

# Extending UML to Define Access Control

Pushkar G. Dhande<sup>#1</sup>, Dr. Bandu B. Meshram<sup>\*2</sup>

<sup>#1, #2</sup> Veermata Jijabai Technological Institute(VJTI), Student, Department of Computer Engineering

**Abstract** - Access control means defining who/what can access which resources or to which level in system. In any software system access control plays large part in achieving security goals for system. As improper access control can lead to various vulnerabilities in system. UML is widely used in software development to represent structure and behaviour of system before developing it. UML can be extended to define access control for system more effectively as UML covers all aspects of the software beforehand actual development.

**Keywords** — Security, Software Engineering, Access Control, UML.

## I. INTRODUCTION

Softwares are becoming more widespread and more accessible to various users with advances in technology. This resulted in softwares becoming complex containing number of resources, sub systems and various types of users. Access control helps to solve this issue as it helps software developers define who has right to perform particular action on given resource. But improper access control leads to vulnerable software which can be compromised very easily. Vulnerabilities caused due to improper access control is included in OWASP Top 10 under Broken Access Control (A5).

Unified Modeling Language (UML) is modeling language which provides standard way to visualise structural and behavioural aspects of software system. UML don't deal with security aspect of system but various modifications such as UMLSec, UML2.0 are proposed to incorporate security in to UML. UML gives detailed view of software system before development phase which can be used to define access control policies in detail minimising chances of any flaws in access control.

## II. LITERATURE SURVEY

### A. Access Control Models

Access control models defines how access control is implemented in the software system. Discretionary access control model (DAC), Mandatory Access Control model (MAC), Role-based access control model (RBAC) are widely used in software to define access control.

#### a). Discretionary access control (DAC)

In DAC access to object by user is defined by explicit access rules which are fundamental part of operating system and database systems[13].

In DAC each file or data has an owner who defines access to that particular file or data. Owner can revoke and give permissions to other users.

DAC attributes include:

- Owner can transfer ownership to other user.
- Users can determine access rights of other users.
- After specific number of failed authorisation attempts user is blocked out.
- Unauthorised user have no information regarding file such as name of file, size of file, directory of file.

DAC is easy to implement and intuitive but has certain disadvantages, including:

- Inherent vulnerabilities (Trojan horse).
- DAC causes the Confused Deputy problem.
- ACL maintenance or capability.
- Grant and revoke permissions maintenance.
- Limited negative authorization power.

#### b). Mandatory Access Control (MAC)

MAC make sure that centralized security policies are enforced in system. In MAC on system administrator can manages access rights of the user. System admin grants and revokes rights to each user in system. MAC has two important characteristics i.e. 1) No Read Up means no user can access object above it. 2) No write down no user can write to object below its level[2]. MAC is mostly used with other access control models for large scale systems.

Advantages of MAC are as follows:

- MAC is provides more strict access control policies as access management is done only by administrator.
- MAC policies reduces the vulnerabilities in system.
- MAC enforced operating systems (OS) delineate and label incoming application data, which creates a specialized external application access control policy[11].

Disadvantages of MAC are as follow:

- It is difficult to manage MAC as everything is to be managed by system administrator.
- For large scale system using MAC alone is difficult.

- MAC make system administrator single point of failure.

### c). Role-based Access Control(RBAC)

In RBAC access to user are provided according to his/her role in the system. In RBAC various Roles are defined each role contains access rights required to carry out that particular role in the system. Each user is assigned with one or more roles according to their responsibilities in the system..More insight in the RBAC is given in [3]

In RBAC A user need to have at least one role assigned to it to perform task in the system. RBAC can be used to with MAC to create access control model further increasing the robustness of system[10].

Advantages of RBAC are as follow:

- RBAC reduces administrative work as roles are defined with proper Role Engineering[4]. So managing access rights of user is easy.
- RBAC offers easy logical view to the system making system working more streamlined and efficient.

Disadvantages of RBAC are as follow:

- Improper Role Engineering can lead to vulnerable software system.
- RBAC don't provide any functionality to provide single user with some extra right new role is need to be defined for such cases causing increase in number of roles in system.

