen the burden on you whilst undertaking this endeavor.

My final debt of gratitude goes to my lovely and dear wife Dumisile, know I appreciate you more than I can say and I can never make up for everything you've done to support me over the years, but I will try with everything I have. Thank you for keeping up with the two little rascals on your own.

For Dumisile, S'viwe, Yenziwe and Robyn.

Table of Contents

List of Figures

Chapter 1
Introduction

Rapid advances in web service technologies, has generated more interest for applications to be available for users wherever they are. However, in a developing world context, the provisioning of these web services is heavily reliant on the type of computing devices available to the vast majority of users. Hence, due to the scarcity of big resource computing devices like personal computers (PCs), mobile devices like cellular or smart phones and tablets become a default solution in these (developing) environments. However, these mobile devices face a challenge of being resource constrained compared to their traditional counterparts like personal computers. Constraints like computational power and bandwidth bring about their limitations as far as performance related functions are concerned. This chapter presents the background and motivation of studying access and information flow control to secure mobile web service compositions in resource constrained environments and the challenges they face in as far as adhering to security policies is concerned. It also presents the contributions of the work and finally the structure or layout of the thesis.

1.1 Background and Motivation

The evolution of the Internet in recent years has sparked a growing popularity for information sharing applications. Information sharing applications allow users to interact without necessarily needing to store data on the device used for communication. Furthermore, applications such as Facebook[2], MySpace[3], and Flickr[4] support this idea of information availability anytime and anywhere.

The Service Oriented Architecture (SOA) concept and more recently, Cloud computing, provide a framework for building applications to enable information sharing across multiple security domains. In SOA environments, services can be composed to form new applications, by

[2] www.facebook.com
[3] www.myspace.com
[4] www.flickr.com

combining the software functionalities of services belonging to different domains. An added advantage of service compositions is that information can be combined from different sources, flexibly, using a set of policies (rules) to respond to queries.

Typically, service composition becomes necessary when there is a resource that some or all of the services involved in the composition need to share. For instance, consider an electronic health records system that consists of three outsourced databases namely, “chronic illnesses”, “allergies”, and “patient details” (e.g. date of birth, race and gender). Access to this data needs to be flexible so that queries can return responses without revealing any more information than is necessary (i.e. only the requested or required information is released and no other information revealed). In addition, since the information is sensitive, care must be taken to ensure that data security and privacy are always guaranteed. Therefore, the queries being run on the database must be checked to verify that the responses returned are consistent with the rules in the security policy in order to control the flow of information. Moreover, downloaded information needs to be tracked to ensure that it is always manipulated in ways that conform to the security policy regardless of where and who made the download request.

1.2 Problem Statement

This research is based on the security problems that arise whenever service compositions occur. Basically, when service compositions occur, a global security policy that satisfies the minimum security requirements of each of the participating services must be formed. This typically requires mapping the role of the user behind the service call to an equivalent role within the target domain. All program executions resulting in the composed service must occur in ways that abide by the underlying security policies of each of the participating services.

The situation is further complicated by the fact that security policies can change during service composition. In addition, this policy change is because the composite service is made up of different services from different security domains with different security policies. Therefore, these services have to agree on the security policy to follow and this can change some security policies of some services because their dominant security policies can be overlooked. Therefore, in order to enforce secure information flow control during service compositions we need to consider how to prevent illegal access and flow of information between the participating

services. This becomes difficult when each service has a unique access and information flow control policy that the service complies to. Hence, a global security (access and information flow control) policy has to be defined for all the participating services to satisfy.

1.3 Contributions of the Thesis

In as much as research in access control and information flow control has gained momentum over the years, few works make use of both security techniques in the same context especially in web service environments. Also, there is little work that looks at information flow control policies and access control in mobile web service environments. The thesis presents a novel approach in this regard. This work exposes the problem of unlawful access and propagation of information during service compositions, brought up by participating services from different security domains having to agree on a global security policy to conform to during composition. Similarly, based on this, the work's novelty addresses location and context dependencies in authentication and authorizations for information flow control in mobile services or resource constrained environments. The principal contribution of this thesis is the design and implementation of an access control scheme that takes into account location and context attributes for secure authentication and authorization to resources. Access to resources is subject to users and/or services' location and context being referenced as input for access control decisions. In addition we use the concept of program dependence graphs to enforce legitimate information propagation between services so that all information shared between services is permissible and verified. This is attainable by making use of path conditions which needs to be satisfied first for the flow of information to take place. Consequently, we propose a solution that offers fast authentication and good performance taking into account bandwidth limitations in the mobile web service environment.

1.4 Organization of the Thesis

The remainder of the work is organized as follows (Figure 1.1); Chapter 2 details our research interest and defines the problem by providing background information and a conceptual examination of related work, specifically, an introduction to Information Flow Control (IFC) and Access Control schemes. Both IFC and Access Control are the major motivation behind the work presented herein as no work has addressed both techniques in the same context in mobile web

service environments. The threat model and its implementation are found in Chapter 3. Chapter 4 presents the formal definition of the approach taken in designing the research. It contains the design chosen, and the system and security prototype design aspect of the work. In Chapter 5 we present the implementation of the work. Analyses of results are discussed in Chapter 6. Finally, Chapter 7 concludes the work and offers a direction for future work.

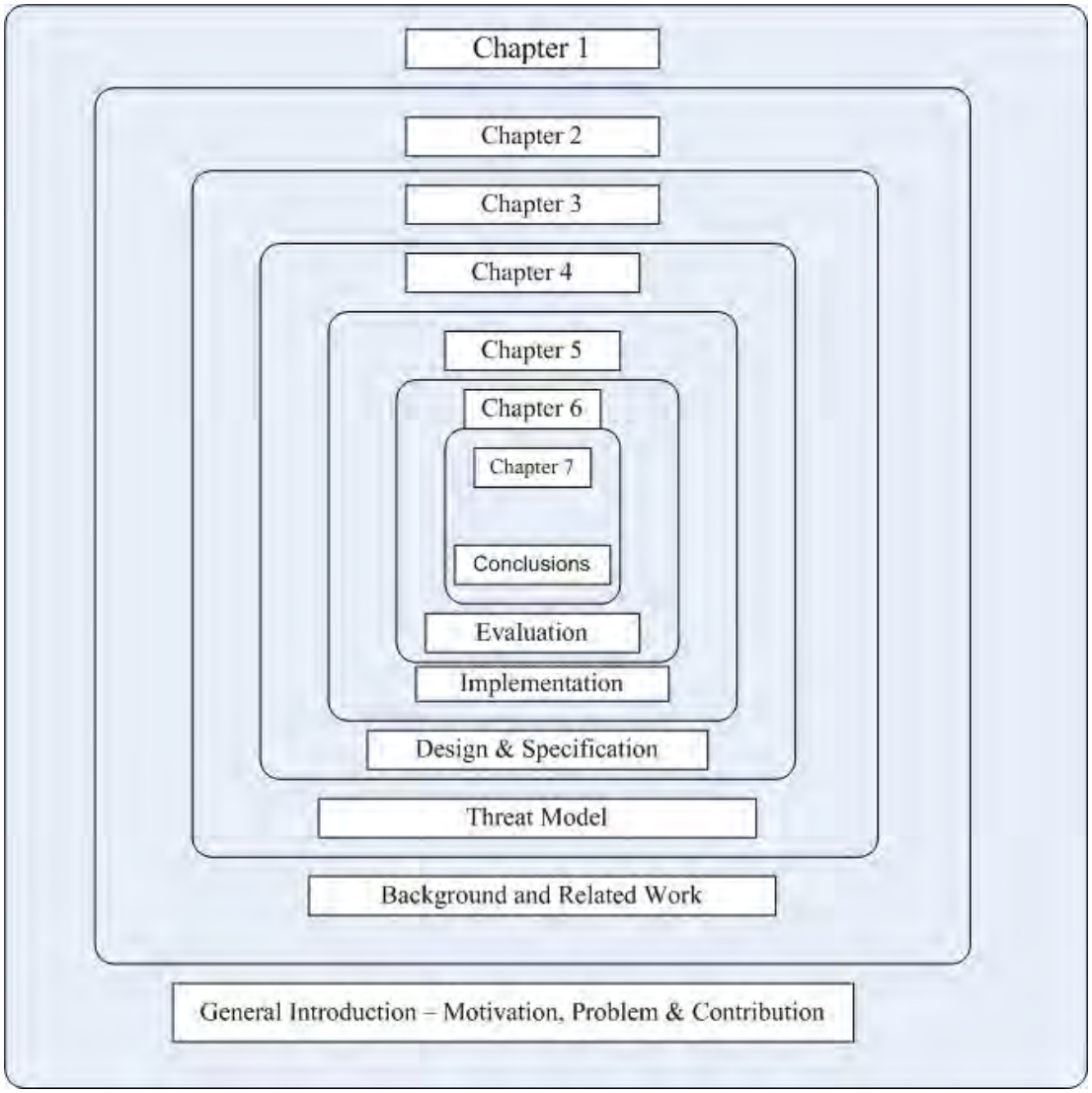

Figure 1-1: Thesis Structure

Chapter 2
Background and Related Work

As a preamble, research on Information Flow Control (IFC) has over the years received abundant attention due to its significant role in securing web services or Service Oriented Architecture (SOA) environments. Furthermore, in recent times, the high mobility of users and services in the emerging mobile applications (mobile web services) has called for stringent security mechanisms to be put in place to guarantee consumers or users information is secure both in storage and transaction. In principle, such security mechanisms used include Information Flow Control, which denotes the security of a software system with respect to a security specification, particularly enforcing information flow policies that guarantee desirable transfer of information in a given process from a variable *a* to a variable *b*. Also this largely borders on three security properties, confidentiality, integrity and availability. Access control on the other hand has also been primarily used to address these three security properties. The confidentiality aspect in access control is addressed by making use of authentication and authorization while IFC has taken care of the integrity aspect. Over the years, there has been a largely renewed interest in addressing these two security properties in the same context. This chapter presents some background or previous work done on different kinds of security techniques that relate to this thesis, mainly focusing on information flow control and access control. The need to monitor how information propagates between services and/or users is of paramount importance in ensuring a smooth secure flow of information. The following section gives a background of mobile web services and their compositions.

2.1 Mobile Web Services

A Web service as defined by [104] is an application that can be accessible by being automatically discovered and invoked by other applications and humans. Generally, these services are applications that are autonomous, self-describing and integrated that can be published, located and invoked as and when required on the Web. However, when these services are accessible from a handheld mobile device which is connected to a mobile network or other wireless network through the Internet, they are referred to as Mobile web services [31]. However, the term Mobile Web Services is not clearly defined. It is used in different domains with different meanings. Hence, in this thesis Mobile Web Services define the area where web services are applications that are accessible from a mobile device connected to a wireless network or mobile network. In short, a Mobile Web Service is a Web service that is deployed on a mobile device and connected through a wireless network.

Since mobile web services differ from a more traditional web service application, as a result, the constraints imposed by the limitations of the device, in terms of limited user interface, low processing power and often low bandwidth will influence access and information flow control in this environment. [84] considers that the influence of the constraints is much more visible when the mobile web services collaborate to create or form a high level business process. This collaboration is formally referred to as composition. The following section gives some background on mobile web service compositions.

2.1.1 Web Service Compositions

Web Services Composition as defined above is a method to connect together different available web services to create a high-level business process. It involves assembling atomic or candidate web services to provide functionalities that are not available at design times. As a result, a new functionality can be developed through reusing of components that are already available, but unable to accomplish a task on their own.

Different works have evaluated web service compositions by looking at the different problems affecting their compositions. [73] addresses in detail the problem of incorporating web service discovery and composition; however they only consider simple workflows where web services

have one input and one output parameter. In their work, the web service composition blueprint is regulated to a sequence of limited web services corresponding to a linear workflow of web services. The proposed solution recovers a sequence of causal links between web services, as a result, a linear and total order of services making up the required composed service. Proposing a similar solution to [73] above, [105] uses a composition path that is made up of a series of operators that calculate data, and connectors that provide data passage between the operators. A shortest path algorithm on the graph of the operator space is used to search for possible operators to create a sequence. Even so, only two kinds of services (connector and operator) with one input and output parameter are considered, meaning only the simplest case of service composition is presented. Presenting a contrary view, [84] proposes workflows with services having more than one input and output parameters.

In [61] a formalism and modeling tool called interface automata is proposed to represent web services and perform compositions. Their work presents atomic services being stored as a graph where each node resembles input and output parameters and edges represent web services. Each web service contains a description of its inputs, outputs, and dependencies of other web services. Web service descriptions and the graph are used to discover composition results that satisfy a service request. Given that several compositions can be found, the work offers no optimization mechanism for selecting the desired requested composition. However, out of all the matching several requested composition, nonetheless, the most suitable composition will still be selected.

An interesting path is taken by [42] by introducing a composer to perform web service composition. The composer requires the end user to select web services for each activity in the composition and to create how the path linking the web services is specified. After selecting a web service, the web services that can create an output that could be used as an input of the selected service are listed, after filtering based on profile descriptions. Ideally, the user can manually choose the service that he deems proper for a particular activity. After all the services are selected, the system generates a composite process. The composition is effected by calling each service individually, and passing the results between services according to the flow specifications.

Consequently, during composition, data exchanged and shared between the web services needs to be protected from malicious users who could be eavesdropping or unauthorized to access the

service. The following section gives the background of the techniques used to safeguard against the unwanted information propagation and access in web service compositions.

2.2 Information Flow Control

Information Flow Control (IFC) [28] is a technique that asserts the security of a software system with respect to a security specification. It is a technique that asserts that data or information disseminates in a proper and secure way in a system. In general, information flow policies are used as mechanisms to enforce information flow control. These mechanisms which include Run-time mechanisms, tag data with information flow labels, have been employed at the operating system level and at the programming language level. Static program analyses have also been established or initiated that ensure information flows within programs do conform to policies set. While information flow has been largely or intensively researched in the last decade, [28] observes that the methods for checking security policies only have an effect on a very restricted part of a program analysis technology, commonly type systems. When compared to standard security practices, IFC gives an explanation for program semantics. [9] gives two principal ways or sources of information flow in programs; explicit flows which are simply direct flows of information resulting from assignment operations and Input/Output (I/O) statements and implicit flows; which may arise when the control flow is affected by secret values or signal information through the control structure of a program.

In this section we discuss and analyze various mechanisms used to address the issue of security in information flow control, taking into account the important aspect of information flow security, which is the preservation of integrity of information.

2.2.1 Noninterference

As noted by Daniel Hedin and Andrei Sabelfeld [36], the fundamental basic logical concept of secure information flow is noninterference. Several authors [28], [9], [91] define noninterference as an idea that confidential data may not interfere with (or affect) public data. Hammer [28] contends that the most noteworthy illustration of a security policy that put in force noninterference involves that the secret input of a program may not flow to public output; in

effect, public output may not even be influenced or subjected from secret input. Noninterference affirms confidentiality in the Confidentiality, Integrity and Availability (CIA) – triad [28]. This is appropriate in terms of formalization or validation; however, it can be regarded as too limiting or restrictive because, in certain cases programs need to disclose confidential information in order to deliver their intended purpose. [9], [28] give an example of a program that needs to write the Boolean outcome of an *if* statement that contains high level confidential or secret data. Even though the return output can be false, it would have partially leaked that the argument (s) in the statement are sensitive or not.

It is however, worth noting that a lot of research in this field has been focused towards modifying the definition of noninterference to support more realistic usage scenarios or even using different designs to secure information flow. Smith [44] proposes a definition of probabilistic noninterference in which it is required that the initial values of high security variables cannot affect the joint probability distribution of the possible final values of any low security variables. Rossi et al. [91] proposes a concept of noninterference for multi-level service compositions and they argue that SOAs are increasingly relying on complex distributed systems that share information with multiple levels of security. In these systems, [91] argue that information with mixed security levels is processed and directed to particular clients. This makes the system's security exposed to adversary participants because secret or confidential information is passed to participants whose clearance level to access this type of information is undetermined. Hence, they define noninterference for service compositions presenting stipulations that public synchronizations are not affected or changed as confidential communications are varied or, in a more general way that the way the composition behaves in its low level (public non sensitive level) is independent from the behavior of its high level (sensitive-confidential) components. As a result, clients are guaranteed that the passage of the data over the internet to a web server remains confidential or it cannot be intercepted and understood by unwanted listeners [91].

Taking the contrary view, Laud [75] defines secure information flow in terms of computational indistinguishability rather than noninterference. He argues that the general objective is to prove that an adversary or attacker cannot learn anything about the confidential inputs of a program by observing its public output. Since real adversaries are resource bounded, computational

indistinguishability assumes an adversary is working in probabilistic polynomial time, where polynomial means polynomial in a suitable security parameter such as the encryption key length [20]. Other works [40], [44], [46] propose models of classifying general features of services that may influence the security of information flow. They also define transformation factors which measure how possible the output of a service can be used to obtain the input and/or the local data used in the computation of the service. They also discuss the honest hypothesis of unreliability or untrustworthiness regarding service composers. They argue that service composers may not always be fully trusted because web services may be provided by different domains under different security administrators. Thus, individual composers may not meet certain security requirements (like authentication and authorization) and may not fully be trusted by all the service providers. Hence the introduction of intended procedures for the service composer to relate with security authorities for distant policy examination and negotiation, resulting in very resolute and flexible policy validation processes. Mantel and Sabelfeld [48] explore a timing-sensitive security property for multithreaded programs, later drawn-out to the distributed environment. Sabelfeld and Sands [10] considers probabilistic bisimulation based designs of confidentiality for multi-threaded programs, focusing on formulations for timing- and probability-sensitive confidentiality. They stem relationships between scheduler specific, scheduler independent and strong confidentiality. Work by Roscoe [17] also investigates confidentiality properties in a process-calculus setting. A concept of low-view determinism is presented, which demands that abstracted publicly observable outcomes are deterministic and, accordingly, independent of secret inputs. The above works have explored different platforms in addressing secure information flow, particularly in multithreaded environments other than noninterference. The following section, details another IFC approach, declassification in addressing the secure flow of information.

2.2.2 Declassification

Declassification [5] is the lowering of a security organization or categorization of selected information. And for many applications, a complete separation between secret and public information/data is too narrowing or constricting. There are four different latitudes of declassification as identified by Sabelfeld et al [10]. They include what is declassified, who is

able to declassify, where the declassification occurs and when the declassification takes place. Hedin and Sabelfeld [36] define the dimensions of declassification as follows;

- ***What*:** asserts the importance of being able to specify what information is being declassified, e.g. the last 4 digits of an identity number. Policies for fractional release must guarantee an upper limit on what information is released.
- ***Who:*** asserts the importance of who controls the release of information? This refer to information integrity – if the attacker is able to control what information is made public he might be able to mount a laundry attack, i.e., accidental disclosures concealed by the systems declassification policy.
- ***Where*:** there are two forms of release locality identified by the researchers. Relating to the what and when dimensions, the where dimension is the most intermediate interpretation of where in terms of code locality. The other form is level locality, describing where information may flow relative to the security of the system.
- ***When*** asserts the temporal aspect of declassification pertaining to when the information is released or leaked. Sabelfeld et al. [36] identify three classes of temporal release classifications, Time-complexity based, Probabilistic and Relative. The two former are related. Time-complexity based states that information will not be released until, at the earliest, after a certain time; typically as an asymptotic notion relative to the size of the secret. With probabilistic considerations one can talk about the possibility of a leak being very small. The class of relative temporal policies is on the other hand related to program correctness. It controls when declassification can occur relative to other (possibly abstract) events in the system. For example: "downgrading of a software key may occur after confirmation of payment has been received".

