

Decomposing ABAC into Explicit Combination of DAC and Attribute Constraints

Submitted in partial fulfillment of the requirements of the degree of

Master of Technology (M.Tech)

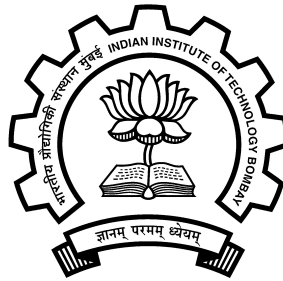
by

Sejal Patel

Roll no. 163050093

Supervisor:

Prof. R.K. Shyamasundar



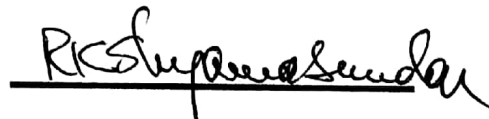
Department of Computer Science & Engineering

Indian Institute of Technology Bombay

2018

Dissertation Approval

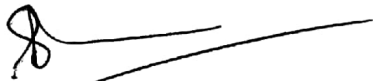
This project report entitled "Decomposing ABAC into Explicit Combination of DAC and Attribute Constraints", submitted by Sejal Patel (Roll No. 163050093), is approved for the award of degree of Master of Technology (M.Tech) in Computer Science & Engineering.



Prof. R.K. Shyamasundar

Dept. of CSE, IIT Bombay

Supervisor



(Prof. Sachin Patkar)

Dept. of EE, IIT Bombay

External and Internal Examiner



(VIRENDRA SINGH)

Dept. of CSE, IIT Bombay

Internal Examiner

Date: 3 July 2018

Place: IIT Bombay

Declaration of Authorship

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Signature: 

Sejal Patel
163050093

Date: **3**... July 2018

Abstract

Due to prominent progress of Internet and web technologies, information and data sharing has been promoted. Access Control is the technique to give the authority for accessing specific resources, system or applications. The traditional access control models are MAC model, DAC model and RBAC model. Attribute based access control is technique to decide whether access can be given or not to requesting users by checking access policies which are defined by merging related attributes (of object and subject) to manage access rights given to users. ABAC provides expressive power as it integrates access control with variations of constraints on attributes of subject and objects. This dissertation provide a framework for access control using DAC, MAC and RBAC that would mimic ABAC. Using appropriate attributes we can represent access control lists and identities, clearances, security labels, classifications and roles. We can use attributes to realize the underlying DAC, MAC, RBAC in ABAC security model. A decision table is basic way to represent an ABAC rule is using table which represents the relationships between attributes of the subject and object. we decomposed that table into explicit combination of DAC and attribute constraints. In the dissertation, we show how formal model of ABAC using decision table can be realized by integrating information flow security models with constraints on attributes.

Contents

Abstract	i
1 Introduction	1
2 Background and Related Work	3
2.1 Access control mechanism	3
2.2 Terminology	5
2.3 Early access models	6
2.3.1 Discretionary Access Control	6
2.3.2 Mandatory Access Control	9
2.3.3 Role Based Access Control	12
2.4 Related Work	15
2.4.1 Attributed Role Based Access Control	15
2.4.2 Mandatory Access Control Model with Enhanced Flexibility	16
2.4.3 Mandatory Access Control in RBAC	18
2.4.4 Compatible and Usable Mandatory Access Control	19
3 Attribute based access control	21
3.1 Introduction	21
3.2 Specification Of ABAC	23
3.2.1 Basic concepts	23
3.2.2 Decision table method	25
3.2.3 XACML	26
3.3 Comparison OF ABAC Specification Languages	27

4	Decomposition of Decision table	29
4.1	Specification Of ABAC	29
4.2	Authorization Policy	30
4.3	Algorithm To separate Tables	32
4.4	Our Approach	34
4.5	Some issues due to separation of the table	34
4.5.1	How separation of tables still satisfy all the rules mentioned in policy set?	34
4.5.2	How can we make the system consistent?	35
4.5.3	How DAC, attribute based constraints, role based constraint are satisfied?	35
5	Conclusion	37
	Acknowledgements	40

Chapter 1

Introduction

As unauthorized access might cause data loss or can misuse the data, so there is a concern for privacy and integrity of data. Unauthorized access can be dangerous for financial software. It makes an access control mechanism more essential for secure access to the software as well as hardware resources. Access control is used in vivid areas like operating system, file management, software resources, internet of things. Access control is the technique to give the authority for accessing specific resources, system or applications. There exists many access control models such as MAC model, DAC model and RBAC model. To decide which access model is most suitable, we have surveyed various traditional security access control methods and ABAC. DAC is mechanism that is based upon the identity of and/or groups while MAC relies on sensitivity labels. Both models can not satisfy all the requirements needed in an ideal access control model. DAC is not so strict to restrict privileges propagation and MAC is not flexible. RBAC provides high security and manages permission easily. It is known due to its robustness as well as ease of administration for access rights. But it has some drawbacks such as role structure complexity. Attribute based access control is technique to decide if access can be given or not to requesting users by checking access policies which are defined by combining related attributes (of object and subject) to manage access to users. Using appropriate attributes we can represent access control lists and identities, clearances, security labels, classifications and roles. Therefore, we can use these attributes and realize DAC, MAC, RBAC in the framework of ABAC. We have implemented ABAC using the decision table method. A decision table is a basic way to represent an ABAC rule is using table which

represents the relationships between attributes of the subject and object. We decomposed that table into explicit combination of DAC and attribute constraints. We have separate this table in terms of subjects' attributes, resources' attributes and DAC table. So, we can realize DAC in the framework of ABAC. Even after separating the table, it will satisfy all the rules mentioned in the given policy set. In the dissertation, we show how formal model of ABAC using decision table can be realized by integrating information flow security models with constraints on attributes.

Chapter 2

Background and Related Work

2.1 Access control mechanism

Due to prominent progress of internet and web technologies, information and data sharing has been promoted. As unauthorized access might cause data loss or can misuse the data, so there is a concern for privacy and integrity of data. Unauthorized access can be dangerous for financial software. It makes an access control mechanism more essential for secure access to the software as well as hardware resources [10].

Access control is a mechanism which gives us a way to provide a specific restriction on data access to legitimate users only. Major components of an access control model are identification, authentication (verification against established policies) and authorization. It allows subjects to identify themselves as authorized users, using their credential and give access to the data resources. A simple example is Linux systems, where users are given access to the files based on an already existing ACL i.e., Access Control List[9], which is predefined by the root administrator. It gives the authority for accessing specific resources, system or applications. Access control contains a set of rules or criteria for accessing resources of the system. The traditional access control models are MAC model, DAC model and RBAC model[6].

Access control is used in vivid areas like operating system, file management, software resources, Internet of Things. The most significant area is the Internet of Things. As IOT involves devices connection and things over the Internet. As this connection is continuous,

information are exchanged/transferred continuously we have to protect the information from unauthorized access and keep it safe and secure. In order to provide secure access and to maintain safety and privacy of data or information these possible rules are specified in policy set and those rules are constructed by trusted parties and these policies must be verified before giving access rights to the information.

An example which explains the application of access control mechanism is EHR. The access control techniques are needed in hospital management system where sensitive information of patient should be accessible only to the authorized users i.e doctor assigned to the patient to maintain the privacy of patient data. In emergency situations when a doctor needs the data of patient to be treated, access control techniques can provide regulated access to the patient's data. Access control mechanism can also be used in other application such as automotive technologies, smart home, and library management. Smart home implementation involves devices which are connected with each other to provide execution of predefined tasks.