### B. Principle Of Least Privilege

Principle of least privilege states that user must be give minimum permission to carry out his/her task. In system no extra unused permissions should be provided to any user. For example employee of any company should not have write access to salary.

Principle of least privilege can be used to harden security of software[14].

### C. Access Control Matrix

Access Control Matrix is an abstract, formal security model of protection state in computer systems access matrix represents right of each matrix against each object in system. Access control matrix acts as a lookup table to find access right for given user and object. In access control matrix each cell contains access privileges provided to user for particular resource.

### D. Access Control List

In Access Control List (ACL) list of all privileges is attached to user or group. ACL states which privileges are given to user and what task user can perform on object. Whenever user asks to access any object system checks whether user is allowed to

access that particular access by referring ACL attached to user.

### E. Role Engineering

Process of developing Role Based Access Control for organization is known as Role Engineering. Getting role structure right from the start can result in significant money saving and fewer operational problems.

#### Methodologies for Role Engineering

[4] gives various methodologies used for role engineering.

#### Top-down Approach

In this approach functional stakeholders and domain experts identifies the initial roles name and permissions associated with them . Steps followed to identify the roles. Identify actors and their operations on objects. Operations on object are permissions by definition (read operation on DB require Read permission). These permissions are high level permissions and not in technical realm. Once set of high level permissions are designed / identified IT specialist translates them to low level technical permission. Difficulty with this approach is that it is difficult to assemble and maintain team of domain experts for time of entire process.

#### Bottom-up Approach

In bottom up approach rather than trying to reinvent entire access model exiting rules are used to design new role based access model. Difficulty of this approach is that as existing systems are observed or used as reference system it is difficult to find single system that fits all the requirements.

#### Hybrid Approach

In hybrid approach domain experts design access model(top down approach) by using data collected from past systems(bottom up approach).

### F. Unified Modeling Language(UML)

*Unified Modeling Language (UML) is standard Modeling language used to design object oriented software system. UML provides with notations which can be used to represent structural and behavioural aspects of software system. Though UML is used widely to model software system, it does not contain notations to represent security of software system. Various extensions are provided to UML to incorporate security in it[7][8][9].*

*Table 1 gives the brief of UML Diagrams*

Use case Diagram	Use case diagram is used to represent how different users interact with systems. Use case diagram shows relation between user and various use case in which user is involved. Use case diagram can help in identifying actors in system.
Class diagram	The class diagram is the main building block of object-oriented modeling. Class diagram represents how various classes in system interact with each other and what are relationships among them.
StateChart Diagram	Statechart diagram represent dynamic nature of system. State chart diagram gives various states of object during its lifetime.
Sequence diagram	A sequence diagram represents interaction between objects in timely manner. It shows objects involved, messages passed among objects during performing an activity.
Activity diagram	Activity diagram is used to represent dynamic nature of system. Unlike sequence and State chart diagram which shows flow of information between various objects, activity diagram gives flow of information among number of tasks(Activities) in system.
Component diagram	Component diagram represents how various components are connected to each other and how they interact with each other.
Deployment diagram	Deployment diagram give view of how system will look after deployment of software system.

Table 1 UML Diagrams

### III. EXTENDING UML TO DEFINE ACCESS CONTROL


Use case Diagrams provides developer detail about all possible actors and use cases that an actor can perform. Use case diagram provides with all task activities that are to be done in the system. It also helps to find out possible actors in system who are going to use system.

Component Diagram provides an overall view of software system helping to identify possible components in the system which are accessible to any of the user. Deployment Diagram on other hand provides visualization of system after deployment which can be used to determine various assets in system be either hardware or software.

Using above information each use case can be mapped with required components required to carry out particular use case..

This mapping can be visualized using the following symbols

	Use case/Task	Represent task that is performed in system
	Asset	Represents assets available in system which are accessible to actors. Asset can be software or hardware.
	update/write access required	Represents privilege to particular asset. Update or write assets changes the data or state of asset. Privileges are system specific each system can have different privilege label
	read/get access required	Represents privilege to particular asset. read or get assets

		does not changes the data or state of asset.
	Database Table	Represents structure of table in database.
	Subset/ part of	Indicate that asset is part of or subset of another asset
	Actor/ Role	Actors are the entities that interact with a system.