2.3 Approaches for Information Flow analysis

Several authors have contended that, once secure information is released, it may be disclosed maliciously or even accidentally through a bug in a program. Therefore, in order to safeguard that such security policy is followed, it is important to look at how information propagates through the program. Given the complexity of modern computing systems, it is not possible to manually examine the flow of information. Language-based methods [9] are generally used by design to efficiently analyze the flow of information within a program so that end-to-end security

policies may be enforced. Hammer and Sabelfeld et al. [28], [9] explore two prominent information flow analysis techniques or approaches; dynamic analysis and static analysis. The next section covers these approaches in detail.

2.3.1 Information Flow Analysis

According to Hammer [28], information flow control can be achieved online when a program executes or offline (often at compile time). The following techniques present the two ways to achieve IFC during program execution and/or at compile time.

2.3.1.1 Dynamic Analysis

Dynamic analysis attempts to analyze information flow within a program while it is executing. One approach includes Bell and LaPadula's [30] mandatory access control. In this approach, each data element is labeled with a security level. Information flow is controlled dynamically by increasing the ordinary computations within a running program to instantly compute the label that will control the future propagation of the data. In general, performance monitoring approaches are not well suitable for analyzing information flow. Run-time analysis mechanisms only have information available about how a program is behaving in a single execution. In addition [40], such mechanisms do not have sufficient information to predict future steps the system might take. On the contrary, the analysis of information flow in mobile code programs requires a dynamic approach. Work by Focardi and Rossi [79] in particular explore the problem of enforcing security properties for programs, such as mobile agents, whose environment will change at run-time. Their approach is to ensure that every state reachable in the system satisfies a non-interference property.

2.3.1.2 Static Analysis

Static analysis on the other hand attempts to analyze the information prior to the execution of a program, often at compile time. Hammer [28], [29] ascertains that if the program can be confirmed, no program execution can reveal illegal information flow, hence avoiding the overhead of runtime checks. However, Sabelfeld et al. [11] implores that precise static IFC

analysis is undecidable, so all static analyses need to be conservative. Works by Denning et al. [33], [35] and later Volpano [45] fall under this approach. There are two major categories of static analysis approach as observed by different authors [5], [9], [75]; Approaches that use type systems and those that use programming language semantics [35]. Furthermore, according to Hammer, type systems do not exploit the whole range of contemporary program analysis; hence, they suffer from restrictive languages and a high annotation burden.

In order to remedy the above mentioned short comings of type systems (contemporary program analysis and high annotation burden), Hammer et al [28, 29], presents a new approach based on program slicing and the system dependence graph. In general, it extends the algorithm for program slicing to allow for precise information flow control and provide a means to downgrade secret information, if necessary. The works [28], [29] explore the idea that path conditions provide further understanding into how one statement influences another. Therefore, they may lead to conditions for illegal information flow, or they may provide confirmation that a presumed flow is impossible. The contrary view taken by Hammer [28] is a novel approach for information flow control which uses system/program dependence graphs (PDGs). Hammer argues that the flow-sensitivity, context-sensitivity, and object-sensitivity of his slicing program method extends naturally to information flow control and thus excels over the leading approach, which is security type systems.

Despite the advances reviewed above, security type systems have still not been used much because the notion of noninterference is difficult to attain in practice for various reasons, like covert channels and declassification whereby confidential information may be partially leaked in data aggregation. For example [48], consider for instance a program that computes average salaries, even though each individual salary is private, we might want to publish the average salary. Therefore, other research avenues have been explored over the years in trying to bridge the gap of securing type systems to enable them to be used in securing service compositions. Consequently, recent research has considered various channels in securing the flow of information in service compositions. One way to introduce flexibility [54] is to consider type systems for information flow that take access control into account. The next section will cover or look at the different access control models or schemes explored to secure information flow.

2.4 Access Control Schemes for Secure Information Flow

As history and research will discover through their usage, standard access control mechanisms as stand-alone mechanisms, only control the release of information but not its propagation once access has been granted. This holds true for Discretionary Access Control (DAC), because, although it is effective for specifying security requirements and is also easier to implement in practice, its inability to control information flow implies that it is not well suited to the context of web-based shared applications where control in some form is required [71]. DAC models also suffer from being vulnerable to Trojan Horse attacks. Trojan Horse attacks are driven by deceiving valid users into accepting to run code that then allows a malicious user to get access to information on the system.

While on the other hand Mandatory Access Control (MAC) [60], [71] counters these threats by governing access centrally. Hence, an ordinary user cannot change access rights a user has with respect to a file, and once logged on the system, the rights he/she has are always assigned to all the files he/she creates. This formula [62] allows the system to use the concept of information flow control to provide additional security. Information flow control allows the access control system to monitor the ways and types of information that are propagated from one user to another. A security system that implements information flow control usually categorizes users into security classes as noted by Denning [9], and all the valid channels along which information can flow between the classes are regulated by a central authority. MAC models are typically designed using the concept of information flow control.

Previous work by Banerjee et al. [7] studied access control mechanisms in relation to stack inspection, and established a connection between authorization of information access and the subsequent flow of the information. They concluded that the noninterference property guarantees that the access control mechanism is serving correctly to enforce flow policy, in a way that once access has been granted, there is no subsequent leak of secret of the information. More work by Banerjee et al. slightly similar but different to the above, encompasses how dynamic access control can allow flexible program interfaces where a data channel can be used for more than one purpose, while ensuring confidentiality. They consider the access control mechanism of Java [8] as defined by J. Gough which aims to protect trusted system code. The principals that are granted permissions in an access policy are programs rather than, say, processes or users as in an

operating system security. They also deliberate on programs that use access control to enforce information flow policy expressed by labeling of input and output channels with levels in a lattice as defined by Bell & L. LaPadula [48] and Denning [9]. Unlike stack inspection, the conclusion is that the mechanism itself introduces a new channel of information flow, but one that can be controlled using the same sort of type-and-effect analysis that was previously developed for stack inspection. As with previous work [54], the analysis validates code with respect to a given policy. And in this respect, policy defines both trust and confidentiality levels.

Total trust in access control systems has in countless scenarios always been a potential threat to information security brought about by system administrators. The evolution of traditional access control has seen this gap being marginally closed by more flexible and dominant systems like Role-Based Access Control (RBAC) [81] and Flexible Authorization Framework (FAF). Other access control models like Cryptographic Access Control (CAC) [79] have been used to enforce access control because of their consideration or use of data encryption, hence, unauthorized access is more difficult because the data remains encrypted no matter where it is located, and only a valid key can be used to decrypt it. An example is the enforcement of hierarchical encryption models to enhance access control mechanisms. One particular work to cryptographic solutions to a problem of access control in a hierarchy is a solution presented by Akl and Taylor [58]. The authors introduce a scheme based on cryptography for access control in a system where hierarchy is represented by a partially ordered set (poset). Its application is direct, requiring users highly placed in the hierarchy to store or keep a large number of cryptographic keys. A time-versus-storage trade- off is then defined for addressing this key management problem. The scheme enables a member of an organization at some level of the hierarchy to derive from his own cryptographic key the keys of members below him in the hierarchy, and therefore to have access to information encrypted under those keys. This solution is interesting because, the protection it offers against illegal disclosure depends neither on the physical security of the storage medium where the information is kept nor on the trustworthiness of the people managing it. It also accommodates not only files that are stored in a central computer memory, but also to messages broadcast on a communication network. Anyone with the proper receiving tools can intercept the message but has access to the information it contains only if in ownership of the right key. Basically, the need of access control in a hierarchy arises in several different contexts, one of which is managing the information flow for an organization where, the users are divided

into different security classes depending on their access needs. Several cryptographic solutions [15], [80] have been recommended to address this problem – and the solutions are based on generating cryptographic keys for each class such that, the key for a lower level security class depends on the key for the security class that is higher up in the hierarchy. A different approach is taken by Sylvia Osborn and Yuxia Guo [58] focuses on modeling users in role-based access control to enforce access control in role hierarchy encryption models and simultaneously reduce redundant user-role assignments.

Therefore, standard methods of enforcing access control in web-based applications include those supported by Cryptographic Key Management (CKM). One standard characteristic presented by the above scheme [87] is that as a Cryptographic Access Control (CAC) scheme, it has to rely on or be supported by Cryptographic Key Management (CKM) schemes to execute proficiently. And unlike authentication schemes that rely on system specific security policies, CAC schemes do not rely on the physical security on which the data is stored as observed by Akl et al [87]. Instead they rely on KM algorithms that place a heavy processing cost on the system. And this has resulted in their unpopularity in web applications.

Consequently, to alleviate this problem one has to enforce cryptographically controlled access to stored data by encrypting it with a single secret key that is then dispersed to the users requiring access [14], [15]. Data security is then achieved by replacing the group key and re-encrypting the affected data whenever group membership changes. However, key management (KM) is expensive when changes in group membership occur frequently and involve large amounts of data. Kayem et al [15], [16] presents a framework based on the autonomic computing paradigm that allows a KM scheme to continually monitor the rate at which changes in group membership happens and generate keys as well as encrypted replicas to guard against future changes to address this problem. Therefore, since the keys and encrypted data are generated by anticipation rather than on demand, the long-term cost of KM is reduced. The framework comes with functionalities that are organized into six components; the sensor, monitor, analyzer, planner, executor and effector, that are interconnected to form a feedback control loop (FBCL). The function of the FBCL is to continuously monitor the arrival rate of rekey requests at the key server and, at regular intervals, calculates an acceptable resource (keys and encrypted replicas) allocation plan to reduce the overall cost of rekeying. The underlying benefit of the scheme is

that each component of it adds to the improvement of a standard CKM scheme's performance without changing its principal characteristics. The proper implementation of this framework guarantees confidentiality and/or integrity in the data being protected or secured. Likewise, service compositions should assure clients or users of their unconditional trustworthiness in guaranteeing confidentiality and/or integrity in their services, which is the underlying property of all security aware systems.

Accordingly, the combined use of Internet, the Web and mobile technologies (e.g. mobile devices, mobile and wireless communication – mobile web services as described in Section 2.1) makes it possible for users to connect to remote resources and services from a wide range of settings. This combination of resources and services found in mobile and wireless communication is commonly known as mobile web services [104]. Even though the focal motivation of such environments and connectivity is to increase the availability of information and services to users, security is also an important requirement. Several applications that benefit from mobility and improved connectivity need to access sensitive information and services that need to be adequately protected. Thus, securing the flow of information and resources in such settings involves not only protecting network communications, but also providing strong authentication and access control, as they are crucial to secure the end-points of computing infrastructures and to make sure that information and services are used according to the organizational policies and legal requirements [18]. The need for relevant access control models in such environments is essential, taking into account the dynamicity and complexity of the environments these applications are deployed in. One such access control model appropriate for these types of settings is location based access control (LBAC). Generally, LBAC [24] models regulate access of a subject to an object, considering only the location of the subject. Denning et al. [32] asserts that Location-based Access Control (LBAC) schemes integrate traditional access control mechanisms (DAC, MAC, RBAC) with access conditions based on the physical position of users and other attributes related to the users' location. Therefore, LBAC takes the location of the user and the time of the request into account in order to decide whether to grant or deny an access request. Other works [25], [27] evaluate LBAC as a double pronged scheme in the way it uses location information for authentication. As observed by [67], the first is to use the location as one element in the authentication process or incorporate location in the security policy. This in turn prompts the user to provide more authentication proofs as a result of location being used as

a multi-factor authentication. The above method yields drawbacks like having untrustworthy users who deliberately do not give or report their location. As a result, the design must consist of mechanisms that inhibit the user from falsifying this information. Secondly, location is used to determine the security policy whereby the location information is external to the policy. This is observed for instance, when a user is connecting to his company's network from home. In this case, he may be required to provide more credentials than when he is connecting from the office. Hence, the use of location information being separate to the policy specified.

A majority of the works [51], [52], [66] evaluate LBAC models by considering both subject and object dynamics. However, emphasis is paid on the fundamental notion that location in these models or schemes describes where the user is accessing information from. The location information is used in numerous types of authorization rules. One type uses location to find the trust domain where the user is accessing information services from. A reasonable policy would deny access to any sensitive information to anyone accessing it from such areas. Furthermore, location can also be used to develop the emergency level of access, for instance, a policy can allow read access to all images of all patients for any user assigned to the role physician and accessing the information from an emergency room.

Consequently, other works monitor access to resources by evaluating attributes associated with the subject, object, requested operations, and, in some cases, environment conditions against policy, rules, or relationships that describe the allowable operations for a given set of attributes [103]. ABAC in general is a logical control model that is distinguishable because it controls access to objects by evaluating rules against the attributes of entities (subject and object), operations, and the environment relevant to a request [41].ABAC enables definite access control, which allows for a higher number of discrete inputs into an access control decision, providing a bigger set of possible combinations of those variables to reflect a larger and more definitive set of possible rules to express policies.

In [50], the authors note that the access control policies enabled that can be applied in ABAC are restricted only by the computational language and the abundant available attributes. Therefore, this adaptability allows for the highest range of subjects to access the highest range of objects without specifying individual relationships between each subject and each object. For example, in a healthcare environment, a subject can be assigned a set of subject attributes when employed

(e.g., Alice Doe is a *Nurse Practitioner* in the *Cardiology Department*). Also, an object can be assigned its object attributes when created (e.g., a folder with *Medical Records* of *Heart Patients)*. Furthermore, objects may be given their attributes either directly from the creator or as a result of automated scanning tools. The administrator or owner of an object creates an access control rule using attributes of subjects and objects to manage the set of permissible abilities (e.g., all *Nurse Practitioners* in the *Cardiology Department* can *View* the *Medical Records* of *Heart Patients).* As a result, under ABAC, access decisions can change between requests by simply changing attribute values, without having to change the subject/object relationships defining basic rule sets. Consequently, this makes up for the provision of a more dynamic access control management capability and control the need for long-term maintenance requirements of object protections. Therefore, in dynamically changing environments like mobile web services, ABAC becomes an ideal access control model to deploy.

2.5 Discussion

Advances in specifying analyses have been growing ever since Denning and Denning's [35] early work on static certification of secure information flow. Others like Sabelfeld and Myers [9] provide extensive surveys of the literature on language-based information flow control. Majority of the works are proposed in the style of a security type system that is shown to enforce the prevailing basic semantic notion of secure information flow, noninterference. Despite the advances, security type systems have not seen much usage popularity because noninterference is difficult to accomplish in practice for a number of reasons not limited to declassification and covert channels. Therefore, other ways to introduce flexibility is the consideration of security type systems for information flow that takes access control into account. Protection models which integrate access control and information flow control in the same framework have not been thoroughly explored. The use of CAC models to secure information flow in service compositions can be beneficial in achieving the underlying principle of confidentiality and/or integrity. In light of the above, our work will focus on enforcing secure information flow control during service compositions. We will consider how to prevent illegal flows of information between participating services by specifying an access control model that combines location and context dependencies for authentication and authorization first. Location dependencies involve the location information of the subject trying to access the object specified. RBAC on the other

hand is a popular approach to enforcing access control in enterprises and so it follows that one might extend this model to handle service compositions. The second step will be to use the concept of dependence graphs to enforce information flow control in cases of service compositions involving services from potentially different domains with possibly different access control policies. Works by Hammer [28], [29] and others [10], [46], [43], [44] will be good references for this work or thesis. The goal of this concept (dependence graphs) will be monitoring requests for access to data and making decisions as to how to adjust the security policy to cope with the observed change without violating the minimum security requirements of the participating services.

Chapter 3
The Threat Model

As earlier stated in Chapter 1, a composite service is made up of different services with different security policies and most particular from different security domains. These services are in their own accord susceptible to specific threats and vulnerabilities from their specific domains. However, once the services compose the threats make a common factor to the composite service. These can be threats related to access control in one domain and threats related to information flow control in another domain. Therefore these threats can impact the access control mechanism of the service and also the way the information is propagated once access has been granted. This can pose a major security risk of the mobile composed service. The following section presents a model of identifying and circumventing these threats when mobile web services compose taking into account constraints like location and context dependencies in their authentication for secure information flow.

3.1 Threat model

This chapter structures the threat modeling process for this work by using the laid down threat modeling principles presented by the Open Web Application Security Project (OWASP) [74]. It details the identification and evaluation of threats and vulnerabilities of the prototype defined in Chapter 4 of the mobile web service application. Since there are two security techniques (access control and IFC) integrated by the prototype, the threat model covered here seeks to primarily address these two techniques' threats and vulnerabilities by looking closer at the authentication, authorization and integrity mechanisms used by the model in a mobile web service environment. Therefore, we scale down our model to address the above mentioned security properties.

We assume a common threat model with a highly motivated adversary who can compromise our system by using shoulder surfing on a legitimate user trying to access the system. We scale down this model to reflect the contingency factors in our application scenario. In our deployment scenario, the attacker shoulder surfs a user's session, the goal being to steal the user's login

credentials in order to attack the system and steal or gain access to the medical records stored in the EMR database. Figure 3.1 illustrates the attack tree which is derived from a list of stakeholders, deployment architecture, and communication protocols.

Stakeholders: we observe that patients are the main stakeholders in this system. Their interest is to maintain individual privacy and also the privacy of their medical data. The second group is made up of health care providers or nurses, whose interest is system availability, ease of use, and maintaining the privacy of patients' medical data. The third group is made up of physicians or doctors, whose interest covers the two groups' interest as well as managing the system. On a high and internal level we also consider a fourth group which is made up of the participating services, whose interest is accessing other services' output to form the composite service.