It is important to authorize the users/owners who will have access rights to these internal devices and can handle their functionality to ensure safety and privacy of authorized users. If safe home is not implemented with care it can suffer from security risks which hamper the confidentiality, privacy and security of users' private information.

Figure 2.1 represents hospital management scenario. In this patient's data are recorded in central database. Patient records are protected from unauthorized access by implementing access policy set which allows only legitimate doctors, practitioners and patients to have access rights to the appropriate records to ensure privacy and security of patient's data [9].

Hence, we can say that access control models decide whether subjects can be given access to the objects or not and how that this access is possible. Access control model is a technique for implementation of security policies and to ensure safety, privacy and the integrity of the data. It also manages how information or data could be accessed and exchanged or shared on the system[3].

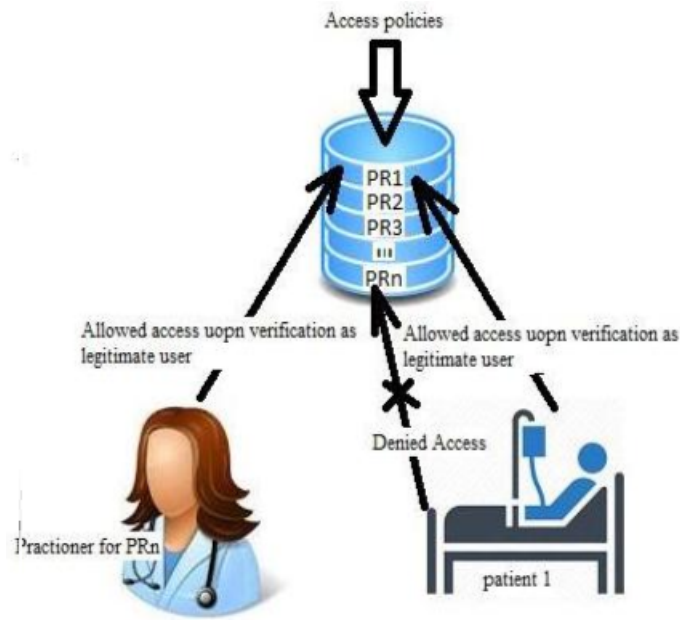


Figure 2.1: Access control management in Hospital System

2.2 Terminology

Subjects are entities which cause information flow between objects. Subjects are entities which manipulate data for their need. Generally subjects are persons/users, but sometimes processes (user process or system process), devices, networks can also be subjects too. Hence, when a user performs some activity on resources it becomes a subject, a process being run which is scheduled by the user to run later is also a subject, and devices like a power sensor that take backup when power failures occur can also be a subject. If users log on to their account interactive systems and start a process, the process works in place of that user who initiated it. The process will take user's attributes, like access rights, and that's how the process related to that user now becomes the subject in the system. Usually users are responsible for the actions started by them. Sometimes, subjects can also be objects. Like, if any process spawns a child process, this process can also be treated as an object.

Objects are entities which contain or receive data/information [7]. We can say that objects are the entities through which information can flow when a subject initiates an action. Examples of objects are file, directory, keyboard, screen, storage, printer[2].

Objects may include records, pages, blocks, segments, directories, files, directory trees. If the systems contains the smallest amount of data as a file, the each file can be considered as an object. But what if each file is stored as smaller pieces and each of that smaller piece of file can be manipulated individually, then these smaller pieces can be considered as an object. And if these files are arranged to give a tree structure, in that case the directories of those files are considered as an object[7].

Security policies are set of rules which defines who should be given authorized access on objects and under what conditions, and the conditions in which authorization should be denied.

Security model is the implementation of access control mechanism. In other words, we can say that it is a working model of access control security policy set.

Security mechanism describes the various low level (i.e software as well as hardware) functions which implements the rules given in the policy set which are stated in access model.

2.3 Early access models

2.3.1 Discretionary Access Control

DAC was defined by Trusted Computer Evaluation Criteria[9]. In this, the owner of the object (file or data) can grant and/or revoke access privileges of the objects through the predefined policy[3]. It is an access control model which does restriction on access of the objects on the basis of identity of user, process, and/or groups. The access controls models are discretionary as a subject which has access rights can transfer that access rights to any other subject in the system. DAC is most commonly used access control in systems and networks[14].

DAC is a type of access control framework that permit subjects to grant specific restriction for accessing the object. It uses access matrix model[10]. DAC was developed for the implementation of access control matrix which was defined by Lampson during his research on system protection. ACM are generally 3D matrices that have subjects represented by row, columns represent objects and subject to object mapping pairs contains

access rights subjects has over that object[10]. As access control matrix has one column for each object and one row for each subject, the number of entries in the matrix are equal to the total count of users/subjects in the times the total count of objects the system. This implies that $O(n)$ increase in subjects and/or objects will lead $O(n^2)$ increase in ACM matrix size[2]. The matrix is not dense as most of the subjects does not have access to the most of the objects. So, the access control matrix is sparse instead of dense. The size of the matrix will not be taken into consideration when the matrix is dense. As matrix is sparse large space will be wasted and look up would also be very expensive and time consuming. That's why DAC are stored either as file permission modes or as lists[2].

DAC is a traditional access control in which the user can control the management of all the programs. DAC is consisted of access and attributes rules[12]. The access attributes gives permissions to the system to outline various distinct level of authorization, and the access rules gives the technique to stop unauthorized access to the sensitive data[12]. DAC provides controlled way of sharing object between different subjects. DAC is the kind of mechanism in which owner determines "which subject will access what objects", as it allows the owner of the objects to grant access permissions to other users.

DAC permits legitimate user to change or alter attributes related to access control of object, so they can specify whether other users should be able to have access to the object or not. DAC is controlled by the owner of the object or root/administrator of the OS, it is not hard coded in the system[2]. In DAC, the predefined policies are on the basis of the requester identity and it has access control permissions that permits who should be, or who should not be, able to execute what actions on given resources[6].

A simple example of DAC is file passwords, where access to the particular file is given to the user who knows the password created by owner of that file. Similarly, in Linux the file permission is given in the form of DAC. Setting permissions of files, directories, and other shared resources is also an example of DAC. The owner (the user who has created that object) in most of the operating system applies DAC. The file ownership can be transferred or managed by root/administrator of system[10]. Another example is file mode used in UNIX, where write, read and execute permissions given to users, groups and others[9].

It enables users to have fine-grained control model over objects of the system using

fine-grained controls. DAC can also be used for implementation of the principle “least-privilege access”. Each individual objects will be able to restrict individual subject access until they have minimal rights required. DAC is framework which is intuitive in its deployment and it is also invisible to the users which makes it most cost effective and suitable for small-business users[2]. The DAC model provides the plasticity of usage on information. This technique can maintain the authorization information that consists variety of approved user[12].

The main disadvantage of DAC is that it is discretionary, as DAC gives permission to owner to permit grant and/or revoke access rights for the objects its own. It makes the access control mechanism discretionary. In DAC, controlling access controls is very difficult. Relationship between clients in the system can’t be shown in clear manner, which makes management of access control tough. DAC fail to differentiate between humans and computer programs.

In DAC there’s no assurance on information flow and additionally there’s no restriction on the usage of data this can build the confusion on the usage of data and additionally information are going to be lost. It can be attacked by third parties easily[12].

DAC permits user to define access control policies on the object it owns. The policies are global, hence it has difficulties in ensuring consistency. It is due to global policy of DAC. These policies allow subject to determine access control rights on its resources or objects and as policies are global, so DAC cannot enforce consistency. Consistency is not maintained on information. There is probability of pirating the duplicate of original memo when it is not owner’s authority. Usually owner might amend the DAC policy set by inserting Trojan horse.