**Table 2 Notations For Mapping Diagram**

Consider Example of simple Home Automation system in which user can turn on/off Device and see and update various parameters of device. System uses Mandatory Access Control and Role based Model Control model so every user in the system is assigned a role by administrator.

System contains three devices Fan, tube light. There are three roles used in system Administrator, basic user and privileged user. Basic user switch on/off fan,tube light ,set speed of fan. Administrator can give and revoke permission from user ,add new user to system update information of user.

Consider UML diagrams for Home automation system are as per use case(fig 1), Component Diagram (fig 2) and deployment diagram (fig 3.)

By referring use case diagram assets and actors can be identified in the system. Actors in system are Administrator and user.

Various use cases listed from use case diagram are as follow:

1. Remove User.
2. Add User.
3. Update User
4. Give Permission.
5. Revoke Permission.
6. Login.
7. Turn On.
8. Turn Off.
9. Get Device Info.
10. Set Speed.

Component diagram and Deployment diagram can be used to identify all the assets in the system. Assets in the system are as follow:

1. Database System
  - a. User
  - b. Device
  - c. Task
  - d. Permissions
2. Device
  - a. Tube Light
  - b. Fan

After all the actors, Use Cases and Assets are identified it can be used to map use case to asset using suggested notations.and mapping diagram can be drawn.