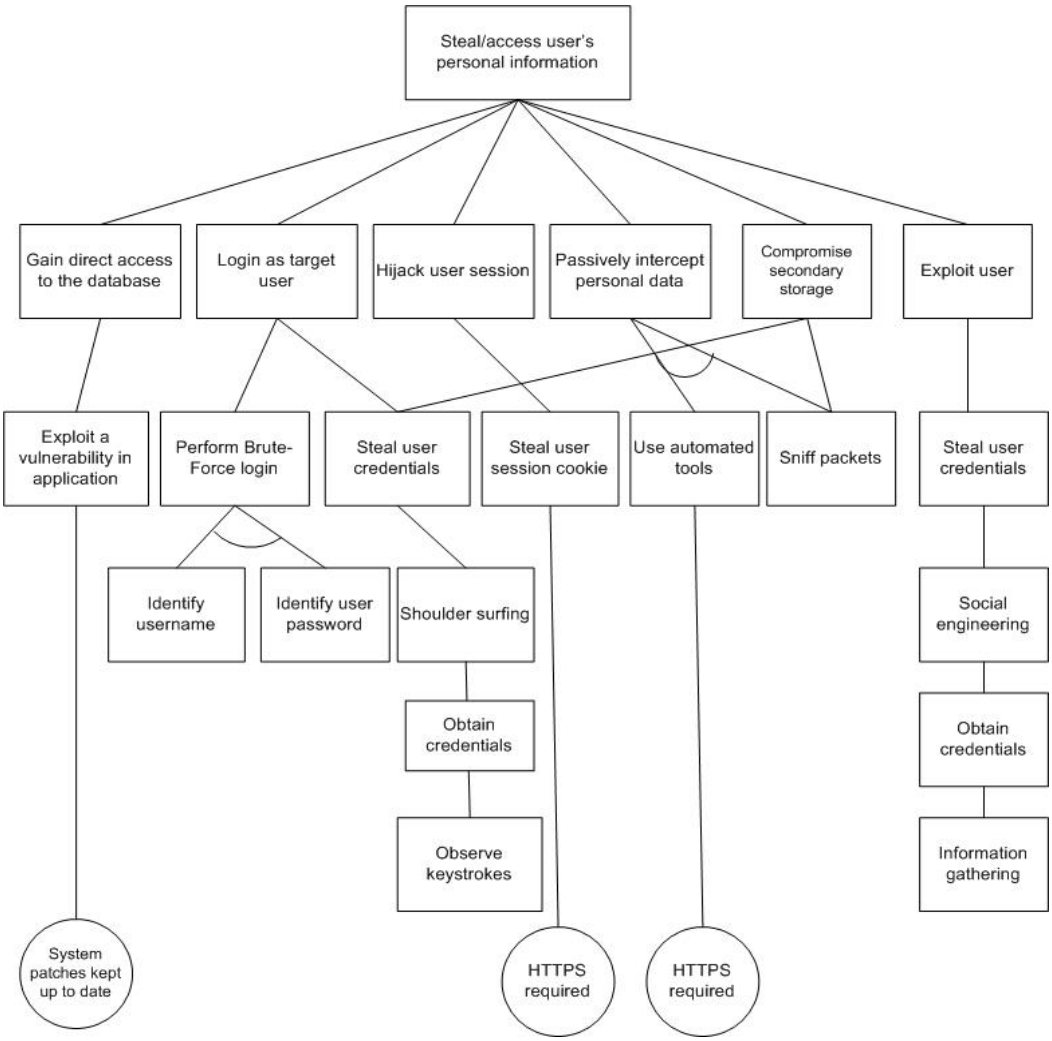

Figure 3.1: an attack tree illustrating a threat in which an attacker steals a user's personal information

To put our threat model into perspective, we use a case of a patient trying to access the system from a low security domain using his/her mobile device. We assume or consider that a low security domain is a public network like an unsecured Wi-Fi hotspot or a cellular network. Furthermore, we assume the patient's location to be a public clinic. In this case we have an adversary looking on the patient's shoulder (shoulder surfing) or a social engineering attack whereby the adversary is pre-interacting with the patient to gather intelligence (user information) trying to steal the patient's login credentials to use and get access or steal the patients' medical records. In a public domain and location the system is highly vulnerable to both logical and physical attacks. While at a high security or private domain and location, the system enjoys improved logical and physical security. Because of the varying degrees of hostility in our deployment environments and attack advantages, we assume a sound design which takes into account individual components of the system to help identify the threats related to that component. Therefore, in subsections 3.1.1 and 3.1.2 we give some assumptions for our threat model to hold in the deployment scenario.

3.1.1 Public or low level security threats

In the public or low level security domain, the system or service application is more vulnerable to different kinds of attacks because there is nothing that regulates its usage as a first line of defense. The attacker can deploy robust computational resources since the network is unrestricted. Below are some assumptions we make regarding threats from unrestricted or public networks/domains and locations.

- It is easy for an attacker to perform shoulder surfing
- It is easy and cheap for an attacker to deploy automated tools to sniff packets from mobile devices
- We assume that the adversary has a high probability of monitoring all communications from network initialization to shutdown given the small number of mobile devices involved in that location and the predictability of network operation.
- We also assume that the adversary can mount active attacks at any time during the operation of the network.

3.1.2 Private or secure level domain threats

Patients or users in a high level security domain face a slightly different challenge. Their location presents new threats as well as defense opportunities.

- In a highly secured domain, the adversary has limited computational resources unlike in the public or low level security domain because the domain presents a first line of defense as it is secured assuming the attacker is not an insider who has full access to computational resources.
- We also assume that the user is susceptible to shoulder surfing like in the low level security domain
- Because of restricted network availability, the attacker has a very low probability of monitoring the network during initialization, but he still has opportunities of monitoring as well as injecting new traffic in the network during operation.

3.2 Capturing threats and threat model discussion

From a list of stakeholders and system configuration, we present our threat model in the form of an attack tree (Figure 3.1) as described in Section 3.1 above. We seek to uncover how our prototype defined in Chapter 3 handles the access control and IFC requirements exploited by the threats identified in this section. The threat model information, external dependencies, entry points, exit points, trust boundaries, threats and vulnerabilities affecting our prototype are all captured in their individual summarized format using the framework presented by [74]. The threat modeling presented here is a software centric model which focuses on identifying and addressing vulnerabilities by looking for types of attacks against each element of the model. All the perceived threats that (may) affect the prototype and their mitigations are covered in the sections that follow.

3.2.1 Threat Model Information

This section presents the information relating to the service application. Such information includes the service application version, application or document owner, service application

description and participants. Although this information may not be relevant to the attacker, we present this information for documentation and presentation purposes.

Threat Model Information	
Application Version:	1.0
Description:	The Electronic Health Record Medical (EMR) service application will be able to provide clients with on the go medical records on their mobile devices. This is the first implementation of the service application; therefore, functionality will be limited. There will be three users of the application: 1. Patients 2. Nurses 3. Physicians Patients will be able to log in and view their medical records; nurses can view and add users and users' medical records to the system or database. Physicians will be able to log in, edit users' medical records, and add users as well as manage the service application.
Doc. Owner:	Lwazi Maziya

3.2.2 External Dependencies

These are items external to the code of the application that may pose a threat to the application. They may also be used by an adversary as third party points of attack to the service application.

External Dependencies	
ID	**Description**
1	The Electronic Medical Record application will run on a Windows server running Apache. This server will be hardened (security enhanced) as per the provider's server hardening

	standard. This will include deploying the application of the latest operating system and application security patches.
2	The database server will be MySQL and it will run on a Windows server. This server will be hardened as per the provider's server hardening standard. This will include the application of the latest operating system and application security patches.
3	The connection between the Web Server and the database server will be over a private network.

3.2.3 Entry Points

Entry points outline the interfaces through which potential adversaries or attackers can interact with the application or even supply it with data. They are therefore basically entry points for attack of the service application. Entry points can include the front-end of the service application like communication ports or internal entry points (which are points that support internal communication with other components of the application) exposed by subcomponents of the application across its layers. Therefore, it is important to know where these entry points are and what type of input data they receive should an attacker plots an attack by evading the front end of the application and directly interacts or attacks these internal entry points. The table below covers the EMR service application entry points.

Entry Points			
ID	**Name**	**Description**	**Trust Levels**
1	HTTPS Port	The service app will only be accessible via TLS. All pages within the service app are layered on this entry point.	(1) Anonymous Mobile Web User with Valid Login Credentials (2) Anonymous Mobile

			Web User with Invalid Login Credentials (3) User with Valid Login Credentials (4) User with Invalid Login Credentials (5) System Administrator
1.1	EMR Main Page	The splash page for the medical health records application is the entry point for all users.	(1) Anonymous Mobile Web User with Valid Login Credentials (2) Anonymous Web User with Invalid Login Credentials (3) User with Valid Login Credentials (4) User with Invalid Login Credentials (5)System Administrator
1.2	Login Page	Patients, nurses and physicians must log in to the service app before they can carry out any of the use cases.	(1) Anonymous Mobile Web User with Valid Login Credentials (2) Anonymous Web User with Invalid Login Credentials (3) User with Valid Login Credentials (4) User with Invalid Login Credentials (5) System Administrator
1.2.1	Login Function	The login function accepts user supplied credentials and compares them with those in	(3) User with Valid Login Credentials (4) User with Invalid

		the database.	Login Credentials (5) System Administrator
1.3	Search Entry Page	The page used to enter a search query.	(3) User with Valid Login Credentials (5) System Administrator

3.2.4 Trust Levels

These are boundaries that indicate where trust levels change. They help focus analysis on areas of concern. They generally represent the access rights that the service application will award external entities. They are cross referenced with the entry points and assets. This allows defining the access rights or privileges required at each entry point, and those required to interact with each asset. They are documented with a unique ID, descriptive Name and a description of the trust level detailing the external entity who has been granted the trust level.

Trust Levels			
ID	**Name**	**Description**	
1	Anonymous Mobile Web User with Valid Login Credentials	A user who has connected to the EMR service app and has provided valid credentials.	
2	Anonymous Mobile Web User with Invalid Login Credentials	A user who has connected to the EMR service app but has not provided valid credentials	
3	User with Valid Login Credentials	A user who has connected to the EMR service app and has logged in using valid login credentials.	
4	User with Invalid Login Credentials	A user who has connected to the EMR service app and is attempting to log in using invalid login credentials.	

5	System Admin	The system admin can create users on the EMR service app and view their personal information.	
6	Database Server Administrator	The database server administrator has read and writes access to the database that is used by the EMR service app.	
7	Website Administrator	The Website administrator can configure the EMR service app.	
8	Mobile Web Server User Process	This is the process per user that the web server executes code as and uses to authenticate itself against the database server as.	
9	Database Read User	The database the user uses to access the database for read access.	
10	Database Read/Write User	The database the user uses to access the database for read and write access.	

3.2.5 Assets

These are the areas of interest to the adversary or attacker. They are essentially threat targets. They can be both physical and abstract. Physical assets are items that can be found or entered into the system or database. For example, a physical asset of this application might be a list of patients and their personal information. An abstract asset is an asset that cannot be entered or found on the system but relates to the system. This might be the reputation of a hospital/clinic. Assets are documented in the threat model as follows: A unique ID is assigned to identify each asset for easy cross reference to threats and vulnerabilities identified, a descriptive Name that clearly identifies the asset, a textual description of what the asset is and why it needs to be protected and Trust levels, which are level of access required to access the entry point.

Assets			
ID	**Name**	**Description**	**Trust Levels**
1	App users and	Assets relating to patients, nurses, and physicians.	

	Admins		
1.1	User Login Details	The login credentials that a patient, nurse or physician will use to log into the EMR service app.	(3) User with Valid Login Credentials (5) System Admin (6) Database Server Administrator (8) Web Server User Process (9) Database Read User (10) Database Read/Write User
1.2	System Admin/Physician Login Details	The login credentials that a Physician will use to log into the EMR service app.	(5) System Admin (6) Database Server Administrator (8) Web Server User Process (9) Database Read User (10) Database Read/Write User
1.3	Personal Data	This is data that entails the patients, nurses or health practitioner, and physicians' name, last name, username, password, medical history, disease, diagnosis, treatment.	(5) System Admin (6) Database Server Administrator (7) Website Administrator (8) Web Server User Process (9) Database Read

			User (10) Database Read/Write User
2	System	Assets relating to the underlying system.	
2.1	Availability of EMR service app	The EMR service app should be available 24 hours a day and can be accessed by all users.	(6) Database Server Administrator (7) Website Administrator
2.2	Ability to Execute Code as a Web Server User	This is the ability to execute source code on the web server as a web server user.	(7) Website Administrator (8) Web Server User Process
2.3	Ability to Execute SQL as a Database Read User	This is the ability to execute SQL select queries on the database, and thus retrieve any information stored within the EMR database.	(6) Database Server Administrator (9) Database Read User (10) Database Read/Write User
2.4	Ability to Execute SQL as a Database Read/Write User	This is the ability to execute SQL. Select, insert, and update queries on the database and thus have read and write access to any information stored within the EMR database.	(6) Database Server Administrator (10) Database Read/Write User
3	Website	Assets relating to the EMR service app.	
3.1	Login Session	This is the login session of a user to the EMR service app. This user could be a patient, a nurse or physician or a system admin.	(3) User with Valid Login Credentials (5) System Admin

3.2	Access to the Database Server	Access to the database server allows you to administer the database, giving you full access to the database users and all data contained within the database.	(6) Database Server Administrator
3.3	Ability to Create Users	The ability to create users would allow an individual to create new users on the system. These are nurses and physicians.	(5) System Admin (7) Website Administrator
3.4	Access to Audit Data	The audit data shows all audit-able events that occurred within the EMR application by patients, nurses and physicians.	(7) Website Administrator

3.3 Service Application Security Mechanisms

The following are the most prevalent known service application security mechanisms identified so far;

- Forms are used for user's authentication.
- Windows authentication is used to authenticate application at the database.
- Roles are used to authenticate access to system logic – the governing behavior of the system.
- Remote access is not defined or given for administration; it is only defined for mobile web users. Only physical logging on to server computer can allow administration.

3.4 Threats

Apart from the threat assumptions made in subsections 3.1.1 and 3.1.2, the service application could be subjected to the following threats;

- Susceptible to brute force attack against the dictionary store to obtain login credentials
- Susceptible to social engineering thus an adversary can steal user credentials

- An adversary obtains encryption keys used to encrypt private and sensitive data (tampering)
- An adversary gets unauthorized access to server resources by a mobile device (information disclosure)
- SQL injection occurs, enabling an attacker to exploit an input validation vulnerability thus taking control of the database (elevation of privilege)
- Client credentials are captured through network eavesdropping between browser and mobile web server (man in the middle)
- An adversary or user gets authenticated and authorized in the wrong context whereby a patient gets physician rights and privileges after authentication.
- An adversary manages to take control of web server thus enabling him unauthorized control of the database (elevation of privilege)
- A service in a lower security domain receives or can read sensitive data from a higher security domain (information disclosure)

The determination or identification of the above threats can be summarized by using the STRIDE [49] categorization technique as shown in Table below:

STRIDE Threat List			
Type	**Threat action**	**Security Control**	
Spoofing	Threat action aimed at illegally accessing and using another user's credentials, such as username and password.	Authentication	
Tampering	Threat action aimed to maliciously change/modify persistent data, such as persistent data in a database, and the alteration of data in transit between two computers over an open network, such as the Internet.	Integrity	
Repudiation	Threat action aimed to perform illegal operations in a system that lacks the ability to trace the prohibited operations.	Non-Repudiation	
Information disclosure	Threat action to read a file that one was not granted access to, or to read data in transit.	Confidentiality	

Denial of service	Threat aimed to deny access to valid users, such as by making a web server temporarily unavailable or unusable.	Availability	
Elevation of privilege	Threat aimed to gain privileged access to resources for gaining unauthorized access to information or to compromise a system.	Authorization	

3.5 Vulnerabilities

These are un-mitigated threats or threats identified with no countermeasures. The service application vulnerabilities therefore are;

- Storage of user passwords – when passwords are stored as clear text with no encryption they become easily visible to attackers.
- Lack of password complexity enforcement – when there is no enforcement of password creation; dictionary attacks easily retrieve sensitive data.
- Weak input validation on server side – when server does not validate the data stored or queried; all executed code is accepted making the system vulnerable to malicious code.
- Covert channels – when all known channels are not secured
- Failure to sanitize data read from a database makes data to be easily interpreted when intercepted thus disclosing sensitive information without problems.

3.6 Countermeasures

The major purpose of a countermeasure (a safeguard that addresses a threat and mitigates risk) identification is to try and regulate if there are some protective measures that can be adopted to prevent the threats identified from being realized. Since these threats have been categorized with STRIDE, it is therefore possible to find the right countermeasures to address them from the service application. The countermeasures adopted for the threats identified for this work are given in a summary below.

Threat Type	**Mitigation Techniques**	**Countermeasure**
Spoofing Identity	1. Appropriate authentication 2. Protect secret data 3. Don't store secrets in the clear	• Authentication tokens and credentials are protected and stored in encrypted format • Passwords are stored with salted hashes • Strong password policies enforced • Trusted server authentication used not SQL authentication
Tampering with data	1. Appropriate authorization 2. Hashes 3. Digital signatures 4. Tamper resistant protocols	• Attribute and role based access control is used to restrict access to selected operations • Role based access control, location and context are used to authorize access to resources • No authorization tokens and credentials sent in clear text • IFC compliant policies are used to protect data integrity • Confidential data only accessible to authorized parties

Repudiation	1. Digital signatures 2. Timestamps 3. Audit trails	• Only verified and signed attribute certificates by security authorities can be accessed or exchanged
Information Disclosure	1. Authorization 2. Encryption 3. Protect secrets 4. Don't store secrets in the clear	• Program Dependence Graph (PDG) compliant policies are used for information flow • Minimum set requirement satisfaction of context and location is used to grant access to resources and allow information flow • Only Attribute certificates authorized can disclose information • Encrypted data in storage
Elevation of privilege	1. Run with least privilege	

Note: this work's focus is on addressing threats relating to the CIA-triad. Although the other security properties are listed or covered on the STRIDE categorization with their countermeasures, however, they are not the primary focus of this work.

3.7 Discussion

In this chapter we outlined the threat model that can be employed by an adversary to attack our model. We derived and summarized the threats and vulnerabilities by making use of the system application's key components that could be targeted by an attacker. These include the entry points, assets, external dependencies, etc. Moreover, we identified the type of attacks that can be

used by an attacker to compromise the system or attacks that the system can be prone to. These attacks include shoulder surfing, brute force attack, dictionary attack, eavesdropping or spoofing and SQL injection. Consequently, we also specify methods that can be used to circumvent or prevent the above mentioned attacks from taking place or deployed. These circumvention methods or countermeasures include but are not limited to using appropriate authentication and authorization techniques like protecting tokens and credentials and storing them in encrypted formats, storing passwords with salted hashes, not sending tokens and credentials in clear text, employing IFC compliant policies which enforce approved authorizations for controlling the flow of information within the system to protect data integrity and meeting the minimum set requirements for location and context to grant access to resources.

Chapter 4
System and Security Model

The threat model presented in Chapter 3 details the threats and vulnerabilities that can impact our access and information flow control mechanisms in the system. In order to circumvent these threats, we need a comprehensive model or prototype that will act against the threats and vulnerabilities presented in Section 3.2. In this chapter we present both the access and information flow control security mechanisms used to make our model secure from the identified threats. We detail the access control scheme used and the way(s) we employ to achieve secure flow of information when mobile web services compose in resource constrained environments taking into consideration location and context dependencies in both authentication and authorization. Furthermore, we present the system and security model adoption guidelines as described in section 4.1 below.

4.1 Design Overview

This chapter details the systematic process that was followed in order to form the guiding principles for designing the model that we used for implementing the system for this work. We explain the research design objectives and the methodology we used to achieve the design objectives. Therefore, we focus on three principles: the research design chosen, methodology and model/prototype design and specification.

This research study adopts an experimental approach [99]. According to Vessey et al. [99], experimental research relies on systematic manipulation and testing to measure changes in variables. As indicated in Chapter 2, various works related to this thesis have produced different models and designs to address the different Information Flow Control issues. Thus, on the basis of these secure information flow issues, this research seeks to identify a solution to a central problem of how to specify a security policy in dynamically changing security domains to ensure secure information flow without violating minimum security requirements for all the participating services. Hence, this research design was appropriate for this work as it is important

to test variables such as the authentication/authorization and integrity of mobile service compositions when they occur, so as to come up with a security policy on how to prevent illegal flows of information.