There is lack of constraints on copy privileges, i.e., if one tries to copy data from a file to other file, it makes it very complicated to preserve safety policy. It is very difficult to confirm that these policies are not adjusted when trying to open potential exploit for Trojan horses. As it allows users to manage access control as a result it makes the system vulnerable to Trojan horse attacks. Also, maintaining system and verifying security principles is very difficult as users has control over access rights to its objects[2].

It is vulnerable to the process as it executes malicious programs. When it executes the malicious programs or software, it exploits the identity of given user in place of

whom they are being executed. Example of such case is Trojan Horses. As the process acquires particular information because DAC does not have control over information flow. Data can be transferred from an object to other object, hence it makes it possible to get duplicate copy of the data even when the owner did not give the access rights for the original information[9]. DAC is a “need to know” type access control model. It supports realization of “least privilege principle” where users are allowed to access only the information which are needed (no more, no less) based on his/her credentials. DAC gives the flexible environment for accessing the resources[9].

In Contrast with MAC, in which rules are determined by system administrator, in DAC rules are determined by the owner of the object. Here, owner of the object has the entire management over all objects and resources that it owns. It can also determine the permissions for different user who has these objects[6].

2.3.2 Mandatory Access Control

MAC was formerly modeled by Bell and LaPadula. It is latices-based access control model that was designed as more secure and stricter access control model than discretionary access control [10]. It is kind of access control mechanism which restricts access rights to objects on the basis of predefined security “labels” which are assigned to the subjects and objects. In this, access controls are mandatory as they are enforced by system and they must not be changed by users and by user’s programs. System administrator allocates a particular security label/attribute to the object and subject present in the system. Usually, one subject should not be able to modify/alter attributes(related to security) of other subject. Here, the system plays a role in determining either the subject should give access to the object or not by evaluating the security labels/attributes of object and subject[10]. The MAC model is a framework for computer security which permits subject to have access rights to resource objects, these accesses are managed by operating system based on configured setting of system administration. Here, subject will not be able to change control list of resources. A subject is given access to the object on the basis of security labels which are determined by system or network administrator and enforced by the operating system.

Security levels assignment is done by the root of system under the restrictions enforced by MAC, according to security policy set. It does not allow dynamic change in the policy set, and it also needs most of the parts underlying operating system and utilities associated with OS “trusted”. It should be placed outside framework of the access control[10]. In MAC, individual cannot amend the access rights. Disadvantages of MAC is that when the explicit subject within the hierarchy knows the security level, it won’t change the security level[12].

Users are not allowed to override the rules as well as policies and thus it is managed centrally by the system administrator. System administrator defines how to use and who can use the resources and it also defines their access policy. Users can not modify or override these policies. System administrator will also decide to whom access rights should be given for programs and files. It is more appropriate for the system in which priority is based upon confidentiality. It is an access model in which a user cannot modify access rules. It completely relies on central system. Policies in MAC are established by administrator, and it should be followed strictly by security kernel or the OS.

MAC can be classified into two form.

Multilevel Security: In this type of MAC, data of the system and users in the system are given discrete levels according to their trust and sensitivity. Data is classified in following three categories: Confidential, Secret, Top Secret. It would also define different levels like clearance, classification and security level.

Multilateral Security: MAC policies do ensure high security than the DAC policies, hence indirect flows of information can be controlled. Many policies were proposed as a combination of discretionary authorizations and MAC[6].

MAC makes restrictions on user access rights and it also does not allow dynamic alteration according to security policy set. It must place OS and utilities related to it outside the framework of access control. To implement it effectively advance planning is required. After the implementation is done, it requires high management of system because of constantly updation of object and security labels as we collect recent data[6].

MAC is required when the access rules of the system states that the owner of the resource should not make any decisions regarding authorization. In real-life, a label on an object is generally named security classification and the label on subject is known

as security clearance. In theory, the system will not allow any kind of data transfer outside the given constraints. DAC and MAC, both are not mutually exclusive. In real life, several systems use both models in conjunction. Here, mandatory access controls are very essential but they are not sufficient conditions for given access. They should be verified first, and then we should check of discretionary access controls. Similarly, mandatory access controls are kicked in until the discretionary access controls are checked and satisfied. There are various access control frameworks that can be used to implement DAC and MAC[14]. MAC is very important to military applications but it is not very much used technique of access control[2]. DAC and MAC are used for secure operating systems[9].

Processes and libraries are considered as trusted components, like declassifying cryptography processes, which violate MAC principles and that's why they must be outside the framework of MAC model. For the maintenance of security policies defined in policy set and to prevent inappropriate or illegal access, the code or process responsible for these trusted components are assumed correct according to the security policy set of system. Extra access control models are used to prevent access rights of trusted components. In examples, it is observed virtually that it is not possible to do implementation of MAC without removing the complete operating system and some utilities associated with it outside the framework of MAC framework. It should be into the domain of these trusted components[2].

It does not address the least privilege, validation of trusted components or dynamic separation of duty or security. System having MAC are expensive and difficult to implement as reliance on those trusted components and as it requires the applications to be re-scripted adhere to MAC labels [2].

Mostly MAC are used for a system in which priority is decided based upon confidentiality and security strategies are simple and security rules are definite. Examples : Trusted Solaris, SELinux, TrustedBSD. Some UNIX operating systems use MAC. MAC is also used in military field and government by allocating a pre-defined label to the file system object. As security policy in MAC does not allow users to have control on access policy applications and to declassify information, it makes the system safe from the Trojan horse[9]. A MAC is moderately straightforward and also considered as a better framework

for commercial systems which operates in hostile environments (i.e. financial institutions and web servers) when the possibility of attack is high, confidentiality is a main concern of access control and the objects that are being secured are valuable[2].

2.3.3 Role Based Access Control

RBAC framework provides a powerful technique that satisfy access control needs. An access control policies are statements about specification of rules which determines how to set up processes for either granting or denying authorizations to the given users. The role is central concept of RBAC[10].

In RBAC role is intermediate element, which is assigned to user, between users and permissions. When RBAC implementation is being done, it assigns users to roles directly (these assignments are many-to-many). It assigns and permissions to roles (these assignments are many-to-many), hence permissions are not assigned directly to the user. RBAC permits a user to activate two or more roles at the same time in one session, though it isn't compulsory to have all roles active. Sessions can do implementation of the standard RBAC model, that supports assignments of user-permission(many-to-many).

There are two part of the standard RBAC model each part contains one function. First functions is to review set of users/subjects that are assigned role and the set of roles given user. Second review functions are not compulsory, as they are advanced[10].

Most of the organizations and enterprises go with RBAC as access rights to objects are given on the basis of roles assigned to users in that organization. As computer applications are in continuous developing mode, and computer security changes continuously but MAC and DAC only supports some of the access control demands. Roles are defined on the basis of responsibilities, authority and job to be done within the organization, user can access privileges which are bound to their role. Therefore, it is also called as Non discretionary Access Control. One of the many advantages of RBAC : it can create roles and can modify the roles as per requirements of organization, hence making it easier to manage policies. However, this advantage can be disadvantage also as it does not allow applying fine-grained access control policies for a customized access control environment[9].

As its name suggest it defines the sets of access rights which could be given to users,

it's also simple to audit that which users can have access rights to a particular permission. It also easy to verify what permissions had been given to a particular user. Roles should be engineered before RBAC is used. And RBAC should be able to constrain to handle if attributes changes dynamically, like time of the day and user location. Standard RBAC can not handle such dynamically changing attributes[5].