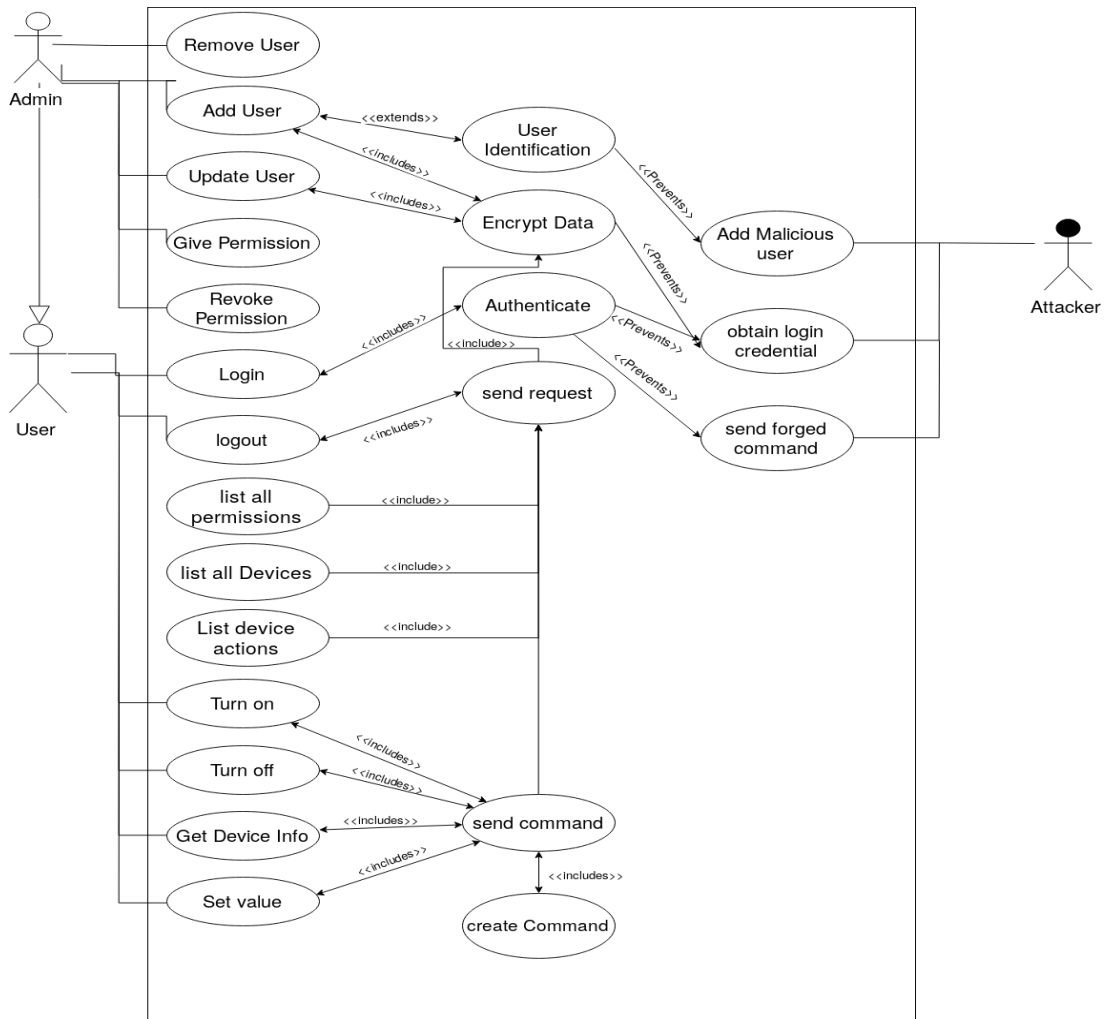


Fig 1 Usecase Diagram Home Automation

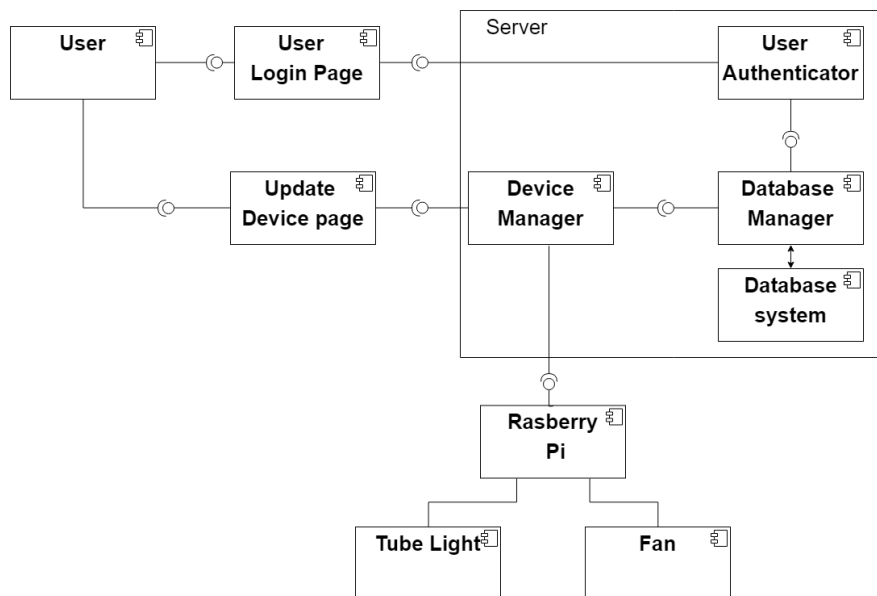


Fig 2 Component Diagram Home Automation