The research design involved mapping and narrowing this thesis to address two constraints that can impact security in mobile web services namely: location and context with respect to secure information flow. The main idea here was to come up with new results on addressing the constraints mobile web services have in security policies during service compositions and what difference mobile web services make. Therefore, these constraints (location and context) will be evaluated on how they are addressed, that is, how they impact the specified access control scheme and information flow control.

4.2 Prototype Design

Drawing considerations from previous related work, we derive our prototype based on the prototype defined by She et al. [103] of a web service system; Figure 4.1 depicts the elements of a mobile web service system which consists of a security domain where data resources, services, security authorities and service composers are found. Each service composer generates services to form a composite service and each service has access to data resources. Also, each service is monitored by a security authority of that particular security domain for compliance with the domain's security policies. Our work therefore defines a mobile web service system as follows:

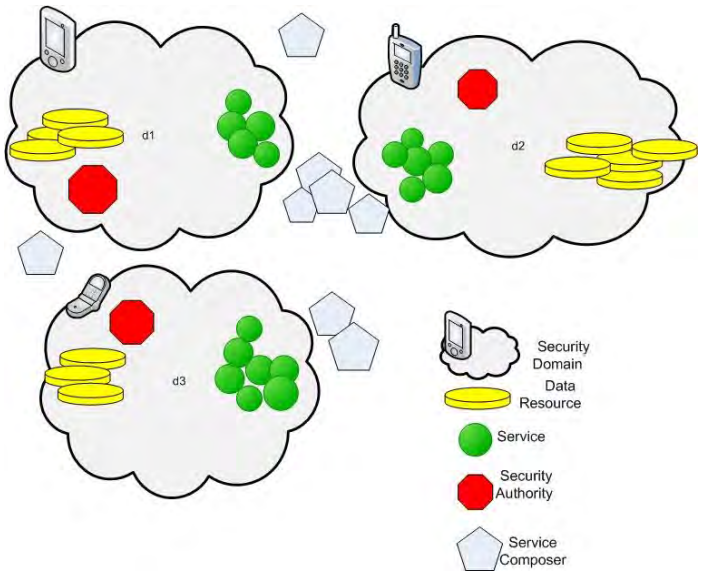

Figure 4.1: Mobile Web Service System

Definition 1: A mobile web service system includes a set of domains $\{d_1, d_2,\ldots\}$, a set of locations $\{l_1, l_2\ldots\}$ and a set of service composers $\{q_{comp1}, q_{comp2},\ldots\}$. Each domain d_i is a tuple $<d_i.S, d_i.R, d_i.sa>$, where $d_i.S = \{d_i.s_i, d_i.s_2\ldots\}$ is the set of all services in d_i, $d_i.R = \{d_i.s_1, d_i.s_2,\ldots\}$ is the set of all data resources in d_i and $d_i.sa$ is the security authority (SA) of d_i. $d_i.sa$ manages a set of access control policies $d_i.Pol = \{d_i.pol_1, d_i.pol_2,\ldots\}$ to control the access to $d_i.R$.

Data resources refer to the data/information itself and any object that may store or receive data/information. Such an object can be a data container, such as a file, directory, a relation, a view, etc. also, we assume that all services are honest-but-curious, that is, they follow the protocol and conform to the access control policies defined, however, these services may attempt to derive sensitive information of others from the information they have received or trying to access. They may derive such information by (cheating, sharing their private information) gaining information about other services' private input sets, other than what can be deduced from the result of the protocol.

4.2.1 Location

The revolution in the field of hand held (mobile) devices has been upward in the last couple of years. And this has been made possible by improvements in processor speeds, screen size, graphic quality and more importantly due to the advancement and emergence of new

technologies that make transfer between a server and a mobile device faster. Furthermore, mobile devices' (mobile phones in particular) user base is rapidly surpassing the use of personal computers (PCs), notebooks and laptops in developing countries or rural areas. Hence, the need for mobile web services development is playing a significant role in providing services in remote/rural areas where without such mobile service developments; it would be difficult for people to have access to these services. Given, that mobile web services offer solutions to where traditional web services can't tap into, mobile web services face the biggest challenge of speed and connectivity which is very limited and which their traditional counterparts enjoy in wired connections. Therefore, it is imperative to consider that their deployment is largely dependent on the wireless/cellular network coverage provided by the service provider.

However, in this work our emphasis and focus is on the impact location has in regards to changes in security domain with heavy reliance on the access control model we define. We look at or try and resolve the challenges faced by service compositions when the security domain changes and how this affects information flow control. In the mobile environment just like in smart spaces or cloud computing, changes in security domains are both unavoidable and autonomous. This is because in mobile web services the relationship between users and resources is dynamic and more ad-hoc. Users and resource providers are generally not located in the same security domain. Therefore, there is a need for a flexible or dynamic access control model to handle such changes with relevant ease without putting a strain or compromise on both their performance and security. We define location as the physical position at a given time when a security domain changes. Therefore each service is bound to a particular location at each security domain during composition given by definition 2 below;

Definition 2: $l_i.c_i.S = \{s_1.l_i.c_i, s_2.l_i.c_i,\ldots\}.d_i$ is the set of all services in $l_i.c_i$ and $l_i.c_i$ is in dom d_i, where $l_i.c_i$ is the location of service i under context i.

As a result, merging definition 1 and 2, we define a mobile web service as follows;

Definition 3: A mobile web service system includes a set of domains $\{d_1, d_2,\ldots\}$, a set of locations $\{l_1, l_2\ldots\}$ and a set of service composers $\{q_{comp1}, q_{comp2},\ldots\}$. Each domain d_i is a tuple $<d_i.S, d_i.R, d_i.sa, l_i.c_i>$, where $d_i.S = \{d_i.s_i, d_i.s_2\ldots\}$ is the set of all services in d_i, $d_i.R = \{d_i.s_1, d_i.s_2,\ldots\}$ is the set of all data resources in d_i, $d_i.sa$ is the security authority (SA) of d_i. The security

authority, d_i.sa manages a set of access control policies d_i.Pol = {d_i.pol_1, d_i.pol_2,…} to control the access to d_i.R and l_i.c_i is the set of locations for S in d_i under context c_i.

In general, a composite location can be defined using a workflow, which is a composition of component locations. As each service is bound to a particular location, it suffices that in an abstract workflow; each component location is abstract and is to be grounded to a concrete location. And in a concrete workflow, each component location composes the mobile web service. This is because from definition 2 above, each service is not complete without being bound to a specific location. Fig.4.2 illustrates how location impacts or influences the composition of services in different security domains. Note that in security domain C the service has two locations, location 3 and 4, clinic and home respectively. Therefore, the service can be bound to either one of the locations; however, the access control policies enforced towards the same service will differ even though both locations are in the same security domain. As a result, location is an important constraint that needs addressing when employing a fine-grained access control model in dynamic environments when security policies change autonomously.

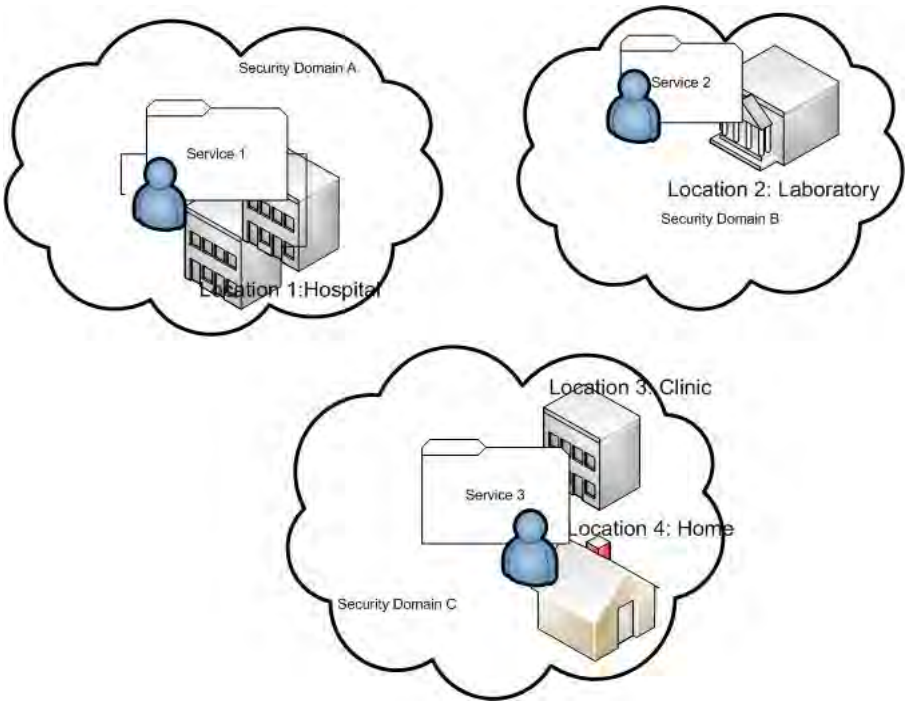

Figure 4.2: services depiction bounded by location

Hence, we define a role-based access control model to handle these changes in both roles and location. Roles are assigned dynamically based on the participating service's or user's trust level

and they help manage access to the resources. The role separation allows simplifying policies and makes them easy to adapt or configure.

The Role Based Access Control model to handle these dynamic changes is covered in Section 4.4. The trust level calculation is based on the participant service or user's context, which includes identification attributes, location, device type, etc. Moreover, access control rules are used to calculate the trust level, assign roles based on the trust level and to grant permissions to requested services/resources. Section 4.4 covers Role Based Access Control in detail.

For experimental purposes in Chapter 5, we look at how a change from a high level security domain (e.g. a secured network connection) to a low level security domain (open network) impacts the access control model and its information flow control. We observe what exactly happens during the change in security domain at a particular location; we look at if the user is required to re-authenticate and if there are any disruptions on the service at that particular time/period.

4.2.2 Context

The uses of context-aware services improve the way users browse information on constrained devices like mobile devices. Services which are context-aware pro-actively select information using context variables that are gathered from the device's environment. Additionally, the context of a user (i.e. location, time, system resources, network state, user's activity, battery power level, etc.) in mobile web services is highly dynamic, and granting a user access without taking the user's current context into account can compromise security as the user's access privileges not only depend on "who the user is" but also on "where the user is" and "what the user's state is and that of the user's environment".

We address the constraint issue of context by addressing how it impacts the secure flow of information and how the access control model defined handles accesses to resources based on surrounding circumstances. As earlier defined in Subsection 4.2.1, in our case context relates to the user's ID, device type, role, and environment. We examine security levels, how public information can be published or made available and processed by all participating services,

however, and the private information provided remains only for authorized/appropriate participants when corresponding access permissions are granted. Therefore, the user's identity (ID), role is taken into account when requesting a service/information at the given security level. For instance, at a given change in security domain, is the service requester cleared to have access to private/sensitive information or authentication is needed first for such to take place based on the user's current context.

Therefore, in our work, context impacts heavily on secure information flow because when the security domain changes in a particular location during composition, attributes of who the user is or what the user's state is must be addressed in order to allow the dissemination of information to occur. Table 4.1 gives a summary of the access control policies with the location and context dependencies. Hence, if the user or service's context defined in that particular domain is not satisfied during the composition, even though the location may meet all requirements, information will not flow. Also, note that even though the right context may be defined in the correct security domain but at a wrong location, access or the flow of information will not be granted. Likewise, giving the right context and location but wrong domain will result in denied access or no flow of information.

Table 4.1: access and information flow control conditions and outcomes with location and context dependencies

User Role	User Location	User Context	Security Domain	Access/info flow
1	✓	✓	✓	Grant
2	✓	✓	X	Deny
3	✓	X	✓	Deny
4	✓	X	X	Deny
5	X	✓	✓	Deny
6	X	X	✓	Deny
7	X	X	X	Deny

✓ Condition met X Condition not met

4.3 Service and Service Chain

We model the flow of information in service chains using an abstract dataflow model [103]. In this model, each service *y* (bound to a location *l* and satisfying context *c*) takes the input data *y.In* from the end user or another service *x*, and finishes its own calculations, and produces its output *y.Out*, which is sent to another service or the end user *z*. The computation of *y* may use some data resources stored in *dom*(*y*), i.e., *y.R*. As we only consider the deterministic system, the set of output data of *y*, *y.Out*, can be expressed as a function of its input, *y.In*, and local data resources, *y.R*.

Since a composite service is a composition of component services, it can therefore be defined with a workflow. We consider abstract and concrete workflows. An abstract workflow is one which has information about all the services needed to compose the required composite service and a concrete workflow is one that has specific services for the required service to compose. In an abstract workflow, each component service is abstract and is to be grounded to a concrete service. In a concrete workflow, each component service is a concrete mobile web service. On composition, the service composer is given the desired abstract workflow and denotes each instance of an abstract component service by a concrete service. Our work only considers a simplified workflow, a service chain. We define an abstract and concrete service chain as follows:

Definition 4: An abstract service chain $<s_0, as_1.l_1.c_1, \ldots, as_n.l_n.c_n, s_{n+1}>$ consists of two end users, s_0 and s_{n+1}, where s_0 is the user who sends the input data to as_1 and s_{n+1} is the user who receives the output data from as_n, and a sequence of abstract services, $as_1.l_1.c_1, \ldots, , as_n.l_n.c_n$, that should be grounded to concrete services. A concrete service chain $<s_0, s_1.l_1.c_1, \ldots s_n.l_n.c_n, s_{n+1}>$ consists of the two end users, s_0 and s_{n+1}, and a sequence of concrete services $s_1.l_1.c_1, \ldots ,s_n. l_n.c_n$ where s_1 is bound to location l_1 under context c_1.

We consider that each abstract or concrete service is bound by a component abstract or concrete location as earlier defined. A concrete location is one that has a specific location and an abstract location is one that can take on one of all the possible locations at a specific time but where the exact location is not known at runtime. This means that for instance in a concrete service chain, the two end users, s_0 and s_{n+1} are in fact $s_0.l_0$ and $s_{n+1}.l_{n+1}$ respectively. We also consider that

when a user submits an abstract service chain to a service composer, the service composer returns a concrete service chain with the two end users s_0 and s_{n+1} included in the chain as services. They may be the same user or may be different. Therefore, to guarantee that the correct user is making a service request, we need to make sure that each user is defined as a correct user. Section 4.4 below defines how the users are given roles to use when making service requests.

4.4 Role Based Access Control (RBAC)

In general web services' access control is implemented using two main security practices; authentication and authorization. Authentication basically defines how to establish identity and authorization permits or denies that identity to access resources. Therefore, to achieve this in our mobile web service prototype, we are going to make use of a RBAC model, with an extension to an attribute-role based access control.

As established by [81], the fundamental concept of RBAC is to introduce a "Role" between users and permissions. An administrator defines various roles according to demands and sets the access authority according to the role while users are assigned to different roles according to their responsibility. In this way, the access authority can be assigned to a certain role and users can get the access authority owned by roles through playing various roles. Fig.4.3 below illustrates the basic relation among user, role and authority/permission in a RBAC model and Fig.4.5 illustrates the basic logical implementation architecture of a RBAC model. We define a role as follows;

Definition 5: *Each service or data resource x is associated with a set of roles* $Role(x) = \{rol_1(x), rol_2(x), \ldots\}$. *Each role* $rol(x) \in Role(x)$ *is defined as a tuple (rol(x).name, rol(x).location) in which rol(x).name is a string that uniquely specifies the name of the role, and rol(x).location is a string that uniquely specifies the location of the role.*

Each service owns a set of attributes and these are the roles, location, context, etc. In order for a service requestor to be granted access to resources or data, the service's location and context are taken into account bound by the role granted to that particular requestor.

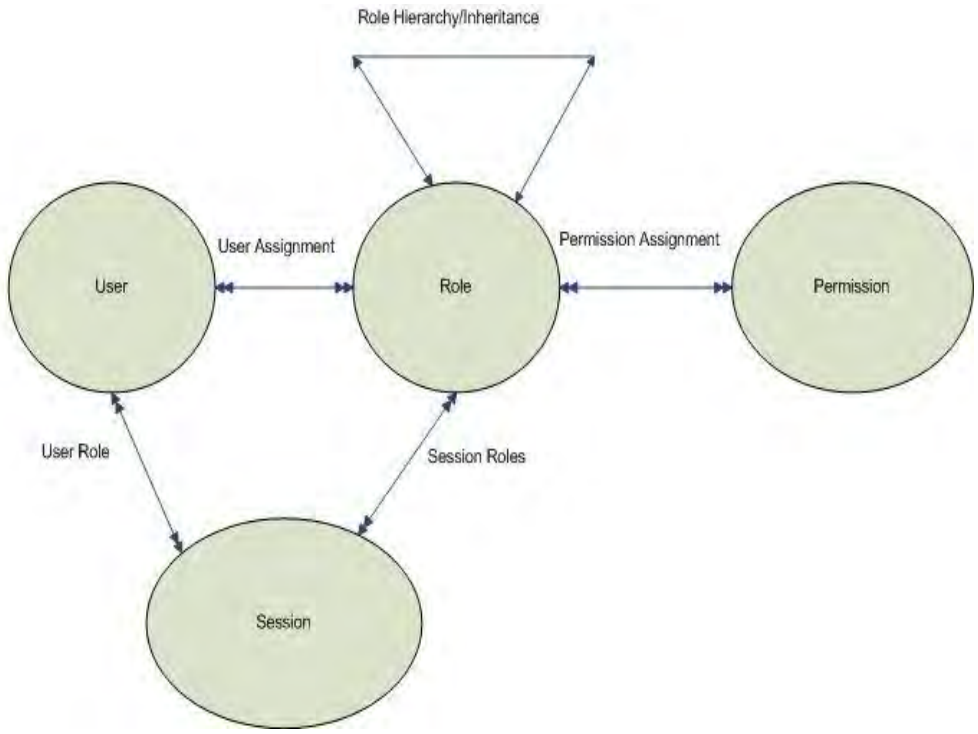

Figure 4.3: The Basic RBAC Model: Relationship between users, roles, permission and session.

This access control model is a base for our work taking into consideration the constraint attributes (location and context) being addressed. The concept of role hierarchy is an important component in addressing location aware access control in particular since different users are assigned roles in different security domains, location and context. See example Figure 4.4 below which demonstrates role and location changes.