In DAC model the user access permission of data has an important part, and these is not a good. While in MAC, users must have security clearances and objects must have security classifications. By doing combination of the enforced organizational constraints and explicit authorizations' flexibility, RBAC tries to reduce this gap. RBAC are used mostly for controlling and manages the access privileges to the computer resources. RBAC is also very useful framework for managing which kind of information users should have access to on the computer, what kind of programs can be executed by the user, and what type of changes the users are allowed to do. In this, roles are assigned to users statically, so it can't be operated in dynamic environment. It is very difficult to modify the access permissions given to the user without modifying the role specified to the user. As the role assignment is done statically, it doesn't have much complexity. So, it does not require much attention for system maintenance. In a way we can say that role is the abstractions of behavior of the user and duties assigned to them. To provide the accessing control mechanism which provide safety and security in the given systems it is beneficial to have concept of role. It decreases authority management cost as well. Essentially, in RBAC there is need of identifying the roles of user in the system, a role is a group of responsibilities and activities associated with working activity. It gives authorized user to access their associated responsibilities. Users can have access to the particular objects by doing execution of transactions on that particular object. In this, we are dealing with illegal or inappropriate access to the system and data, as we have to consider access rights that users need to execute a given transaction on a given object from the predefined objects set. While allotting access rights to roles it is necessary to ensure that principle of the least privilege is being satisfied i.e., each role must only required access rights as per its functional requirements.

Disadvantages of RBAC is that there is always some work undone as to satisfy all the requirements that are represented in the real world is very difficult. Adding the new

roles in a various context is complicated and it can lead to large number of role definition in the system. Sometimes it may produces more numbers of roles than number of users in the system. Fine grained results are required but it does not give fine grained results. It is very difficult to implement RBAC where the environment of the system is dynamic and distributed. It makes it tough to modify the access privileges of the user unless modifying the role of that particular user. It keeps the relation among its users and roles well maintained. It also keeps maintenance of the relation among its permissions and its roles. Hence, to implement the RBAC, roles should be predetermined and it is impossible to modify access permissions without changing the roles. RBAC has many advantages. Some of the advantages of ABAC are described below.

Authorization management: It provides logical independence while specifying user authorizations. We can break down the user authorizations into two parts: first is assignment of roles to the given users, and second is assignment of objects to roles. This make it easier as well as simpler to control the security policy. For instance, when there is a new user joining the organization, the administrator have to allocate particular roles according to work responsibilities; when an already existing user's work responsibilities get modified, then the system administrator just have to modify the roles according to user. If a new function is added to system, then the system administrator just have to determine which roles are given permissions to execute it.

Hierarchical roles: There can be a hierarchy of roles In some organizations, roles hierarchy on the basis of principles of generalization and principles of specialization. The hierarchy of role could be used to know the details of authorization. For instance, authorizations would be granted to roles as per the specializations. Authorization implication could be mandatory on assignments of roles, by permitting users to use generalizations. The hierarchy have to be exploited into administrative privileges: according to hierarchy of roles in organizational, an additional hierarchy can be defined for the administrative roles.

Least privilege Roles: It describes that the least privilege which user need to have to execute the particular operations. The users who have authorization to execute powerful action do not need to use this authorization until and unless those access rights are needed. The minimal damage sometimes happens due to the accidental errors.

Separation of Duties :It specifies that user can not have more access rights than needed so he/she can misuse it. For example, the person who can authorized paycheck, and person who prepare that paycheck must not be same person [6]. Duty Separation can either be done dynamically or statically. In statically, it will be done if we define conflicting roles. In dynamic behavior, it will be done if we provide the access control at the time of access. The concept used in the RBAC model is do assignment of access privileges into two steps: first step is to separate or distribute permissions into system-specific roles, second step is to assign roles to the users[6]. RBAC simplifies access control management[4], and it solves problem occurred during accessing the systems, that have large number of users as well as large number of data objects, and many types of access permissions. The main concept of RBAC is introduction of the role, it gives authorized access permissions to the roles instead of directly giving it to the users. And similarly, users may obtain these corresponding permission as per the role assigned to them [4].

2.4 Related Work

2.4.1 Attributed Role Based Access Control

RBAC provides high security and manages permission easily. It is known due to its robustness as well as ease of administration for access rights. But it has some drawbacks such as role structure complexity. It also does not support dynamic behavior. ABAC gives ease in implementation of access control system as it has attributes. But, it is a difficult job to manage permissions for the administrator. While ABAC provides dynamic access control and it also gives ease in role structuring unlike RBAC. After implementation of access control model, analyzation and management of roles and permission given to that role are difficult in ABAC[1]. Therefore, they are trying to develop model that could provide high security as well as ease of management, and does not have difficulty in role structuring for dynamic behavior. They aimed to provide a new access control model by combining ABAC and RBAC to have strength of both models and for removing their deficiencies. In this paper, a new access control model is proposed that captures the strengths of ABAC and RBAC while removing deficiencies of both models. The proposed

model does implementation and exploitation of the of object's attributes, users, roles, and permissions. In addition, the proposed model has the ability for the role structuring as in ABAC as well as it maintain the tight security behavior of RBAC. Model has mainly two parts. First part of the proposed model creates automatic permissions. Users are assigned roles in second part. Attributes are used while assigning the roles. So, first the permissions are defined and then assigned to roles followed by roles assignment to users based on the attributes of users. As creation of permission, assignment of permissions to role as well assigning role to the user are done automatically with the help of subject and object's attributes, this technique decreases the load of administrator. This model support ease of management, dynamic behavior, and automation of role structuring. This proposed model is secure as the basics of the model are similar to RBAC, while the model support dynamic behavior as attributes are added[1].

2.4.2 Mandatory Access Control Model with Enhanced Flexibility

DAC and MAC are two broadly used access control models that are used for operating systems' security. DAC is mechanism that is based upon the identity of and/or groups while MAC relies on sensitivity labels. Both modes can not satisfy all the requirements needed in an ideal access control model. DAC is not so strict to restrict privileges propagation and MAC is not flexible. Mostly these two models are combined by confining DAC in MAC framework like Bell-Lapadula model. It brings some flexibility in access control. This paper talks about few scenarios that are not handled by standard MAC based on bell-lapadula model. It also proposes a new access control method to combine the flexibility provided by DAC with security of MAC. The system which is responsible for the security and other things related to safety and security in operating system is called system security officer (SSO).

They have implemented security policy as given below. They have adopted traditional DAC which was specified by SSO by providing all the objects and subjects. They have also adopted MAC. They have used authorization matrices for recording temporary authorizations. First we have to check MAC and DAC. After the request will be rejected

by MAC and DAC, it will search for special authorization that can allow the requesting subject to access that object. If that special authorization exists, then that request will be accepted. They have considered only file as object for simplicity.

Objects are classified into two types to represent different state of objects. First type is *draft* and the second is *published*. If the type of object is *draft*, only the subject having same Range ID as the object can read or write. When the type of object is *published*, only the subject having same Range ID as the object should be able to read, not write. Subject can write the object with *published*, if only if it is given access by MAC rules. So we can say that the *published* type can be read discretionary and can be written mandatory. This will protect object's integrity with high security level. Also, we have defined the rule that the subject will be able to write objects if it has at least maximum security level that subject has. When subject does not have Range ID, then security level will be equal to the current security level of that subject. It will guarantee that data stored in the file will not be leaked by doing write operation. A subject must have only one RID. If subject can have multiple RIDs, then it can lead in illegal information flows.