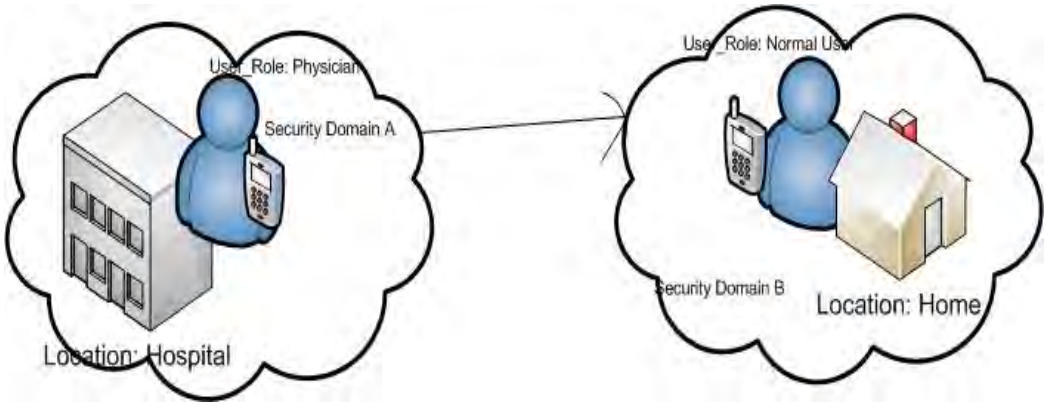

Figure 4.4: Role change depiction of a user under different security domains and location

Figure 4.4 above illustrates a role change of the same user when security domain and location changes. The same user is assigned different roles in different locations when there's a change in

security domain. This also holds true for the same user in the same security domain but different locations. However, a super user for instance a system administrator can have the same role across the board irrespective of what domain and/or location he/she is at. An example role hierarchy in a healthcare environment is a case whereby a health care provider's role is inherited by a physician and the physician's role in turn inherited by a specialist physician or a primary care physician. Therefore, the specialist physician in this regard can be considered a super user of the system, thus inheriting all the roles and privileges/permissions given to all users.

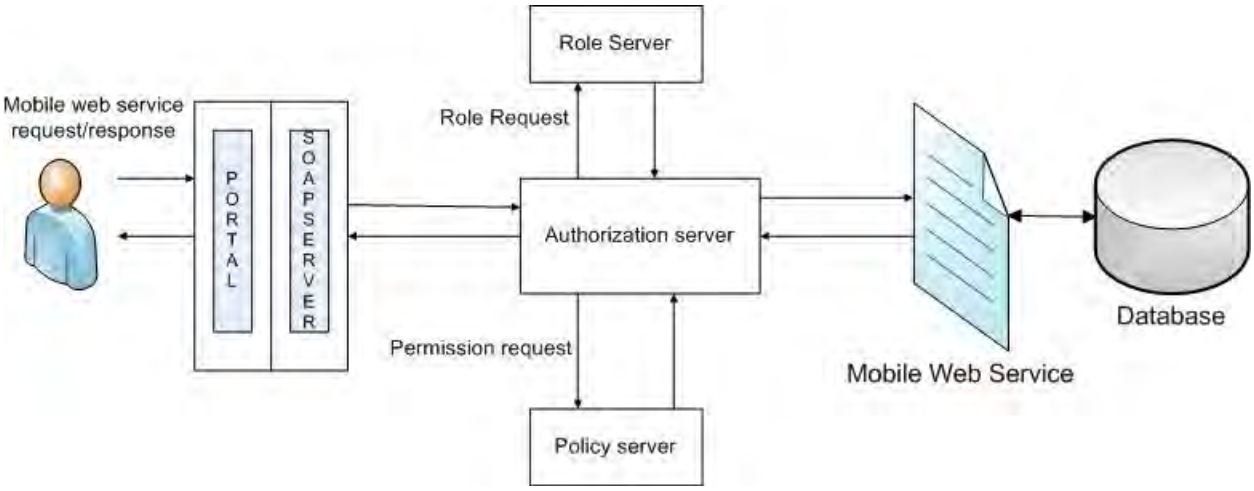

Figure 4.5: The logical implementation of access control

To illustrate the logical implementation of our access control (RBAC) in relation to Figure 4.5 above, we use an Apache Axis framework to publish the Mobile Web Services [98]. This framework is based on Java Web Technology (Servlet). An Axis Handler is used to perform authentication and authorization control. The data which is service definition, user definition and permission definition is stored in a MySQL database. We integrate with open standards to use Security Assertion Markup Language (SAML) [87] encoding for representing user authentication, user-role assignment, and permission-role assignment. As an example, take a physician user, we describe the working process as follows.

1. User (a physician) logging on portal;
2. Portal capturing user's authentication and authorization credentials;

3. Portal creating and signing SAML assertion, and placing SAML in a Simple Object Access Protocol (SOAP) message;
4. Portal sending SOAP message to Mobile Web Services;
5. Mobile Web Services Handler accepting or denying request to Mobile Web Services based on original user's role (RBAC);
6. Mobile Web Services Handler finally sending message to Mobile Web Services;
7. Mobile Web Services processing;
8. Mobile Web Services sending response back to portal.

Drawing similarities from the She et.al [102, 103] access control model, an attribute-based access control model is considered for this work as an extension of the RBAC model to help in handling dynamic access to resources. The model we now define combines both attributes and roles. Each service or data resource is complemented by a set of attributes defined. These attributes may comprise service name, the Web Services Description Language (WSDL) pointer, the permission granted to the service, role, reputation, etc.

Each data resource's attribute may contain owner, security level, etc. Attributes of a resource are incorporated in the metadata and held in reserve with the data. All service attributes must be affirmed by a security authority (SA) and included in a certificate, called the attribute certificate. An issuer of an attribute certificate must sign each and every attribute certificate the issuer hands out.

As a rule of thumb, we consider that in each security domain, a SA manages the attribute certificates of all the services in that particular domain. Furthermore, attributes of a service can be sensitive (secret) or non-sensitive (public). Public attributes can be shared or exchanged freely among participating services without meeting specified access rules. However, secret attributes needs to be negotiated i.e. authorized participants have to be evaluated to meet the access control policies in place before allowed access to attribute certificate.

The basic concept of attribute-based access control represents a logical access control model which controls access to objects by evaluating rules against attributes of the entities (subject and object) actions and the environment relevant to a request. Also, an access control policy is a set of conditions defined over the set of all attributes used by the domain (the set of all attributes

defined for services and resources in the domain). To simplify our model, we consider a unified set of attributes defined across all domains. Therefore, when a service A accesses a data resource b, A presents its attribute certificate, which contains a set of attributes to domain b. Domain b's SA verifies A's attribute certificate from its issuer. A's attributes are evaluated against the access control policies of service B. If they correspond, access is granted else access is denied.

4.5 Addressing Information Flow Control

Information flow control is a critical aspect of this work and is addressed first by ensuring that the access control requirements of the service are met by all participating services. This is to guarantee not only that direct access to resources is considered but also for indirectly interacting services. For example, considering a service chain <A, B, C>. Assume that B's output is computed from some of its own sensitive information and some sensitive data received from A. When B's output is sent to C, C may use the received data to derive the sensitive information of A, resulting in an information flow from A to C (a running example to illustrate this concept is given in section 4.5.1). Therefore such information flows, if not handled carefully, may result in undesired information leakage. Relevantly, in our case a service with a high security level in a high security domain leaking sensitive information when the security domain changes (secured connection to unsecured/open connection) from a high to a low security domain. Henceforth, our scheme (use of Program Dependence Graphs (PDGs) and path conditions combined with the access control model) will address such challenges when changes in security domains occur. This will be enforced by meeting the path conditions like trust levels (is service requester permitted to revoke a service given its participating context, roles given and/or permissions assigned).

4.5.1 Dataflow with Program Dependence Graphs (PDGs)

We use the concept of PDGs to enforce the secure flow of information between services from potentially different security domains with possibly different access control policies. We consider the motivating example application workflow presented by She at al. [103] (Fig. 4.6) to address and demonstrate the need for the IFC in service composition and the benefit of considering access control at composition time.

The following rules apply for dataflow to take place;

- Service *a* has to be authenticated first in order to access resources of another service
- Information can only flow from one service to another when path conditions are met. These conditions include trust levels, authentication verification.
- Information flow is achievable by using PDGs

Additionally, we use the example presented by [28] to help demonstrate the concepts (of PDGs) in our model. A workflow is used to help with screening of disease x by first extracting association rules from medical data of patients with and without disease x. Association rules are then used to determine how likely a new patient does have disease x. The workflow consists of the following abstract services; a client program CLN, a medical database MDB, a template image database TDB, an image enhancement service IES, an image registration service IRS, an object recognition service ORS, an association rule mining service ARM, and a classifier CLS as shown in Figure 4.6 below.

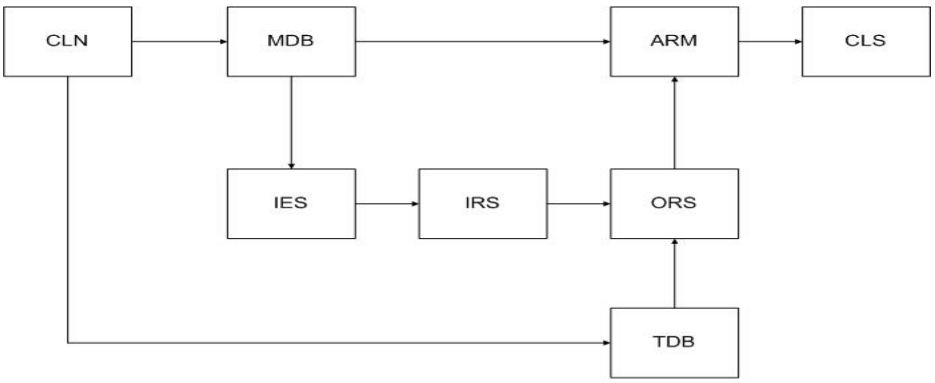

Figure 4.6: Example workflow

CLN first searches MDB (with keyword x) for the medical records of the patients who are diagnosed to have the disease x, and searches TDB (with keywords such as “bone,” “polyp,” “nodule,” and so on) for the template images for object recognition. Each medical record stored

in MDB includes the alphanumeric medical data, for example, the patient's medical history, family history, personal data (e.g., gender, age, height, weight, living area, etc.), and the medical images (e.g., CT, X-ray, Nuclear, etc.). The alphanumeric medical data of MDB are sent to ARM. The template images are sent to ORS. The medical images are first sent to IES, which performs image enhancement (e.g., noise cancellation, etc.). The enhanced images are sent to IRS, which performs image registration to align different images into one coordinate system. The aligned images are sent to ORS, which detects and recognizes the objects in the images (e.g., bones, polyps, nodules, etc.) using the template images.

After recognition, it assigns labels to the recognized objects in the image. The labeled images are sent to ARM, which uses these images together with the alphanumeric medical data received from MDB to extract association rules that are sent to CLS.

We now consider the sensitivity of the data that are used by the composite service (Note that this abstract composite service includes three abstract service chains). We assume that the search keywords that CLN sends to MDB and TDB are sensitive and, hence, require protection and the recipients are required to have read permissions to these data. The alphanumeric medical data that MDB send to ARM and the medical images that MDB send to IES are also sensitive and the recipients are required to have read permissions to these data as well. (Note that the recipient rather than the invoker needs to have the proper privilege. For example, IES needs to have read permission to the medical images in MDB, but CLN does not.) The template images are used in a pay per- use manner and, hence, require the recipients to present proper privilege.

Next, consider the concrete services that can be used to instantiate the abstract services and the privileges they have (Fig. 4.7). For simplicity, we assume that CLN, MDB, TDB, IES, IRS, and CLS are already concretized by *cln1; mdb1; tdb1; ies1; irs1*, and *cls1*, respectively. From these concretized services we can derive a PDG that ensures a secure flow of information between these services. Each service is a node control dependent on the node invoking it. Moreover, set conditions (like trust levels) are used as path conditions (necessary conditions for information flow between nodes/services) between the nodes. For instance, for *cln1* to invoke access to *mdb1* (which contains the patient's details), authentication and authorization conditions have to be met before any further computations or interactions can take place. *cln1* and *cls1* are hosted by hospital A (domain A). *mdb1* and *tdb1* are hosted by hospital B (domain B) and research institute

C (domain C), respectively. *ies1* and *irs1* are hosted by research institute D (domain D). ORS can be instantiated by *ors1; ors2,* and *ors3*. Note that ies1 can modify the content of the medical image received from *mdb1*, and *irs1* can modify the content of the images received from *ies1*.

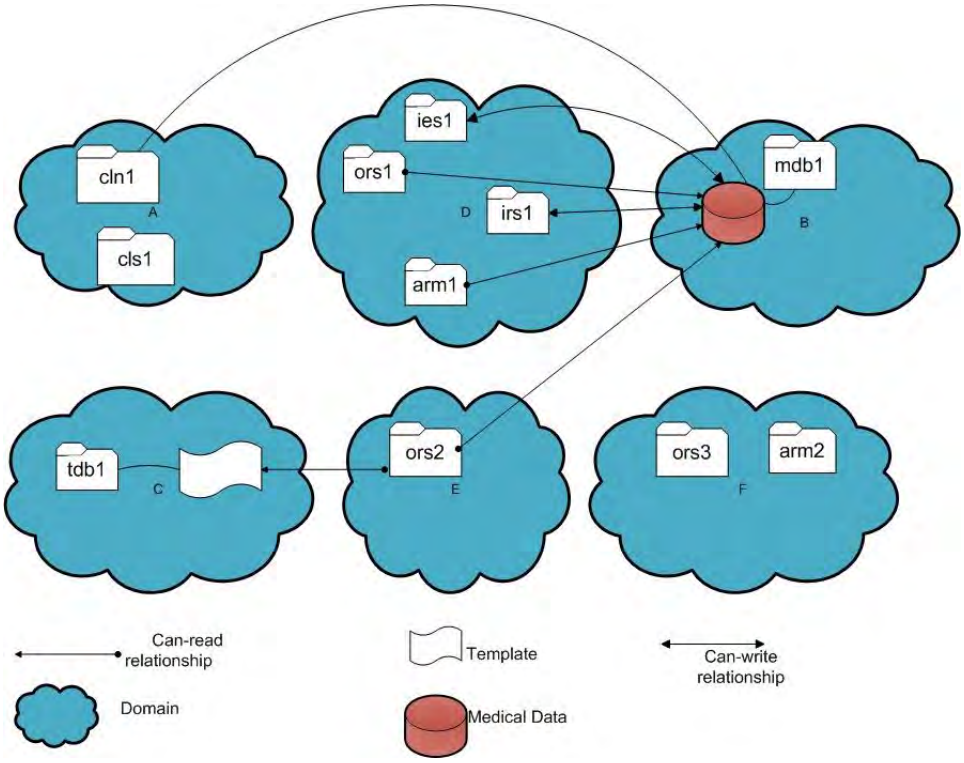

Figure 4.7: Concrete services and their permissions

Hence, the medical images of *mdb1* are essentially delivered to the ORS service (*ors1; ors2, or ors3*) in their modified forms. ARM can be instantiated by *arm1* and *arm2.* We consider that ors1 and arm1 are hosted by institute D, *ors2* is hosted by research institute E (domain E), and *ors3* and *arm2* are hosted by university F (domain F).

For simplicity, we assume that all the services can be invoked by anyone and we only define the resource-based access control policies here. Consider that service *x* invokes service *y*. If *y* does not read/write any sensitive local data (e.g., a table, etc.) in its computation, then the invocation can be directly granted. If *y* reads some sensitive local data resources *r*, then the invocation is granted when *x* has the read permission to *r* and its location is concretized with the service and deemed to be secure. Similarly, if *y* writes to *r*, then the invocation is granted when *x* has the

write permission to r and is at the right location. Fig. 4.7 depicts the resource access permissions. Table 4.2 shows a simple access control matrix used for MDB. We assume there are two kinds of services, i.e. admin management service and medical records management service. Admin management service provides a function of maintaining user table (where all the users login credentials are added and stored) and the medical records table (where all the users' medical data is kept), while medical records management service includes services of View_MDB, Edit_MDB and Delete_MDB. Three roles including patient, nurse and physician have different operation permissions to access user table and medical records table as shown below. We consider that institutes D and E are federated with hospital B and, hence, *services ies1; irs1; ors1; arm1; ors2* have the read permission to the medical data in domain B. Also, we consider that institute E has purchased the service of *tdb1* from institute C, and hence, *ors2* has the read permission to the template images in domain C. No other read/write accesses to the medical data or the template images are allowed.

Table: 4.2: Access control matrix employed for the Medical Data Records

Function / Role	View_MDB	Edit_MDB	Delete_MDB	Maintain Users and MDB
Patient	✓	X	X	X
Nurse	✓	✓	X	✓
Physician	✓	✓	✓	✓

4.5.2 IFC Illustration with PDGs in Mobile web services

Employing the concept of PDGs, we consider a model whereby a mobile web service is an entry node in a dataflow model to perfect the flow of data/information between services or composing services. Note that subsequent nodes under the entry node are attributes of the entry node in the same security domain. However, other services can also be invoked from different security domains to form the required composite service solution. These subsequent services are control dependent on the entry node if they are invoked to complete the desired solution even though

they may originate from a different domain. Hence their participation is dependent on the node invoking them (entry node or subsequent node) to form part of the required composite service. Therefore they need to be authenticated and authorized in order to participate. For services to allow the flow of information between them, we derive such flows from Program dependence graphs defined by [28] as follows;

Definition 6: a data/information dependent service A → B means that service A assigns or disseminates information used by B (without being released elsewhere underway or without being assigned to any other service thereafter) [28].

Note that although Hammer uses PDGs as standard tools to model information flow through a program, we extend this notion to suffice for modeling the flow of information through or between web services. Therefore, program statements used to resemble graph nodes will resemble services in their initial or invoked states. Also, note that a graph node is mapped to the given context of the service. For instance, a node may be comprised of a service in a given network state. The change of the network state may mean a change in context for that particular node. Thus a control dependence of service A → B means that B's use of the information depends on the context given by A. Take for instance in our running example, *ies1* can only enhance the image given by *mdb1* if the request from *mdb1* is for enhancement, say noise cancellation. Then that is when *ies1* can act on that desired request. This is done by using path conditions, which according to Hammer [28] are necessary conditions for information flow between two nodes. Although Hammer uses typical conditions in while or if loops, in our case these conditions may include the roles and/or permissions given to each service requesting the service from A or any other service. Therefore, since *ies1* is dependent on *tdb1* for carrying out its operations after meeting the set conditions, it then suffices to conclude that there is a path from *tdb1* to *ies1*. This relatively means that information can flow through that path. However, each service requesting information/data from a service higher in the security level must be cleared or authorized to do so without leaking any information that is sensitive or private regardless of which security domain it is in, given its context at participation time.

Fig.4.8 below illustrates an example of a simple PDG extracted from the concrete services and their permissions in Fig 4.7. The figure shows a path from the entry node, *cln1* accessing the medical data in *mdb1*, with *ors2* granted access to *mdb1* and in turn *tdb1*.