Flexibility is main issue when system having MAC is considered. Classical BLP model has DAC but it does not capture the flexibility of DAC. FEMAC model combines the MAC model to DAC model. The proposed model will have the security provided by MAC along with flexibility provided by DAC. By introducing a temporary security property, special authorization is implemented to improve flexibility of the system. It is not a simple combination of DAC and MAC, rather it's a combination of two types of access control models. It also takes the relationships between these two access control modes into consideration while combining this two model. They analyzed all the feasible information flows that can happen between the given two types of access control frameworks and they have assured that this information flow are legal in FEMAC. If want safety and security in the system then we must also consider flexibility as well. Only system having flexibility should be actually applied.

2.4.3 Mandatory Access Control in RBAC

RBAC was introduced with claims that DAC and MAC can be incorporated into it. This paper presents a new access control mechanism which gives a realization of MAC in an existing framework of role-based security[8]. We have to label all the users with clearances and we also have to label data objects with classifications. RBAC are generally used in commercial sectors. In this framework, users are provided membership in terms of roles which is based on their job, responsibilities, and capability in the organization. This makes it easy for the administration to manage privileges and permissions as roles can be added, deleted or updated without modifying the access rights for each and every user in the organization. MAC enforces the policy which have information flow only in one directional in the lattice. MAC is less flexible than RBAC, but MAC has features that are also needed where RBAC flexibility is also used. Since the RBAC is introduced, researchers have discussed the relationship of RBAC with other traditional models. They have also attempted to configure RBAC such that it can enforce MAC and DAC models in its framework.

This paper provides a way to enforce MAC in an existing RBAC security system. They have deployed MAC in Patient DataBase using CORBA. Application has predefined the security classification of resource, subject, service and method. They have defined the least security level of method as the service classification level, and the least security level of services as the resource classification level. At the time of role creation itself, classification level is assigned. Clearance level will be assigned to a user when it is created. While a role to the user, the user must hold equal or higher level clearance than clearance level the role itself. They have to label all the users with clearances and we also have to label information objects with classifications. Main concept used is to consider clearances of subject and classifications of object as constraints. In a way they are constraints. For practical use it would be better to have a system that will support standard RBAC and can specialize these into MAC.

2.4.4 Compatible and Usable Mandatory Access Control

OS security is one of the many security problems. Although there are many access control methods, but they can not be accepted by some users due to its compatibility and usability. This paper has introduced a new MAC model, which is named CUMAC. It is designed to have high security, usability, and high compatibility. It mainly has two features. First is access control which is based on tracing intrusion that will decrease false negatives and will provide security configuration, it will improve compatibility as well as usability. Second is solving all the compatibility exceptions automatically that generally leads to incompatible problems. Experiments were performed on the proposed CUMAC model, it has shown that it can defend attacks from network, mobile disks, local not trust-able users by not accepting their requests for performing operations which are security critical.

This paper proposes a new form of access control which does not allow the system to compromise in a compatible as well as usable manner in order to handle the OS problem. It was designed mainly for three goals. First is good security, it should prevent attackers (remote and local) from taking over a host. At the same time it is not needed to have a higher security level than required as well. Second is high compatibility, it should be compatible with the already existing implemented software. This means that CUMAC must not replace or modify currently existing software and their configurations for operating smoothly, and it must not fail working software. Third is high usability, it should be more transparent for the user. This satisfies that the user must not be needed to learn new functions to work in a CUMAC environment, and configuration of CUMAC must be done automatically.

The CUMAC policy can be defined as : Every process and file will have either of two states, which can be either non-intrusion or potential intrusion. When a potential intrusion enters in OS, CUMAC gives labels to processes as well as to executable-files relevant to that intrusion as potential intrusion. CUMAC will trace these processes' activities as well as activity of processes that will be derived from the files which have potential intrusion labels. CUMAC also does labeling of the processes forked by them as well as processes which communicate with them as potential intrusion. When processes which have labels request for operations that are security critical, CUMAC does not

allow them in order to prevent host compromise. As in traditional MAC systems entity labels are configured manually by administrator, the main benefit of tracing potential intrusion is that it can label OS entities automatically. As access control which are based on the tracing potential intrusion decreases false negatives which are the main reason for incompatibility in system with MAC. The good feature is that it can solve all compatibility exceptions automatically that caused incompatibility problems.

CUMAC is a kind of access control which provides more compatibility, good security and usability to user. It deploys two methods. First is to trace intrusion automatically and mark potential intrusion entity to not allow these potential intrusion processes to execute operations which are security critical. Second is to solve exceptional accesses automatically in order to remove compatibility exceptions. In other way we can say that CUMAC is kind of access control model which performs access control on the basis of tracing potential intrusion. Experiments were performed on the proposed CUMAC model, it has shown that it can defense attacks from network, mobile disks, local untrustable users by not accepting their requests for performing operations which are security critical. It also offers high compatibility with currently existing software and also provided high usability to normal users[11].

Chapter 3

Attribute based access control

3.1 Introduction

Attribute based access control is technique which decide whether access can be given or not to requesting users by checking against access policies which are defined by combining relevant attributes (of object and subject). It provides a mechanism which enables fine grained access to the data/resource. It helps to realize the most needed principle, principle of least privilege to ensure the safety and privacy of resources and data stored in the system.

In ABAC access privileges are based on the set of attributes of subject and object. In RBAC there is lack of assignment of privileges to the user which has been solved in ABAC by having set of attributes of user. It is also known as authentication primarily based access control. In ABAC the attributes are basically based on three components : user's need, the type of access rights user need to access and requested services of user [12].

Role are not needed in ABAC until and unless roles are used as attribute in access control mechanism. There may be some attributes which changes dynamically, for example time of day, user location, they can be considered while deciding access control decision. There should be an expert personnel who will be responsible for managing and selecting large number of attributes. Attributes does not have any meaning until and unless they are associated to a user, a relation, or an object. It is not valid to audit who has access

rights to a particular object and which permissions are granted to particular user.

ABAC model is consisted of two parts: first, the policy model that specifies ABAC policies, and second the architecture model that is applied to the policies for web services. Unlike RBAC, it defines permissions based upon any Attributes related to security.

For access control, attributes are classified into three types as given below:

1. **Subject Attributes** : It can be user, process, application, process or device which executes action on data/resource. Every subject has its own attributes which determines the identity as well as characteristics of that particular subject. These attributes can be the subject's id, name, job title, organization, etc. Role can be considered as an attribute as well.
2. **Resource Attributes** : It is an entity (data structure, system component, a web service) on which subject takes action. Resources also have attributes that are taken into consideration while making access control decisions. For example, Word document, it has attributes like title, date, subject, author etc. Attributes of the resources can be taken from the "meta-data(data about data)" of the information.
3. **Environment Attributes** : These are mostly being ignored in some of the access control policies. Environment attributes describes the technical, operational, and sometimes situational environment, and situation in which data access occurs.

For an instance, attributes like current date and time, location and the security level of network (Internet or Intra-net), are neither associated with a subject nor a resource, but they relevant when applying an access control policy. Hence, it is believed that ABAC model is a natural convergence of other access control models. It can surpass their functionality as well. Representation of policy is rich and expressive as it is based on combination of subject, resource, and environment attributes[14]. In ABAC, access rights are not given to the subject directly. It considers subjects attributes and objects attribute while providing authorizations to user. For subjects, we will consider static attributes such as subject's id, subject's name, or it's designation in a company and there will be dynamic attributes such as age, current location of subject or session time etc. For objects, we will have meta-data properties like subject or title of document could be used. In ABAC

permissions depends on combination of values of attributes of subjects and object. ABAC does not have user role assignment issue presents in RBAC and rather than focusing on the roles of user, it focuses on user attributes to decide the access permissions. ABAC provides more flexibility in a sharable, distributed and dynamic environment when total users are large in quantity. It makes ABAC framework flexible for administering functions and it works better than RBAC. ABAC supports global agreement of attributes, such that attributes given in one domain can also be forwarded to another domain at the time of interaction between domains. ABAC can also be used in web administrator in order to improve structure of the website using Web Enhancement Model. It also increases the interoperability between various service providers to determine the user rights[6].