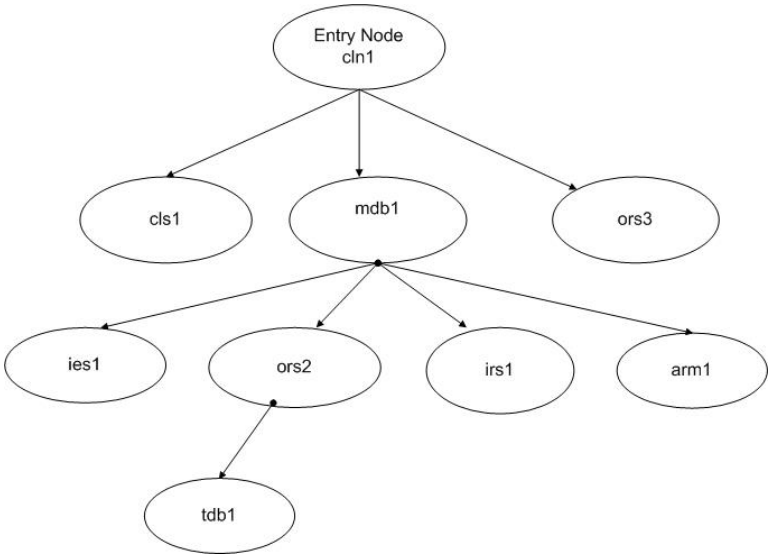

Figure 4.8: Example PDG derived from concrete services and their permissions

Henceforth, a path A $\rightarrow^{*}$ B means that information can flow from A to B; likewise if there is no path, then it is guaranteed that there is no information flow. As further validated by [28, 29], PDGs relate very well to the traditional notion of secure information flow noninterference. Noninterference between two security levels, written as d /~> e means that no statement with security level d may influence a statement of security level e [28]. This concept maps up well even for our case, as no service with security domain d may influence a service of security domain e without satisfying the desired conditions needed.

In Fig. 4.8, *ies1* cannot under any circumstances have an influence on *tdb1* because no path is specified for *ies1* to engage or invoke *tdb1*, similarly for *irs1* and *arm1*. Furthermore, if *ors2* misrepresents or lie about its location when requesting access to *mdb1*, the access request will be denied and *mdb1* will set its trust level to a red flag to close the path for information propagation to *ors2*. As a result, both the access control and information flow control policies will fail to execute rendering a request denial for *ors2* to collaborate with *mdb1*. As a result, the use of PDGs in our work put an emphasis on determining if there is an information propagation from service *a* to service *b* or whether this is without doubt not the case.

4.6 Discussion

We propose a sound prototype that uses Role Based Access Control to monitor requests to access resources and uses the concept of Program Dependence Graphs to monitor information

propagation once access to the resources or information is granted. Furthermore, our scheme takes into account the location and context dependencies in authentication and authorization of information flow services in mobile environments. Services are bound to location and context to allow the secure flow of information to take place. Path conditions which are necessary conditions for information to propagate between services are also introduced. If these path conditions are not met, it is guaranteed that there'll be no information flow. Likewise, if location and context are not verified or falsified by the service requestor, no access is granted to resources thus unauthorized users are kept out of the system.

Chapter 5
Implementation

5.1 Overview

In Chapter 4 we discussed the architecture of our model prototype at a generic level, stating components the model employs and how the components should interact. The formulation of the model prototype features aspects that need to be implemented so as to verify the security of the model both in individual components and as whole. These include implementing or verifying how access to resources is achieved, how information propagates between services once that access is granted. Furthermore, we look at how PDGs are set to realize the secure flow of information. Therefore, in order to assess the overall effectiveness of the prototype design described in Chapter 4, a set of experiments were implemented to answer the following questions on the model;

1. How long does it take to generate a tree of passwords?
2. How to check how secure the model is?
3. How much time does logging on the system take?
4. Finally, we test the number of security policy changes when security domain changes.

5.2 Technologies

We implement our analyses using an application web server platform called WAMP. WAMP is a software bundle running on a Windows platform and consists of an Apache server, MySQL database and a scripting language called PHP. We briefly describe each software component below;

- ✓ Windows 7: A graphical operating system (OS) developed by Microsoft running on most personal computers.

- ✓ Apache: This is the most popular Hypertext Transfer Protocol (HTTP) server in deployment on the public internet. It is an open source software which supports a wide variety of features which are implemented as compiled modules.
- ✓ MySQL: This is a multithreaded, multi-user SQL database management system (DBMS).
- ✓ PHP: This is a server-side scripting language designed for web development but also used as a general purpose programming language. PHP code is interpreted by a web server via a PHP processor module, which generates the resulting application. PHP commands can optionally be embedded directly into an HTML source document rather than calling an external file to process data. We use PHP as the implementation language for our model.

5.2.1 PHP RBAC

For relevant ease of implementation, this work uses PHP-RBAC [1] which is a standard NIST level 2 Hierarchical Role Based Access Control library implemented as a library for PHP (it is a de facto authorization library for PHP). This standard allows perfectly maintainable function access control for applications or frameworks. PHP-RBAC relies heavily on an SQL or MySQL backend for fastest implementation. Furthermore, this standard provides ease of use and reliability.

5.3 Structure

For our analyses we created program stubs using PHP to implement a hierarchical RBAC model for our native access control. Figure 5.1 below presents the user clearance tree hierarchy and Figure 5.2 presents the stubs of code used to implement a tree of passwords or roles for the five different users, namely; root, system admin, moderator, public user or normal user and a guest user to handle their access to resources as outlined in the comments. The hierarchal RBAC model defines an inheritance relation among roles. The model resembles a tree structure with root as the tree top inheriting user roles and permissions of subsequent leaves or nodes below root. In principle, root has access to all resources. System admin has access to all resources below its password clearance level, likewise a moderator and a normal user and finally a guest user has access to resources only directly assigned or cleared to access.

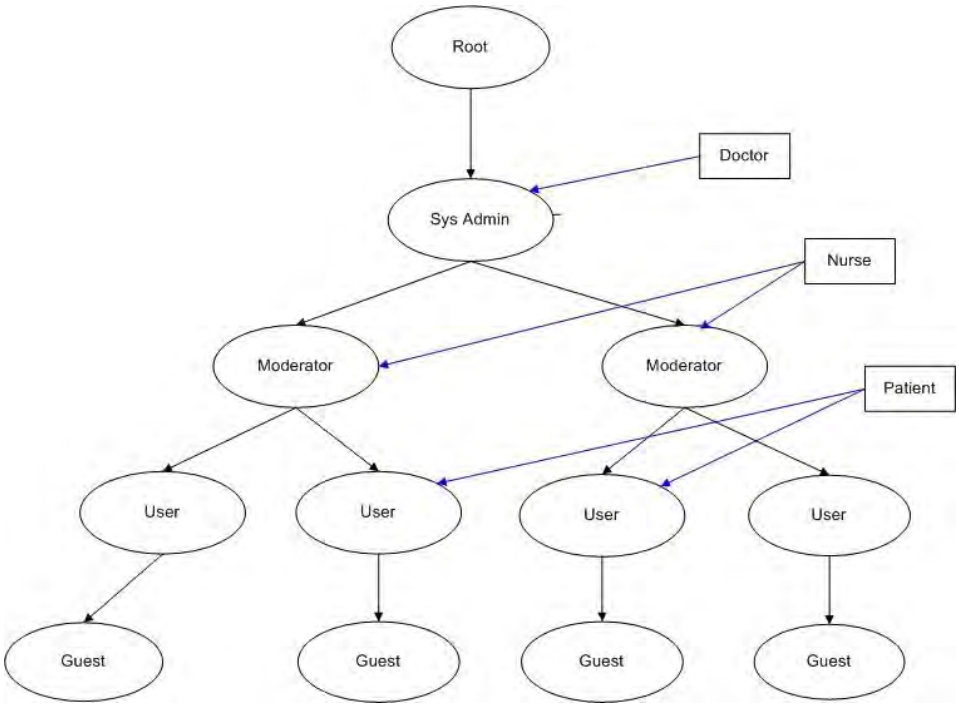

Figure 5.1 Role hierarchy tree structure: Roles and Users

```php
<?php
use yii\rbac\Item;

return [
    // MANAGEMENT TASKS
    'manageResource0' => ['type' => Item::TYPE_OPERATION, 'description' =>
'...', 'bizRule' => NULL, 'data' => NULL],
    'manageResource1' => ['type' => Item::TYPE_OPERATION, 'description' =>
'...', 'bizRule' => NULL, 'data' => NULL],
    'manageResource2' => ['type' => Item::TYPE_OPERATION, 'description' =>
'...', 'bizRule' => NULL, 'data' => NULL],
    'manageResource3' => ['type' => Item::TYPE_OPERATION, 'description' =>
'...', 'bizRule' => NULL, 'data' => NULL],

    // CREATING THE HIERACHICAL ROLE/TREE STRUCTURE
    'guest' => [
        'type' => Item::TYPE_ROLE,
        'description' => 'Guest',
        'bizRule' => NULL,
        'data' => NULL
    ],

    'user' => [
        'type' => Item::TYPE_ROLE,
        'description' => 'User',
```

```
        'children' => [
            'guest',
            'manageResource0', // User can edit Resource0
        ],
        'bizRule' => 'return !Yii::$app->user->isGuest;',
        'data' => NULL
    ],

    'moderator' => [
        'type' => Item::TYPE_ROLE,
        'description' => 'Moderator',
        'children' => [
            'user',          // Can manage all that user can
            'manageResource1', // and also resource1
        ],
        'bizRule' => NULL,
        'data' => NULL
    ],

    'admin' => [
        'type' => Item::TYPE_ROLE,
        'description' => 'Admin',
        'children' => [
            'moderator',     // can do all the stuff that moderator can
            'manageResource2', // and also manage resource2
        ],
        'bizRule' => NULL,
        'data' => NULL
    ],

    'root' => [
        'type' => Item::TYPE_ROLE,
        'description' => 'Super admin',
        'children' => [
            'admin',         // can do all that admin can
            'manageResource3', // and also manage resource3
        ],
        'bizRule' => NULL,
        'data' => NULL
    ],

];
```

Fig. 5.2 Role hierarchy implementation using PHP code

5.4 Roles and Role assignments

To enable effective assignment and deployment of roles in the model together with the constraints (location and context) bound to the system, we label parameters to handle the assignment of roles and permissions in relation to the constraints. For example, we use parameters as follows to illustrate roles bound by location/context;

doctor_on_duty(doctor_id, patient_id) and doctor (doctor_id, patient_id)

nurse_on_duty(nurse_id, type) and nurse(nurse_id, type)

The above parameters represent a role bound by location and context which the access control decisions depend on and accordingly monitored. *doctor_on_duty(...)* above presents a role of a doctor attending to a certain patient. We can infer that the doctor is on duty (context) and in a hospital/clinic (location) where s/he works. *doctor()* refers to a general doctor *x* who is responsible for treatment of patient *y* but not on duty. The above distinction of roles is very important for the permissions granted or denied the two different roles under a different environment and location. The same applies for the nurse role distinction. Each doctor's context is represented as a binary pair 0 or 1. The 0 is an inactivated role, and the 1 is the activated role. Likewise we give a binary value for the location, 0 being offsite and 1 being onsite or in the hospital/clinic. This role assignment is a database lookup to find whether the person identified by the parameter is on duty or not. Therefore, three elements must be verified for the authentication of a doctor requesting data resources to happen. First, s/he must be a local user (registered in the system), must be an employed practitioner and finally must be on duty or not. Access is then granted depending on what context the request to resources is based on. This access control policy expresses therefore that a person can act in the 'doctor on duty' role as long as he is on duty, has activated the 'local user' role, and has been appointed as an employed practitioner. This assignment of roles is mutually exclusive, meaning no one user can assume both roles at the same time or the same user is not allowed to take both roles.

For role, permission, enforcements and check assignments, we use functions to smoothly execute these tasks. Standard Role Based Access Control functionality gives dynamic easy to handle methods for executing tasks such as role-permission assignments, revocations, and verifying authorizations. The following functions were used to implement the above tasks. See Figure 5.3 below.

```
public function assign($role, $permission)
 {
     return Jf::$Rbac->assign($role, $permission);
 }

 public function check($permission, $user_id)
```

```
    {
        return Jf::$Rbac->check($permission, $user_id);
    }

    public function enforce($permission, $user_id)
    {
        return Jf::$Rbac->enforce($permission, $user_id);
    }
```

Fig.5.3 Code snippet for a RBAC functionality assignment of roles, permissions, check and enforcement

- Function assign is used to assign permissions to roles and returns the permissions assigned to a role.
- Function check is used to check what permissions belong to a user and returns those permissions.
- Function enforce is used to enforce permissions to a user and if the permission enforced is true, it is returned otherwise it is false.

5.5 Administration of RBAC

The implementation of rules in the system is defined by using simple user to role-domain mappings. Each user is assigned a role in a particular domain under a particular context. We also define RBAC administration functions to manage access and/or management of assets in an authorized domain. We use the following functions to enforce these operations;

can_manage_UR(ur (uar, d)) indicates that only a super user (root) in the system can manage the assignment of user roles (uar) in domain d.

has_access_UA(user_id, type) indicates that user with user_id *x* and of type *y* can access resources in that particular domain. Or user _id *x* type *y* has access to resources in that particular domain. As an example, the function below returns the query result of a selected user with user_id and of user_type 1 (which is an administrator in this case) who has access to particular resources.

```
function has_access($user_id, $type)
 {
    $user_id         = (int)$user_id;
    $type            = (int)$type;

    return (mysql_result(mysql_query("SELECT COUNT(`user_id`)
    FROM `users` WHERE   `user_id` = $user_id AND `type` = $type"), 0) ==   1) ?
```

```
        true : false;
    }
```
Fig.5.4: sample function that returns an authorized admin user

Figure 5.5 shows a function that protects user admin page/resources by checking the user's id and type authorization clearance to that particular page. If user's id and type is matched to 1, access is granted else denied. Figure 5.6 on the other hand verifies if a user is authenticated first in order to be granted permission to access a certain resource page. If not the user is denied permission to that particular resource page.

```
function admin_protect() {
     global $user_data;
     if (has_access($user_data['user_id'], 1) === false) {
         header('Location: index.php');
         exit();
   }
}
```
Fig.5.5: a sample function that protects resources authorized to admin users

```
function protect_page() {
    if (logged_in() === false) {
        header('Location: protected.php');
        exit();
    }
}
```
Fig.5.6: sample function authentication verification for a resource page

5.7 Discussion

This chapter presented the implementation of our model design outlined in Chapter 4 and the experiments undertaken to test the security of the model. The main focus of the chapter was looking at how our role based access control system is implemented and tested, as well as the location and context dependencies for authentication and authorization for secure information flow. The role hierarchies and their inheritance give direction of how information flows between or amongst users. A role higher in the hierarchy inherits permissions of roles lower or below their level.

Chapter 6
Results and Evaluation

Evaluation of access and information flow control in mobile web services is an important aspect of this work because, not only do we focus on the security aspect of the work but also include the performance of the model taking into account the limitations brought by the constrained environments. In an attempt to understand the broad view of mobile web service compositions in resource constrained environments taking into considerations location and context dependencies in authentication and authorization for information flow, we classify and evaluate our model using four major dimensions: security of the access control scheme, performance, response time and failure rates. Furthermore, the experiments conducted bordered on evaluating the CIA[5]-triad of the work.

6.1 Experimental Results

As highlighted in Section 5.1, we carried out experiments to study the performance of our prototype model in relation to finding solutions to tasks such as; how much time does password generation takes, how much time does login takes, checking or verification of security and the length of passwords the system employs under different environments and success rates. In this chapter we present and analyze the results and findings of the experiments carried out in Chapter 5 of this thesis. We particularly make analysis on the countermeasures employed, Response Time and Performance of the system.

6.1.1 Experimental setup

As discussed in section 5.2 above, we use a role based access control system in which each data resource is assigned a security class (protection) and each service assigned a clearance level by each domain [103]. The security class measures the security and trust levels and offers protection to the data resources required with regards to who is cleared to access the resources. To qualify the security class levels to the user hierarchy defined in section 5.2 we define multiple security

[5] Confidentiality, Integrity, and Availability

class levels as, No Protection (NP) – Guest, Low Protection (LP) – User, Medium Protection (MP) – Moderator, Medium High Protection (MHP) – Administrator and High Protection (HP) – Root where NP < LP < MP < MHP < HP [103]. Our simple access control policy considers that a request is only granted if the clearance level (from NP to HP) of the requesting service or user is greater than or equal to the security class of the requested resource data. For instance an admin user (MHP) can only be granted access to resources equal or below his security clearance level (from LP to MHP) and not HP. For experimental purposes, we use the administrator user's login credentials to test if access to protected resource pages for guests, normal user and moderator users is granted or not. We also test for the HP security clearance using the administrator's credentials.

We use two role parameters or labels for each user as discussed in Section 5.3 to illustrate location and context dependencies of the access control for secure information flow. For simplicity we give access to a download link to illustrate information flow control based on the user's defined role parameter. Therefore, if access is denied to the download link it means information cannot flow otherwise if granted then the secure flow of information is permissible.

To illustrate changes in security domains we conduct our tests under two Wi-Fi network environments; unsecured and secured networks. We first use a non-secured or open Wi-Fi network environment to test the security of our model on how robust the authentication and authorization mechanisms employed when for instance a moderator user logs in under this environment. We do this test to verify if say, an adversary snooping over the network trying to steal login credentials can be able to gain access to the system with having obtained the right credentials but under different location and context. Tests to evaluate the authentication mechanisms such as correct username and password length and complexity are carried out. We also test how long logging into the system takes, granted or denied under this network environment.

Similarly, we conduct the same test under a secured Wi-Fi network domain. We secure the Wi-Fi network using a Wi-Fi Protected Access II (WPA2) Enterprise security mode with an Advanced Encryption Standard (AES) WPA Algorithm type using a minimum 9 character security key for the password. Therefore, under this network users have to authenticate first to the network by giving the correct username and password (security key) to be connected to the Wi-Fi network.

Moreover, we also test for failure rates and response time (Section 6.1.2) when moving from one network to another. Therefore, a user logs in under a secured network and moves to a non-secured network or vice versa. This we do to check if there will be any interruptions on the service (both in authentication and downloading) caused by the change in domain and if so what they are.

6.1.2 Countermeasures

In order to keep away unwanted users or adversaries to gaining access to the system, we employed countermeasures as presented in Section 3.6 on authentication and authorization by using a robust access control mechanism (RBAC) for our login. The login page is the first entry point to the system, so we used this same page as a first line of defense to attackers by directing users on a compulsory registration step for first time users. Thereafter, users were assigned and grouped to roles and a strict username and password combination bound with a role had to be satisfied to allow users to be logged in.

Experiment: Mimic unauthorized access. This experiment was conducted to test how the system handles confidentiality of users or services' secret or sensitive data, like login credentials and profiles.

Methodology: We mimic unauthorized access to the system by first trying to gain access to the system by supplying or using unregistered user credentials to login. This process was tested by using a combination of five different user credential combinations.