Due to the involvement of attributes in ABAC, it offers better security than other access control models. ABAC is more secure, adaptable and climbable. It provides perpendicular design. ABAC does not provide the subject role detail notion [12]. Due to heterogeneity in the information of the user complexity is increased, hence to reduce this it is required that the central database should have all attributes in identical format. And if various organizations choose similar/common set of standardized attributes, it will lower expressiveness of representation of the subjects and objects, hence it will lose the flexibility and dynamic functionality [6]. ABAC gives ease in implementation of access control system as it has attributes. But, it is a difficult job to manage permissions for the administrator. While ABAC provides dynamic access control and it also gives ease in role structuring unlike RBAC. After implementation of access control model, analyzation and management of roles and permission given to that role are difficult in ABAC [1].

3.2 Specification Of ABAC

3.2.1 Basic concepts

We can specify an attribute of the object by an identifier i.e., attribute variable, data type of that variable, and domain of that (domain is a finite set of values of that particular data type) of that variable. Attributes data types of can be typical data types used in computing systems like integer, string, float, char and boolean. The data types can be

specified implicitly or explicitly. Similarly, domain of attributes in a given ABAC system can be specified implicitly or explicitly. Table represents various attributes for library management system. An ABAC policy is represented by the function which decided whether requested access can be allowed or not, given the attributes values.

First row represents an attribute “role”, that can have value from (student, librarian, faculty). Attribute of role is of string data type. Similarly, last row represents an attribute “day”, whose value can be from (0-working, 1-holiday, 2- maintenance) of the string type.

Table 3.1: Attribute variable and its domain in library management system

Variable	Data Type	Domain
Role	String	{student,faculty,librarian}
Object	String	{book,account}
Activity	String	{borrow,return,read,write, fix}
Day	String	{working, holiday, maintenance}
Timing	String	{Working Hrs, Non Working Hrs}
Clearance	String	{unclassified, confidential,secret,top secret}
Security level	String	{unclassified, confidential,secret,top secret}

An ABAC policy can be defined as : $\langle X, Y, f \rangle$ where:

1. X specifies set of attributes whose domains are D_1, \dots, D_n ,
2. Y specifies set of access control decisions (not defined, permit or deny),
3. f is access control function,

An ABAC policy is said to be complete if f is a complete function i.e., for given set of any attributes values, f should give deterministic decision all the time. Different ABAC framework uses different access control decisions like permit, undefined, deny, Not-Applicable, intermediate. There are many ABAC specification languages already identified which gives different approaches to specify an access control functions which uses rules. Completeness of the policy set is generally gained using default or predefined decisions specified (mostly deny) for the situations which are not given in policy set. As it is using rules, it raises the problem of conflicts and inconsistency – rules giving different decisions although same attribute values are given. As we are using rules, it can cause the conflicts or inconsistency in the policy set. Different rules having same attribute values can yield different

Table 3.2: decision table for library management system

Variable	Value	Rule1	Rule2	Rule3	Rule4	Rule5	Rule6	Rule7	Rule8
Rule	student	T	T						
	faculty			T	T			T	
	librarian					T	T		T
Object	book	T		T		T		T	
	account		T		T		T		T
Activity	borrow	T				T			
	return			T				T	
	read		T		T				T
	write						T		
	fix								
Day	working	T	T	T					T
	holiday					T			
	maintainance				T		T	T	
Timings	Working	T	T		T		T	T	
	Non working			T		T			
Clearance	unclassified								
	confidential			T			T		T
	secret	T			T			T	
	top secret		T			T			
security Level	unclassified	T		T		T		T	
	confidential		T		T		T		T
	secret								
	top secret								
Permission		P	P	P	P	P	P	P	P

decision. Conflicts of the rules in policy set should be resolved by implicit prioritization or explicit combining algorithms.

3.2.2 Decision table method

A basic way to represent an ABAC rule is using a decision table which represents the relationships between attributes of the subject and object. It has all attributes, domains, and decisions that is based on different combinations. Table 3.2 represents a decision table of library management system as given in table 3.2. Each rule in column defines an access control decision provided combination of attribute values. In other word we can say that it is a rule defined in propositional logic.

Here, rule 1 indicates that the access decision is P (permit) if Role is student, Object is book, Activity is borrow and Day is working, clearance is secret and security level is unclassified. Similarly, rule 2 indicates that the access decision is P (permit) if Role is faculty, Object is book, Activity is borrow and Day is working, clearance is top secret and security level is unclassified..

Notice that these rules given in the table do not specify a decision for the situation where Role is student, Object is book, Book Activity is borrow, but Day is not specified. The policy maker or policy administrator can predefine an access decision for scenario like this, we can use a default decision, or we can apply some rule. Else, we won't have the complete policy. It has a default decision for the situations which are not predefined. Conflicts between two or more rules can be resolved using some rule combining algorithms. For example First applicable, Deny overrides, and Permit overrides.

3.2.3 XACML

XACML is short form for extensible Access Control Markup Language. It is an OASIS standard which specifies ABAC policy set in XML format. Attributes are classified into four types : subject attribute, resource attribute, action attribute, and environment attribute. User-defined attribute might be defined for specific use. XACML handles different data types, path expressions, and attributes name types (e.g., int, char, string, regular expressions, XPATH, internet-based names). Instead of specifying domains of attributes, their data types are specified, while using attributes. Suppose there is an attribute day, whose value can be weekday, weekend. In XACML, we do not specify domain for role, rather we can directly use attribute values (weekday, weekend) for the attribute role. XACML gives various mechanisms for modularization to deal with complex policies. In XACML specification of policy is hierarchical. It can be consisted of one or multiple policy sets and policy set can also be consisted of list of rules. The target component specifies attribute-matching conditions and it also provides another level for abstraction and thus tries to improve the performance of attribute matching. A target is nothing but a logic formula created by logical operators (logical AND and logical OR) from attribute matching predicates. XACML also provides several predetermined attribute- matching

predicates (string-equal and name-match) for the approved attribute types and expressions. XACML also provides user-defined predicates that can be built from predefined as well as user-defined functions. Rule combining algorithms are used for resolving conflicts of different rules in the given policy. There are policy combining algorithms which are used for resolving conflicts of rules or policies in the given policy set. There are many policies combining algorithms such as deny-overrides, ordered-permit-overrides, only-one-applicable-policy, first-applicable, ordered-deny-overrides, permit-overrides, deny unless-permit, and permit-unless-deny. Access decisions (in other words answers to request access) are not only limited to Permit and Deny, but it can be Intermediate, undefined and Not Applicable.

3.3 Comparison OF ABAC Specification Languages

As specified in paper [13], there are more than one way to deploy ABAC. There is a decision table method, or we can implement it using XACML. Here is table which has compares different specification languages on different parameters.