Results: Figure 6.1 gives a snapshot of the results returned each time unknown or invalid login credentials are supplied to the system. This error authentication mechanism acts a first line of defense to resources used or found in the system to prevent illegitimate accesses to resources by unlawful users. Figure 6.2 shows an error message displayed when an unauthorized user tries

Figure 6.1: Error message for unlawful login credentials

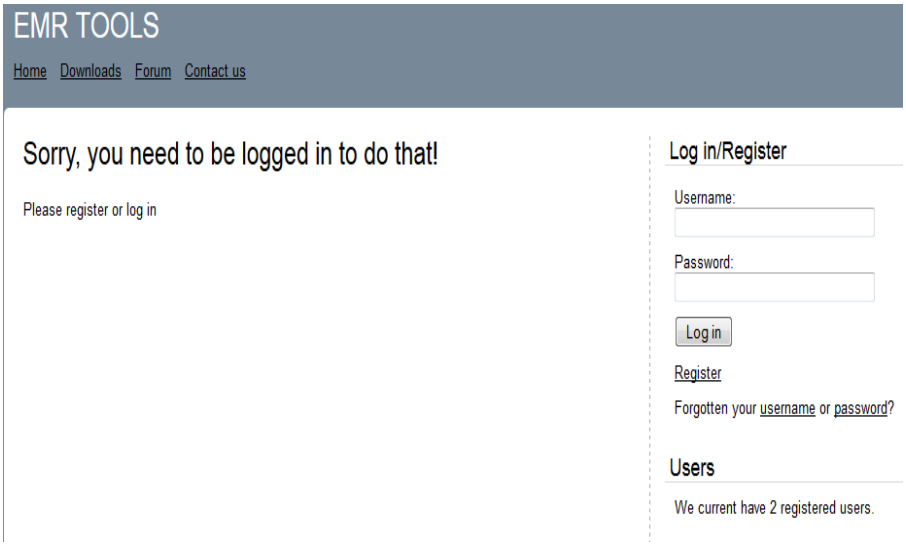

Figure 6.2: Protected assets error message for unauthorized user

to trigger a download from a database. In addition to testing unauthorized accesses to the system and resources, login times had to be verified to test how fast it took authentication to occur. We measured the number of successful logins against the time it took for a successful login to take place. In Figure 6.3 we compare these two elements and show the average time for a successful login occurrence under the two security domains; open network and secured network respectively. The comparison is fundamentally based on the performance of the authentication and authorization mechanism (RBAC system), against the success rate under the different security domains.

Discussion

As can be seen, the results obtained in Fig.6.3 indicate that the login time under an open network took little time compared to the login time success rate under a secured network. Although the

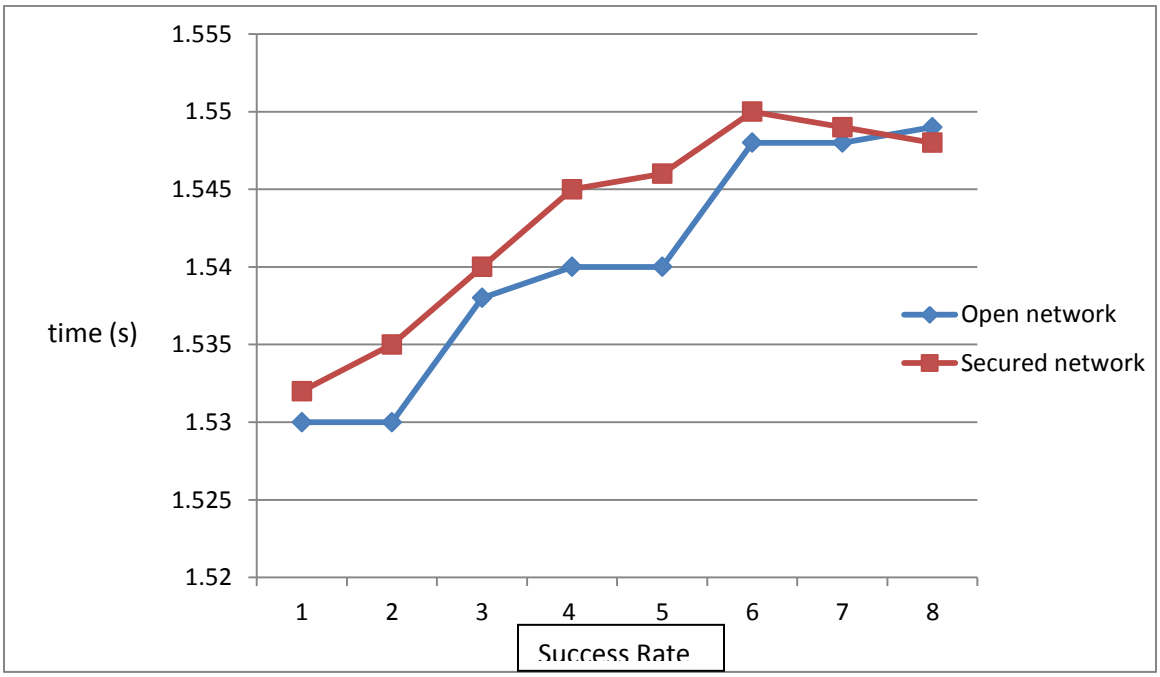

Figure 6.3 Login success rate versus time (s) between the two security domains

difference between the results is very close; the factors that may contribute to this behavior cannot be ignored. Under the secured network, apart from the application authentication taking place the device has to authenticate to the network at the same time. These parallel processes occurring at the same time can contribute to the slight sluggish exchange of requests and responses under this domain. Thus the device may use or consume a bit more resources to execute the logging in session than the whole authentication process taking place under the open or unsecured network domain. Although the success rate is high irrespective of which domain it is carried out, the results doesn't in any way demonstrate a weak authentication and authorization mechanism employed by the system. In fact, the results show that irrespective of which domain a user is, the access control mechanism will be rigid in the same way.

6.1.2.1 Response Time

Experiment: query user profile: Once access is granted to the system, we measured or tested the response time made for queries by taking the number of queries made versus the time it took for a query result to be shown. These queries were made to the backend MySQL database. For easy experimentation, we query a user's profile from the database. Note that the database is a federated database because services are concretized from different databases across multiple domains. Thus the queried service is a composite service. We do not measure the service composition time in this work.

Methodology: We measure the response time by comparing the number of queries made against the time taken for each query response under the two different security domains. For instance, the user's profile is made up of the user's details (gender, age, and color), "allergies", "and chronic illnesses" all from different databases in different domains. Therefore, running such queries under different domains helps display the integrity of the data kept in the federated database or composed service.

We don't measure the time for the service to compose, however, the composing services are the different medical data or information being gathered from different health centers (hospitals and clinics) to form a medical record for an individual patient on the EMR service. Thus we make the assumption that the federated database is already populated or concretized by the composing databases. Therefore we make queries of the composed service to determine inconsistence or consistence to the queries made.

Results: The mean response time taken to generate search query result sets is shown in Figure 6.4. In order to ascertain the overall distribution of the search query response times, the time taken for the search phases under each domain were noted.

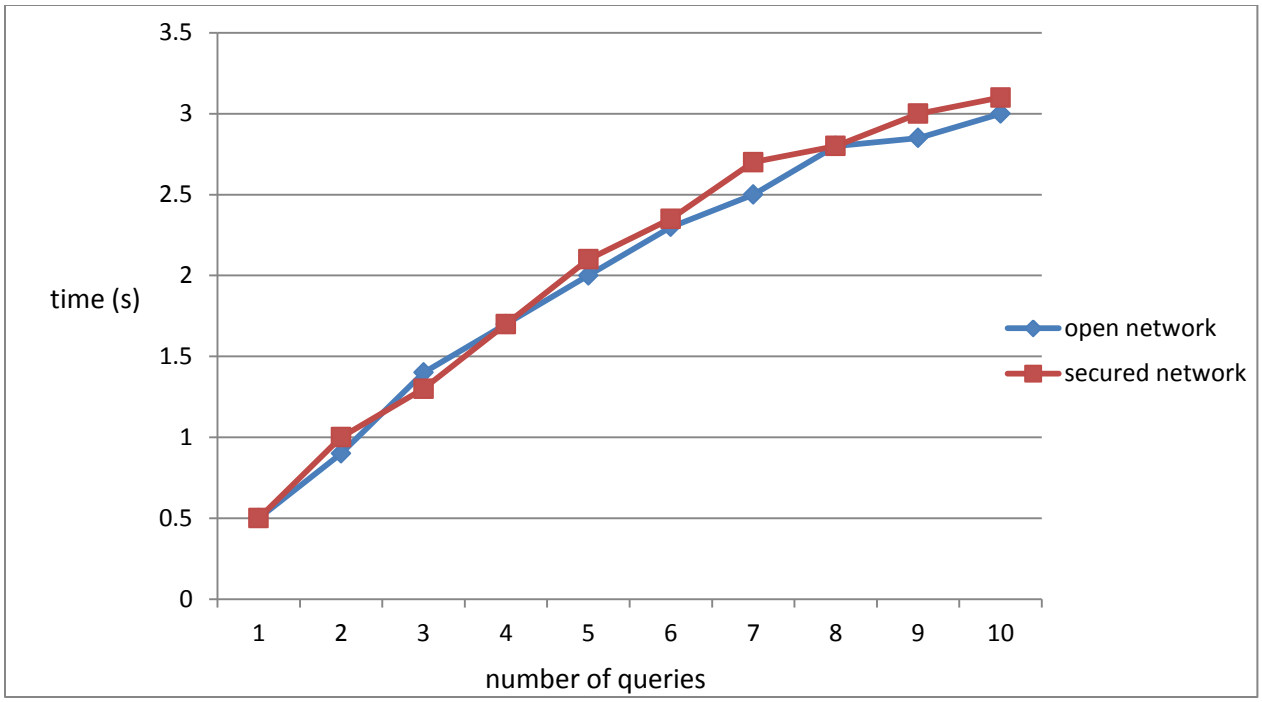

Figure 6.4: Number of queries versus time (s) to respond to a query

Discussion: The results in Figure 6.4 indicate an increase almost linear correlation between the number of queries submitted and the query response time. This is due to the fact that as the workload increases the time for a query result to be returned also increases depending on how large the search query data required is. This linear increase is evident in both tasks under the two different networks. Therefore, this consistent return shows the integrity aspect of the system (that the same size or amount of data queried yields the same result showing no tampering of data size or amount) as expected that when the workload doubles, so does the response time regardless of which security domain the query is carried out from. Moreover, the results confirm that no matter where the query is made from (domain), the same size or slightly smaller/bigger output result will be obtained proving no meddling or interference with the data hence integrity preserved.

6.2.1.2 Performance

The purpose of the following experiment was to determine the impact the number of security policy changes that occur with respect to time to respond has on the performance of the system.

Experiment: number of security changes. This experiment was conducted to ascertain the number of security changes that occur in a given time frame when a service is queried under a specific security domain and moved to a different domain. The aim of this experiment was to test the impact elements like response time and response failure rates have on the performance of the system.

Methodology: In order to get the best results for our test, we conducted this experiment using a cross domain hop. Here a query is started in one domain and then moved to a different domain to check the availability of the system session when this hop occurs. The aim was to determine if a session will be lost or a new session is required for both authentication and authorization for the session to continue so as to determine how robust the access control scheme is when security domain changes or a session is interrupted in terms of requiring the user to re-authenticate or not. Also, we try to determine how long it takes for a particular session to fail or pass during this domain hop or change.

Results: the results of the above experiment are shown in Table 6.1 below.

Table 6.1

Network	No. of policy changes	Time (ms)
Secured	1. Authentication 2. Authorization 3. Role and Permission	25.43 25.6 38.3
Unsecured/Open	0	0

Discussion: The results in Table 6.1 indicate that a significant amount of time is spent with respect to time to respond when policy changes occur from an open network to a secured network domain for each corresponding change. However, the opposite yields a zero time response. This outcome is largely due to the fact that when the change from a secured to an open network takes place, the system and/or device automatically picks up the open network with no form of authentication required to enable or determine a new session taking place. Consequently, when the opposite is true, the system and/or device require some form of authentication to register to the new network (secured) to enable a session to take place. Hence, a number of

policy changes occur during this period since the users and devices entering this (secured) domain have to be legitimate users to allow them to execute their tasks. This change in policies impacts the performance of the system in terms of availability, because as the change in security domain occurs the system stops or terminates whatever session it started in the open network. Therefore, a new session requires authentication and authorization of users to enable them access to resources.

6.2 Analysis of Results

The results of the experiments carried out in this work helped confirm the following issues with regards to performance and security of our prototype or model;

- The model produces a fined grained Role Based Access Control for Information Flow control with location and context dependencies.
- Changes in security domain impact the authentication and authorization policies of the model and the way information is released to requestors.
- A dynamic access control model is relevant for dynamic web environments where policy changes are both instantaneous and ad hoc.

Although this work's interest wasn't on testing service composition protocol success per se, however, its implementation bordered around or was aligned on results and work done by She et.al. on the background. The results obtained in this work are of paramount importance as a breakthrough or first take analysis on binding an access control model, in our case RBAC to constraints like location and context as dependencies for authentication and authorization for secure information flow in dynamic environments.

Moreover, the countermeasures employed for bridging the access control mechanism (RBAC) prove the preservation of confidentiality of users and services' sensitive information. Likewise, the consistent results obtained when measuring integrity confirm that the right path for secure flow of information is upheld or adhered to. Finally, high failure rates experienced when hopping from an unsecured domain to a secured one illustrate that unwarranted users can't be granted access to resources thus proving illegitimate flow of information. Also, this result shows the

unavailability of the system to all users in general which is, though critical it is frustrating to experience.

The threat model presented in Chapter 3 helped to define a set of possible attacks considered for our prototype. These attacks included but not limited to shoulder surfing, brute force attack against the access control scheme, eavesdropping on the network. Results obtained after testing against these attacks provided or proved how robust the security of the model is. For instance, results obtained from performing shoulder attacks, using unauthorized login credentials showed the robustness of the access control scheme with unauthorized logins prevented from occurring. However, aspects like availability were proven to be hard to achieve considering the unavailability of a service when moving from one security domain to another. However, this also allowed for unlawful flows of information between services not to be realized as a result because, as soon as the service is unavailable, the session was simply terminated with no request or responses taking place.

Chapter 7
Conclusions

This thesis has examined security guarantees in securing mobile web services using access and information flow control. Specifically, using RBAC, location and context dependencies for both authentication and authorization for information flow control as outlined in Chapter 4. Furthermore, Chapter 5 defines the methods used for implementation of the model used in guaranteeing security for web services in the mobile sphere. The results in Chapter 6 provides a good measure of the security or arguably how our model prototype secures or takes care of the CIA-triad in mobile web service environments. Above all, we tested aspects like response time, both on performance of the system in totality and individual elements such as authentication and authorization mechanisms.

7.1 Summary

The results helped us address or confirm that in dynamically changing security environments or domains, different security policies come to play with different actors or players, however, all participating players (services) must agree on basic security policies/terms in order to compose a secure solution for a service requestor. On the other hand, the results showed that when the participating services don't agree on the basic security policies in order to compose, the session is dropped indefinitely because of untrusted services wanting to compromise the desired composed service. Consequently, the user/service's context and location is taken in account when processing the said request. This means, access and manipulations to resources is given only when the right context and user's whereabouts are satisfied or fulfilled as shown in results obtained in Chapter 6.

Security assurance is critical to the success of information sharing in any environment. Our formal RBAC model bound by location and context dependencies for authentication and authorization for information flow control is intended to address this critical aspect of information sharing in dynamically changing security environments. Our research and concept prototype shows that it is possible to provide a security framework for mobile or resource

constrained environments with flexible, dynamic service composition characteristics. Our security model is an improvement over other models in that, first; the service or user's surroundings and physical location are considered or bound with the user's role in making authentication and authorization decisions to access and manipulate resources. Second, information flow decisions are based on a concept previously used in program statements, Program Dependence Graphs which features path conditions for information to be allowed to propagate from one service to the next. This concept is a first of its kind to be used in a mobile web service platform other than in type systems. A third feature for this security model is our usage of a hierarchical structure to govern access to resources, although this method is not a novelty, its usage helps tightens the security of our model. Therefore, our approach provides a fine and simple secure access control scheme which gives access to resources by allowing multiple roles, which uses the same resources in different environments, to have different invocation of service constraints without violating the minimum security requirements set.

The relevance of this work can be applied directly to Electronic Health Record systems or any other information sharing applications in mobile web services or over the Internet where executing necessary tasks requires interaction between systems and shared databases with location and context dependencies for access to resources regarded as necessary constraints. This work has demonstrated success in this area by way of a potential use of a system that can be used in a federation or coalition, where the need to dynamically combine users and resources while at the same time maintaining information assurance is critical. Explicitly, we have met the goals of:

- Binding RBAC to location and context. The objective here was to use or develop a RBAC model that uses location and context reliance for authentication and authorization for secure information flow in mobile web environments particularly using an EHR system.
- Establishing or using PDGs for information flow control. Our objective here was to use PDGs with path conditions to allow for secure information flow between services and users. This was accomplished by our security prototype or framework design. We were successful in using this approach to build a very fine-grained information flow between services by user context. Moreover, path conditions are critical in ensuring unlawful or unwanted flow and exchange of information between users and services.

Finally, this work proves to be a good and valid presentation of a model that can be adopted in securing mobile service compositions in resource constrained environments because security techniques like confidentiality, integrity and availability are addressed, which are critical aspects of any security aware service.

7.2 Future Work

As discussed in Chapter 2, there are ongoing efforts dedicated to secure web services by using both access control and information flow control in the same context, [102, 103] but using different approaches or methods. Breakthrough work by [103] illustrates the need to have such systems in place in different security domains. Following the path or solution laid down by this work, our future work would like to find solutions to scale down models for service compositions to handle mobile service environments without putting heavy resource demands in overhead computation power and bandwidth in mobile web service domains. This includes easy and quick ways to compose services, and easy but robust access control schemes for this environment. Ideally, the results of this work offers an opportunity to explore other access control schemes (like attribute based access control) that can be explored for mobile environments with location and context dependencies in authentication and authorizations for information flow control in resource constrained environments.

This thesis has provided a new model and techniques for providing security for mobile web services in resource constrained environments. Providing better security for mobile web services in these environments is a challenging and important problem for future computing environments. Moreover, these environments are likely to be large and distributed and to contain untrusted users, services and domains. This problem has not received as much attention as it deserves, and I hope the contributions of this work will serve as a fresh motivation to its further consideration.

References

[1] American National Standard Institute, "ANSI INCITS 359-2004 for Role Based Access Control", 2004.

[2] A. Askarov, S. Hunt, A. Sabelfeld, and D. Sands. Termination-insensitive noninterference leaks more than just a bit. In *Proc. European Symp. on Research in Computer Security*, volume 5283 of *LNCS,* pages 333–348. Springer-Verlag, October 2008.

[3] A. Banerjee and D. A. Naumann. Secure information flow and pointer confinement in a Java-like language. In *Proc. IEEE Computer Security Foundations Workshop*, pages 253–267, June 2002.

[4] A. Banerjee and D. A. Naumann. History-based Access Control and Secure Information Flow, 2005.

[5] A. C. Myers. "Mostly –Static Decentralized Information Flow Control". PhD Thesis, Massachusetts Institute of Technology, February 1999.

[6] A. Kumar, N. Karnik, G. Chafle, "Context sensitivity in role-based access control", *ACM SIGOPS Operating Systems Review,* Volume 36 Issue 3, July 2002.

[7] A. Russo, A. Sabelfeld, and A. Chudnov. Tracking information flow in dynamic tree structures. In *Proc. European Symp. on Research in Computer Security, LNCS*. Springer-Verlag, September 2009.

[8] A. Sabelfeld. The impact of synchronization on secure information flow in concurrent programs. In *Proc. Andrei Ershov International Conference on Perspectives of System Informatics*, volume 2244 of *LNCS*, pages 225–239. Springer-Verlag, July 2001.

[9] A. Sabelfeld and A. C. Myers. Language-based information-flow security. *IEEE J. Selected Areas in Communications*, 21(1):5–19, January 2003.

[10] A. Sabelfeld and D. Sands. Probabilistic noninterference for multi-threaded programs. In *Proceedings of the 13th IEEE Computer Security Foundations Workshop*, pages 200-214, Cambridge, England, July 2000. IEEE Computer Society Press.

[11] A. Sabelfeld and H. Mantel. Static confidentiality enforcement for distributed programs. In *Proc.Symp. on Static Analysis*, volume 2477 of *LNCS*, pages 376–394. Springer-Verlag, September 2002.

[12] A. Stoughton. Access flow: A protection model which integrates access control and information flow. In *Proc. IEEE Symp. on Security and Privacy*, pp. 9–18, Oakland, CA, 1981.

[13] A.V.D.M Kayem. "Adaptive Cryptographic Access Control for Dynamic Data Sharing Environments", PhD Thesis, Queen's University, Kingston, Ontario, Canada, October 2008.

[14] A.V.D.M. Kayem, P. Martin, and S.G. Akl. Heuristics for improving cryptographic key assignment in a hierarchy. In *Proceedings, 3rd IEEE Int'l Symposium on Security in Networks and Distributed Systems (Niagara Falls, Canada)*, pages 531–536, May 21-23, 2007.

[15] A.V.D.M Kayem, P. Martin, S.G. Akl, and W. Powley. A framework for self-protecting cryptographic key management. In *Proceedings, 2nd IEEE International Conference on Self-Adaptive and Self-Organizing Systems,* Venice, Italy, Oct 20-24, 2008.

[16] A.V.D.M Kayem, S.G. Akl, and P. Martin. On replacing cryptographic keys in hierarchical key management systems. *Journal of Computer Security*, 16(3):289–309, 2008.

[17] A. W. Roscoe, "CSP and determinism in security modeling", in *Proc. IEEE Symp. on Security*

and Privacy, pp.114-127, May 1995.

[18] B. Benatallah, Q. Z. Sheng, and M. Dumas. The Self-Serv Environment for Web Services Composition. IEEE Internet Computing, 7(1), January/February 2003.

[19] B. Carminati, E. Ferrari, and P.C.K Hung. Exploring Privacy Issues in Web Services Discovery Agencies. *IEEE Security & Privacy Magazine*, 3(5): 14-21, 2005.

[20] B. Carminati, E. Ferrari, and P.C.K Hung. "Security Conscious Web Service Composition", ICWS pp. 489-496, 2006.

[21] B. Carminati, E. Ferrari, and P. C. K. Hung, "Web service composition: a security perspective". *Proceedings of the 2005 International Workshop on challenges in Web Information Retrieval and Integration,* pp. 248-253. 2005.

[22] B. Shafiq, J. Joshi, E. Bertino, and A. Ghafoor, "Secure interoperation in a multi-domain environment employing RBAC policies", *IEEE Transactions on Knowledge and Date Engineering,* 17(11):1557-1577, November 2005.

[23] C.A. Ardagna, S.D.C.D. Vimercati, S. Paraboschi, E. Pedrini, P. Samarati, and M. Verdicchio, "Expressive and Deployable Access Control in Open Web Service Applications," *IEEE Trans. Services Computing*, vol. 4, no. 2, pp. 96-109, Apr.-June 2011.

[24] C. Ardagna, M. Cremonini, E. Damiani, S. De Capitani di Vimercati, and P. Samarati,: Supporting location-based conditions in access control policies. In: *Proc. of the ACM Symposium on Information, Computer and Communications Security* (ASIACCS'06), Taipei, Taiwan (March 2006)

[25] C. Ardagna, M. Cremonini, E. Damiani, S. De Capitani di Vimercati, and P. Samarati, : Location-

based metadata and negotiation protocols for LBAC in a one to- many scenario. In: *Proc. of the Workshop on Security and Privacy in Mobile and Wireless Networking* (SecPri MobiWi 2006), Coimbra, Portugal (May 2006)

[26] C. Ardagna, M. Cremonini, E. Damiani, S. De Capitani di Vimercati, and P. Samarati,: Privacy enhanced location-based access control. In Gertz, M., Jajodia, S., eds.: The Handbook of Database Security: Applications and Trends. Springer-Verlag (2007)

[27] C. Ardagna, M. Cremonini, E. Damiani, S. di Vimercati, and P. Samarati, "Supporting location-based conditions in access control policies," in *Proceedings of the 2006 ACM Symposium on Information, computer and communications security*. ACM, 2006, p. 222.

[28] C. Hammer, "Information Flow Control for Java: A Comprehensive Approach Based on Path Conditions in Dependence Graphs", PhD Thesis, Karlsruhe University, ISBN: 078-3-86644-398-3; 2009.

[29] C. Hammer, J. Krinke, and G. Snelting. Information flow control for Java based on path conditions in dependence graphs. In *Proc. IEEE International Symposium on Secure Software Engineering (ISSSE '06),* pages 87-96, March 2006.

[30] D. Bell and L. LaPadula. Secure computer systems: Mathematical foundations and model. *MITRE Report*, MTR2547, page 2, 1973.

[31] Chakraborty, D., Joshi, A., Finin, T., Yesha, Y.: Service Composition for Mobile Environments. Journal on Mobile Networking and Applications, Special Issue on Mobile Services (2005) 435-451

[32] D. Denning and P. MacDoran, "Location-based authentication: Grounding cyberspace for better

security," Computer Fraud & Security, vol. 1996, no. 2, pp. 12–16, 1996.

[33] D. E. Denning. A lattice model of secure information flow. *Comm. of the ACM*, 19(5):236–243, May 1976.

[34] D. E. Denning. Secure Information Flow in Computer Systems. PhD thesis, Purdue University, W. Lafayette, Indiana, USA, May 1975.

[35] D. E. Denning and P. J. Denning. Certification of programs for secure information flow. *Comm. of the ACM*, 20(7):504–513, July 1977.

[36] D. Hedin and A. Sabelfeld. A Perspective on Information-Flow Control. *Technical Report*, Chalmers University of Technology, Gothenburg, Sweden.

[37] D. Gollmann. Computer Security. John Wiley and Sons, Ltd, 2005.

[38] D. Volpano, G. Smith, and C. Irvine. A sound type system for secure flow analysis. *Journal of Computer Security*, 4(3):167–187, 1996.

[39] E. Bertino, A.C. Squicciarini, L. Martino, and F. Paci, "An Adaptive Access Control Model for Web Services," *Int'l J. Web Services Research*, vol. 3, no. 3, pp. 27-60, 2006.

[40] E. Poblenz. Language-Based Approaches to Secure Information Flow. 2004.

[41] E. Yuan and J. Tong, "Attributed Based Access Control (ABAC) for Web Services," *Proc. IEEE Int'l Conf. Web Services*, pp. 561-569, 2005.

[42] E. Sirin, J.A. Hendler, and B. Parsia,: Semi-automatic composition of web services using semantic descriptions. In: 1st Workshop on Web Services: Modeling, Architecture and Infrastructure. (2003) 17–24.

[43] F. Paci, M. Mecella, M. Ouzzani, and E. Bertino, "ACCONV—An Access Control Model for Conversational Web Services," *ACM Trans. Web*, vol. 5, no. 3, article 13, 2011.

[44] G. Smith. A new type system for secure information flow. 2001.

[45] G. Smith and D. Volpano. Secure information flow in a multi-threaded imperative language. In *Proc. ACM Symp. on Principles of Programming Languages*, pages 355–364, January 1998.

[46] H. Mantel. Information flow control and applications—Bridging a gap. In *Proc. Formal Methods Europe,* volume 2021 of *LNCS*, pages 153–172. Springer-Verlag, Mar 2001.

[47] H. Mantel. On the composition of secure systems. In *Proc. IEEE Symp. on Security and Privacy*, pages 81–94, May 2002.

[48] H. Mantel and A. Sabelfeld. A generic approach to the security of multi-threaded programs. In *Proc. IEEE Computer Security Foundations Workshop*, pages 126–142, June 2001.

[49] https://msdn.microsoft.com/en-us/library/ee82387(v=cs.20).aspx

[50] I. F. Cruz, R. Gjomemo, B. Lin, and M. Orsini, "A Constraint and Attribute Based Security Framework for Dynamic Role Assignment in Collaborative Environments", in *Collaborative Computing: Networking, Applications and Worksharing*, Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol. 10, 322-339

(2009). http://dx.doi.org/10.1007/978-3-642-03354-4_24

[51] I. Ray, M. Kumar, and L. Yu, "LRBAC: A Location-Aware Role-Based Access Control Model," Lecture Notes in Computer Science, vol. 4332, p. 147, 2006.

[52] I. Ray and L. Yu, "Short paper: Towards a location-aware role-based access control model," in Proceedings of the First International Conference on Security and Privacy for Emerging Areas in Communications Networks. IEEE Computer Society, 2005, pp. 234–236.

[53] I. Ray and N. Narasimhamurthi. A cryptographic solution to implement access control in a hierarchy and more. In *Proceedings of the 7th ACM Symposium on Access Control Models and Technologies*, Monterey, CA, pages 65–73, 2002.

[54] J. A. Goguen and J. Meseguer. Security policies and security models. In *IEEE Symposium on Security and Privacy*, pages 11–20, 1982.

[55] J. A. Goguen and J. Meseguer. Security policies and security models. In *Proc. IEEE Symp. On Security and Privacy*, pages 11–20, April 1982.

[56] J. Banatre, C. Bryce, and D. Le M'etayer. Compile-time detection of information flow in sequential programs, 1994.

[57] J. Han, R. Kowalczyk, and K.M. Khan. "Security-Oriented Service Composition and Evolution", ASPEC, pp. 71-78, 2006.

[58] J. Joshi, R. Bhatti, E. Bertino, and A. Ghafoor, "Access control language for multi-domain environments", *IEEE Internet Computing*, pages 40 - 50, November - December 2004.

[59] J. Zhu, Y. Zhou, and W. Tong, "Access Control on the Composition of Web Services,"

Proc. Int'l Conf. Next Generation Web Services Practices, pp. 89-93, 2006.

[60] K. Biba. Integrity considerations for secure computer systems. *Technical Report ESD-TR-76-372, ESD/AFSC*, Hanscom AFB, Bedford, MA, April, 1977.

[61] L.D. Alfaro and T.A. Henzinger,: Interface automata. In: 8th European software engineering conference 9th ACM SIGSOFT international symposium on Foundations of software engineering. (2001) 109–120

[62] L. Kagal, T. Finin, and A. Joshi. A Policy Based Approach to Security on the Semantic Web, in *Proceedings of the second International Semantic Web Conference (ISWC)*, Florida, USA, 2003.

[63] L. Liu. On secure Flow Analysis in Computer systems. In *Proc. IEEE Symposium on Research in Security and Privacy*, pages 22-33. 1980.

[64] L. Zeng, B. Benatallah, M. Dumas, J. Kalagnanam, and Q. Z. Sheng. Quality Driven Web Service Composition. In Proceedings of The Twelfth International World Wide Web Conference (WWW'2003), Budapest, Hungary, 2003.

[65] M. Bartoletti, P. Degano, and G.L. Ferrari. Plans for service composition. In *Proceedings of the Workshop on Issues in the Theory of Security (WITS),* Vienna, Austria, 2006.

[66] M. Decker, "Requirements for a location-based access control model," in Proceedings of the 6th International Conference on Advances in Mobile Computing and Multimedia. ACM, 2008, pp. 346–349.

[67] M. Duckham, L. Kulik,: Location privacy and location-aware computing. In: Dynamic & Mobile GIS: Investigating Change in Space and Time. Taylor & Francis (2006) 34–51

[68] M. J. Covington, W. Long, S. Srinivasan, A. K. Dev, M. Ahamad, G. D. Abowd, "Securing context-aware applications using environment roles", *Proceedings of the sixth ACM symposium on Access control models and technologies,* May 2001.

[69] M. Pistoin, A. Banerjee, and D. A. Naumann. Beyond stack inspection: A unified access-control and information-flow security model. In *SP'07: Proceedings of the 2007 IEEE Symposium on Security and Privacy,* pages 149-163, Washington, DC, USA, 2007. IEEE Computer Society.

[70] M. Srivatsa, A. Iyengar, T. Mikalsen, I. Rouvellou, and J. Yin, "An Access Control System for Web Service Compositions," *Proc. IEEE Int'l Conf. Web Services*, pp. 1-8, 2007.

[71] M-A. Jeong, J-J. Kim, and Y. Won. A flexible database security system using multiple access control policies. In *Proceedings. 4th International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT' 2003),* pages 236–240, 2003.

[72] N. Luo, J. Yan, M. Liu, and Shuxin Yang, "Towards Context-Aware Composition of Web Services," in *Proceedings of the Fifth International Conference on Grid and Cooperative Computing (GCC'06)*.

[73] O. Lassila, and S. Dixit: Interleaving discovery and composition for simple workfows. In: First International Semantic Web Services Symposium. (2004)

[74] Open Web Application Security Project (2002) J2EE and .NET Security. http://www.owasp.org/downloads/J2EEandDotNetsecurityByGerMulcahy.pdf.

[75] P. Laud. Semantics and program analysis of computationally secure information flow. In *Proceedings of the 10th European Symposium on Programming Languages and Systems*, pages 77–91. Springer-Verlag, 2001.

[76] P. Li and S. Zdancewic. Unifying confidentiality and integrity in downgrading policies. In *Foundations of Computer Security Workshop (FCS)*, 2005.

[77] P. Samarati, "Access Control: Policies, Models, Architectures, and Mechanisms," FOSAD, Italy, Sept. 2000.

[78] R. Bhatti, E. Bertino, and A. Ghafoor, "A trust-based context-aware access control model for web-services," Distributed and Parallel Databases, vol. 18, no. 1, pp. 83–105, 2005.

[79] R. Focardi and S. Rossi. Information flow security in dynamic contexts, 2002.

[80] R. Sandhu. Cryptographic implementation of tree hierarchy for access control. *Information Processing Letters*, 27:1–100, January 1988.

[81] R. S. Sandhu. Role-based access control. In M. Zerkowitz, editor*, Advances in Computers*, volume 48. Academic Press, 1998.

[82] R. S. Sandhu. Role Hierarchies and Constraints for Lattice-Based Access Controls. *Proc. Fourth European Symposium on Research in Computer Security*. Rome, Italy. 1996.

[83] R. S. Sandhu. Lattice-Based Access Control Models. *IEEE Computer*, 26(11):9-19. 1993.

[84] R. Zhang, I.B. Arpinar, and B. Aleman-Meza: Automatic composition of semantic web services. In: *1st International Conference on Web Services*. (2003) 38–41

[85] S. Chandran and J. Joshi, "LoT-RBAC: A location and time-based RBAC model," Web Information Systems Engineering–WISE 2005, pp. 361– 375, 2005.

[86] Security Assertions Mark-Up Language (SAML) OASIS. XML-Based Security Services Technical Committee. http://www.oasis-open.org/committees/security/

[87] S.G. Akl and P.D. Taylor. Cryptographic solution to a problem of access control in a hierarchy. *ACM Transactions on Computer Systems*, 1(3):239–248, August 1983.

[88] S. Osborn. Integrating role graphs: A tool for security integration. Data and Knowledge Engineering, 43:317–333, 2002.

[89] S. Osborn. Mandatory Access Control and Role-Based Access Control Revisited. *Proceedings of the second ACM workshop on Role-based access control*, Pages: 31-40. Fairfax, Virginia, USA, 1997.

[90] S. Osborn and Y. Guo. Modeling Users in Role-Based Access Control.

[91] S. Rossi and D. Macedonio. Information Flow Security for Service Compositions. *Technical report*, Universita Ca Foscari, Dipartimento di Informatica, via Torino 155, 30172 Venice, Italy, 2009.

[92] S. Piromruen and J. Joshi, "RBAC framework for time constrained secure interoperation in multi-domain environments", In *Proceedings of 10th IEEE International Workshop on Object-ORIENTED Real-Time Dependable Systems (WORDS' 05)*, pages 36-48, 2005.

[93] S. Oh, R. Sandhu, and X. Zhang, "An Effective Role Administration Model Using Organization Structure", *ACM Transactions on Information and System Security*, Volume 9, Number 2, May 2006, pages 113-137.

[94] T. Amtoft and A. Banerjee. Information flow analysis in logical form. *Technical report*,

George Mason University, 2004.

[95] T. Pobschink and G. Snelting. Efficient path conditions in dependence graphs. In *ICSE'02: Proceedings of the 24th International Conference on Software Engineering*, pages 478-488, New York, NY, USA, 2002, ACM Press.

[96] T.Y. Lin. Managing information flows on discretionary access control models. *2006 IEEE International Conference on Systems, Man, and Cybernetics*, pages 4759–4762, Oct. 8 – 11, 2006.

[97] U. Yildiz and C. Godart, "Information Flow Control with Decentralized Service Compositions," *Proc. IEEE Int'l Conf. Web Services*, pp. 9-17, 2007.

[98] V. Machiraju, G. Alonso and F. Kuno; Web services: concepts, architectures and applications: Springer-Verlag (2004).

[99] V. Ramesh, R. L. Glass and I. Vessey, "Research in computer science: an empirical study", *Journal of Systems and Software*, Vol. 70, Pages 165-176, Feb. 2004.

[100] Web-Services-Axis. http://ws.apache.org/axis/

[101] W. She, I. Yen, B. Thuraisingham, and E. Bertino, "Effective and Efficient Implementation of an Information Flow Control Protocol for Service Composition," *Proc. IEEE Int'l Conf. Service-Oriented Computing and Applications,* pp.1-8, 2009

[102] W. She, I. L. Yen, and B. Thuraisingham. Policy Driven Service Composition with Information Flow Control. *IEEE International Conference on Web Services*, 2010.

[103] W. She, I. L. Yen, B. Thuraisingham, and E. Bertino. Security-Aware Service Composition with Fine Grained Information Flow Control. *IEEE Transactions on Services Computing*, vol.6, No.3,

July-September 2013.

[104] Y. Jadeja, K. Modi and A. Goswami. "Context Based Dynamic Web Services Composition Approaches: a Comparative Study". *International Journal of Information and Education Technology*, Vol. 2, No. 2, April 2012.

[105] Z.M. Mao, R.H. Katz, and E.A. Brewer,: Fault-tolerant, scalable, wide-area internet service composition. Technical Report CSD-01-1129, University of California at Berkeley (2001)

[106] Z. Zhang, X. Zhang and R. Sandhu, "ROBAC: Scalable Role and Organization Based Access Control Models", *Proceedings of CollaborateCom -2006/TrustCol – 2006*, Atlanta, Georgia, USA, Nov. 06.

[107] Z. Zhang, X. Zhang, and R. Sandhu, "Towards a Scalable Role and Organization Based Access Control Model with Decentralized Security Administration", in Manish Gupta and Raj Sharman edit: "Handbook of Research on Social and Organizational Liabilities in Information Security", IGI Global publications. Accepted for publishing in April 2007.