Table 3.3: Comparison OF ABAC Specification Languages

Category	Decision table in ACPT	XACML	Logic framework	Our approach
Attribute types	Basic data types	Data types, XPath, URL	Symbols (single type)	Basic data types
Attribute domains	Explicit specification	Implicit specification	Explicit specification	Explicit specification
Attribute classification	Subjects, resources, actions, environments	Subjects, resources, actions, environments User defined	Subjects, resources (called services)	Subjects, resources, actions,
Hierarchical attributes Hierarchical attributes	No	Yes	Yes	No
Access decision	Permit ,Deny	Permit, Deny, Intermediate, NotApplicable,	Permit, Deny	Permit, Deny
Default access decisions	Yes	Yes	Yes	Yes
Formalism of rules	Decision table / Propositional logic	Predicate logic	Set theory and constraint logic	decision table
Conflict resolution	Explicit operators	Explicit operators	Stratification of ranks	Longest matching rule
User-defined functions	No	Yes	No	No
Policy modularization	No	Policy set, combining algorithms	No	No
Query	Combination of all attributes	Combination of partial attributes	Attribute predicates and constraints	Combination of all attributes

Chapter 4

Decomposition of Decision table

4.1 Specification Of ABAC

As specified in paper [13], there are more than one way to deploy ABAC. There is a decision table method, or we can implement it using XACML. So, here we are deploying the same example in decision table Method. A basic way to represent an ABAC rule is using a decision table which represents the relationships between attributes of the subject and object. It has all attributes, domains, and decisions that is based on different combinations.

Table 3.2 represents a decision table of library management system as given in table 3.1. Each rule in column defines an access control decision provided combination of attribute values. In other word we can say that it is a rule defined in propositional logic.

Here, rule 1 indicates that the access decision is P (permit) if Role is student, Object is book, Activity is borrow and Day is working, clearance is secret and security level is unclassified. Similarly, rule 2 indicates that the access decision is P (permit) if Role is faculty, Object is book, Activity is borrow and Day is working, clearance is top secret and security level is unclassified.

Rules

1. Student is allowed to borrow and return book on working day.
2. Student is allowed to read account details on working day.

Variable	Value	Rule1	Rule2	Rule3	Rule4	Rule5	Rule6	Rule7	Rule8
Rule	student	T	T						
	faculty			T	T			T	
	librarian					T	T		T
Object	book	T		T		T		T	
	account		T		T		T		T
Activity	borrow	T				T			
	return			T				T	
	read		T		T				T
	write						T		
	fix								
Day	working	T	T	T					T
	holiday					T			
	maintainance				T		T	T	
Timings	Working	T	T		T		T	T	
	Non working			T		T			
Clearance	unclassified								
	confidential			T			T		T
	secret	T			T			T	
	top secret		T			T			
security Level	unclassified	T		T		T		T	
	confidential		T		T		T		T
	secret								
	top secret								
Permission		P	P	P	P	P	P	P	P

3. Faculty is allowed to borrow and return book on working day as well as on maintenance day.
4. Librarian is allowed to borrow and return book on working day, holiday and maintenance day.
5. Librarian is allowed to read and write account details on all day.
6. Faculty is allowed to read account details on working day.

4.2 Authorization Policy

this section contains authorization policy of library management system.

1. If role is student and day is working then subject, having clearance secret, is allowed to borrow book with security level unclassified within working hours.

$borrow(student, book) \rightarrow (day == working) \wedge (clearance(student) == secret) \wedge (securitylevel(book) == confidential) \wedge (time(activity) == withinworkinghours)$

2. If role is student and day is working then subject, having clearance secret, is allowed to read account details with security level confidential within working hours.

$read(student, account) \rightarrow (day == working) \wedge (clearance(student) == secret) \wedge (securitylevel(account) == confidential) \wedge (time(activity) == withinworkinghours)$

3. If role is faculty and day is working then subject, having clearance top secret, is allowed to return book with security level unclassified in Non working hours.

$return(faculty, book) \rightarrow (day == working) \wedge (clearance(faculty) == topsecret) \wedge (securitylevel(book) == unclassified) \wedge (time(activity) == Nonworkinghours)$

4. If role is faculty and day is maintenance then subject, having clearance top secret, is allowed to to read account details with security level confidential within working hours.

$read(faculty, account) \rightarrow (day == maintenance) \wedge clearance(faculty) == topsecret \wedge securitylevel(account) == confidential \wedge time(activity) == withinworkinghours$

5. If role is librarian and day is holiday then subject, having clearance confidential, is allowed to borrow book with security level unclassified in not working hours.

$borrow(librarian, book) \rightarrow (day == holiday) \wedge (clearance(librarian) == confidential) \wedge (securitylevel(book) == unclassified) \wedge (time(activity) == Notworkinghours)$

6. If role is librarian and day is maintenance then subject, having clearance confidential, is allowed to write account details with security level confidential within working hours.

$write(librarian, account) \rightarrow (day == maintenance) \wedge (clearance(librarian) == confidential) \wedge (securitylevel(account) == confidential) \wedge (time(activity) == withinworkinghours)$

7. If role is faculty and day is maintenance then subject, having clearance top secret, is allowed to read book with security level unclassified within working hours.

$$return(faculty, book) \rightarrow (day == maintenance) \wedge (clearance(faculty) == topsecret) \wedge (securitylevel(account) == unclassified) \wedge (time(activity) == withinworkinghours)$$

8. If role is librarian and day is working then subject, having clearance confidential, is allowed to read account details with security level confidential in non working hours.

$$read(librarian, account) \rightarrow (day == working) \wedge clearance(faculty) == topsecret \wedge securitylevel(book) == \backslash confidential \wedge (time(activity) == Nonworkinghours)$$

4.3 Algorithm To separate Tables

In decision table, the access decision is taken on the basis of attributes of subject and object. Some of the attributes are static whereas some are changing attributes. Here, day and timings are changing attribute which means they are dependent on time of access.

Here, Rule 1 indicates that the access decision is P (permit) if Role is student, Object is book, Activity is borrow and Day is working, clearance is secret and security level is unclassified. It can be written as :

Student is allowed to borrow book If : $(day == working) \wedge (clearance(student) == secret) \wedge (securitylevel(book) == unclassified) \wedge (time(activity) == withinworkinghours)$.

First we have to check if student is allowed to borrow book, then if it is allowed we have to check for attribute constraints. Based on this, we can decompose the decision table into two parts. So, we can separate out this attributes in to different tables. DAC table will contain who can access what objects. Attribute constraint table will specify constraints based on attributes of subject and object. Instead of decision table we can verify authorization rules from these two table. First, we will check if DAC allows this access. If it is permitted then we will check attribute constraints.

Algorithm To separate Tables

We will do the following for each column of the decision table. As each column gives us one rule of policy set, so from each rule we can extract the information from DAC and attributes table.

For example, in above example, the first column in decision table says (student, book, borrow, working) if these are the parameter, then permission should be given. So, from these in DAC table there should be (student, borrow, book). Hence, for that we will check

row by row, whenever we get true value. These are the attributes we want in either DAC table or attributes table. In first column we will get true value in (student, book, borrow), therefore we will insert it into DAC table. We will also get true value in (student, working day) we will insert it into attributes table.

Input : Decision Table (by first column we can know what are the subjects, object, activity and other attributes of them).

Output : It will separate the given table into two distinct tables. First is DAC table and another is attributes' table.

Steps : Process the table column by column(as column gives us one rule, so from each rule we can extract the information for DAC and Attributes Table) and do the following :

1. We will check the value of table either it is true or false.
2. If it is true that means the row it is corresponding to is the subject or object or activity . In this way we can extract subject activity and object for DAC.
3. In similar way,we can extract subject and remaining attributes for attribute table from the decision table.
4. We will keep mapping between this DAC table entry and attribute table entry.

Tables After Separation

Table 4.1: DAC Table

	book	account
student	borrow 1	read2
librarian	return3, return7	read4
faculty	borrow5	write6

Here, It indicates student is allowed to borrow book. Student is also allowed to read account details.

Table 4.2: Attribute Table

	Day	Timings
1	Working	Working Hrs
2	Holiday	Working Hrs
3	Working	Non Working Hrs
4	Maintenance day	Working Hrs
5	Holiday	Non Working Hrs
6	Maintenance day	Working Hrs
7	Maintenance day	Working Hrs
8	Working	Non Working Hrs

4.4 Our Approach

We have developed a technique which says whether access should be given or no based on the attributes of subject and object. A system is implemented which uses decision table method along with underlying DAC, MAC and RBAC. First, it takes decision table of policy set and decompose it into DAC table and attributes table. Given a query which is combination of attributes of subject and object. It applies MAC policy first, that is security level of object and subject are checked. Based on their security level, it is decided that either access should be given or not. If it permits then we check for RBAC. It check if user has a role that is assigned to him or illegal role. If role is legitimate then it checks DAC and attribute table. We are also checking that if there is any indirect flow of information.

4.5 Some issues due to separation of the table

4.5.1 How separation of tables still satisfy all the rules mentioned in policy set?

Instead of looking at one big table, now we will have to look at separated table. While separating data, we are not doing any data loss, so it will still check all the constraints defined in policy set. Even after separating the table, it will satisfy all the rules mentioned

in policy set. It will be more efficient now. For example, if we want to check whether authorization should be given to the user or not. If we have not separated the table, we have to check whole table. But now, as we have separated the table into DAC table and attribute table. First we will check DAC table, if it allows then and then only we will check attributes table.

4.5.2 How can we make the system consistent?

Policy rules may conflict with each other, accidentally allowing unauthorized access. To avoid such scenario we must identify conflicts and resolve them. While adding new rules to policy set, we check if there is a rule already present in decision table which is a subset or a superset of new rule. There are two way to make the system consistent. First is before adding new rule. Check if it is conflicting any rule already present in table. If yes then you can't add that rule, otherwise we won't add that rule. Second is to set policy algorithm -(Rule combining algorithm) denying or permitting, if they are two rule that are conflicting one says permit and one says deny what should be done at that time should be predefined. Rule combining algorithms are used for resolving conflicts of different rules in the given policy. There are policy combining algorithms which are used for resolving conflicts of different policies in the given policy set. There are many policies combining algorithms such as deny overrides, ordered permit overrides, first applicable, permit overrides, only-one-applicable-policy, ordered-deny-overrides, deny unless-permit, and permit-unless-deny. For example, if rule 1 stating, permissions should be granted with parameters (student, book ,borrow, working day), is already present in the table and we want to add new rule (student,book,borrow,working day,working hours) which is superset of rule 1. These two rule conflict with each other.

4.5.3 How DAC, attribute based constraints, role based constraint are satisfied?

For DAC, there is separate table which provides who(subject) should access what(object). Before giving access to the subject, we are checking that table. For attributes, there is another table which tells either the subject should give permission or not based on the

attributes of subject and object. In this table we will capture the relationships among attributes and whether holding this relationship access should be given or not. For RBAC, either we can make separate table or we can combine it with DAC itself, as DAC is also based on the identity of the user itself.

Chapter 5

Conclusion

Attribute based access control is technique which decide if access can be given or not to requesting users by checking access policies which are defined by combining related attributes (of object and subject) to manage access to users. Using appropriate attributes we can represent access control lists and identities, clearances, security labels, classifications and roles. Therefore, we can use these attributes and realize DAC, MAC, RBAC in the framework of ABAC. We have implemented ABAC using the decision table method. A decision table is basic way to represent an ABAC rule is using table which represents the relationships between attributes of the subject and object. And by decomposing that decision table into explicit combination of DAC and attribute constraints. We can realize DAC in the framework of ABAC. We will get two different tables first is DAC table second is attribute table. While separating data, we are not doing any data loss. So, it will still check all the constraints defined in policy set. It will be more efficient now. In the dissertation, we show how formal model of ABAC using decision table can be realized by integrating information flow security models with constraints on attributes.

Bibliography

- [1] M. U. Aftab, M. A. Habib, N. Mehmood, M. Aslam, and M. Irfan. 2015. Attributed role based access control model. *Proc. Conf. Inf. Assurance Cyber Security* .
- [2] Ausanka-Cruces and Ryan. 2018. Methods for access control: Advances and limitations .
- [3] Vinith Bindiganavale and Ouyang Jinsong. 2006. Role based access control in enterprise application security administration and user management[c]. *IIIntegration 2006 IEEE International Conference* .
- [4] J. Cheng, R. Kang, and X. Zhao. 2013. Role based access control and its application in high speed railway. *Conf. Advanced Computational Intelligence* .
- [5] R. Coyne and T.R. Weil. 2013. Abac and rbac: Scalable flexible and au-ditable access management. *IT Professional* vol. 15, no. 3, pp. 14-16.
- [6] Bokefode Jayant. D, Ubale Swapnaja A, Apte Sulabha S, and Modani Dattatray G. 2014. Analysis of dac mac rbac access control based models for security. *International Journal of Computer Applications (0975 – 8887) International Conference on Internet of Things, Next Generation Networks and Cloud Computing STES* 104 – No.5.
- [7] D. D. Downs, J. R. Rub, K. C. Kung, and C. S. Jordan. 2013. Issues in discretionary access control. *Proc. IEEE Security Privacy Mag* pp. 208-218.
- [8] Jin Ma. 2001. implementation of mandatory access control in role-based security system. *CSE367 Final Project Report Professor Steve Demurjian Fall 2001* .

- [9] Nancy Ambritta P, Yogita S, and Santosh A. Darade. 2014. A survey on access control models and applications. *International Journal of Computer Applications (0975 – 8887) International Conference on Internet of Things, Next Generation Networks and Cloud Computing STES* .
- [10] Dipmala Salunke, Anilkumar Upadhyay, Amol Sarwade, Vaibhav Marde, and Sachin Kandekar. 2013. A survey paper on role based access control. *International Journal of Advanced Research in Computer and Communication Engineering* Vol. 2.
- [11] Zhiyong Shan. 2009. Compatible and usable mandatory access control for good-enough os security. *computer Science Department, Renmin University of China Computer Science Department, State University of New York at Stony Brook* .
- [12] Anjali soni, Dr. Sanjay Silakari, and Prof. Uday Chaurasia. 2015. A survey on access control mechanism used in cloud computing. *Anjali Soni et al , International Journal of Computer Technology and Applications* Vol 8(5),595-601.
- [13] D. Xu and Y Zhang. 2014. Specification and analysis of attribute-based access control policies: An overview. *oftware Security and Reliability-Companion (SERE-C) 2014 IEEE Eighth International Conference* .
- [14] E. Yuan and J. Tong. 2005. Attributed based access control (abac) for web services. *In ICWS'05: IEEE International Conference on Web Services, Orlando, page 569. IEEE* .

Acknowledgements

I am thankful to my supervisor Prof. R.K. Shyamasundar for his enormous support and insightful suggestions throughout my project. His insightful suggestions to the various problems that I faced during my project, were not only useful, but also helped me in broadening my basic understanding of project area. I am thankful to Dr. Vishwas Patil for his support and insightful guidance throughout my project.

Signature:

Sejal Patel

163050093

Date: July 2018