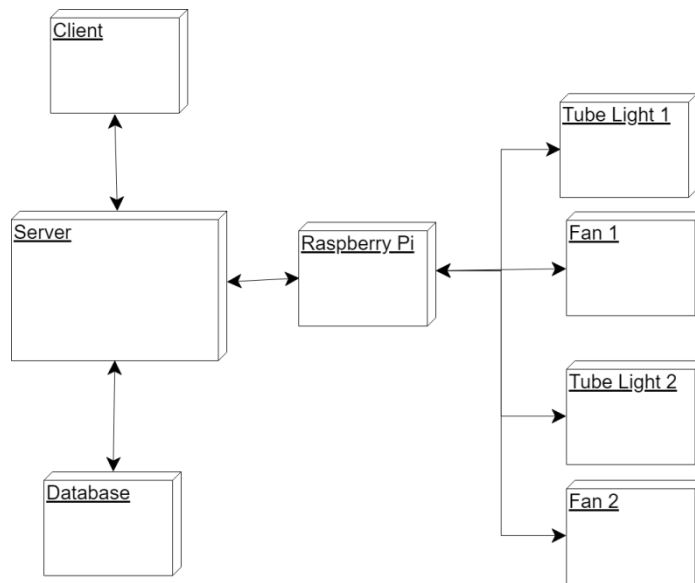


Fig 3 Deployment Diagram Home Automation

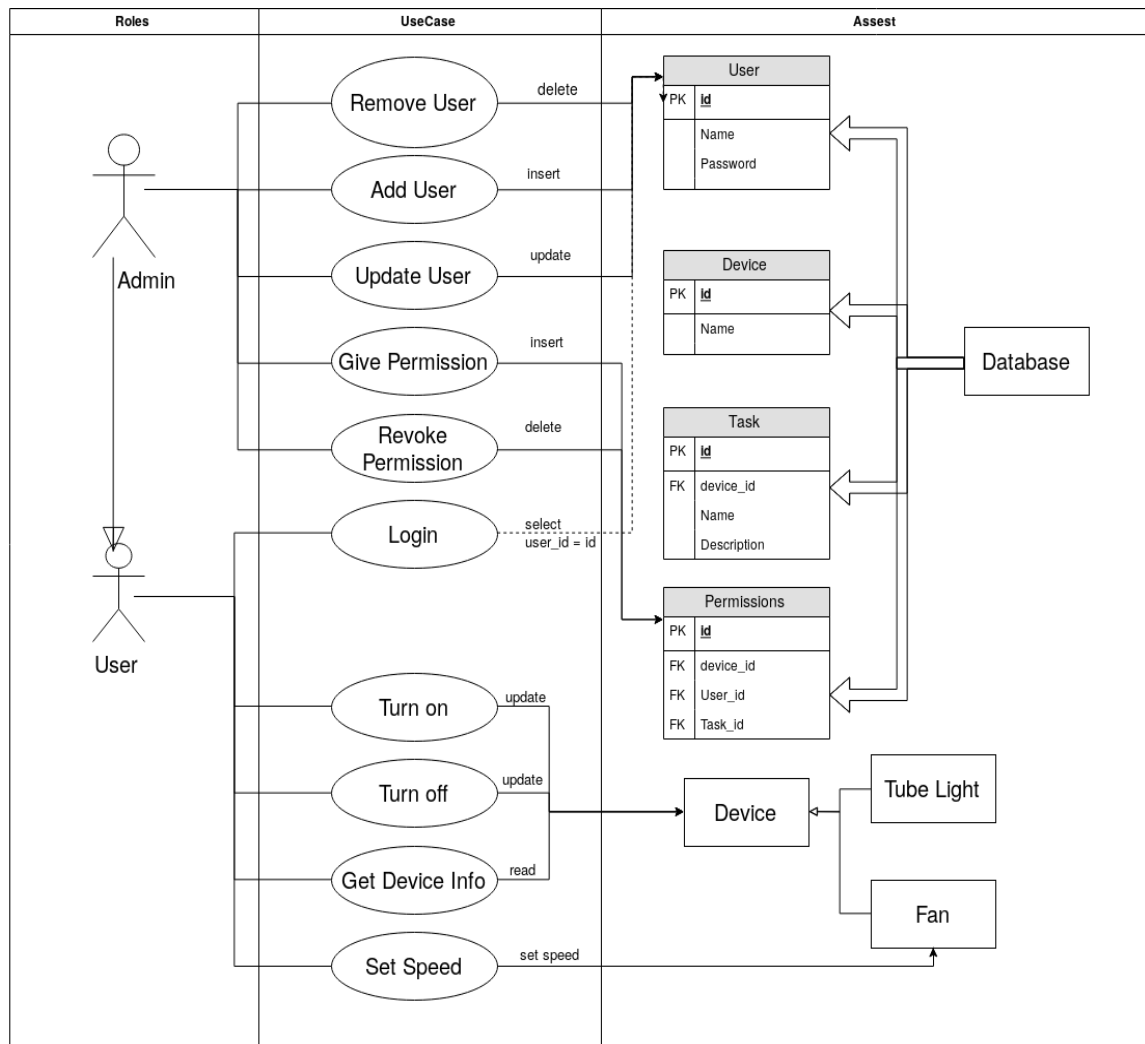


Fig 4 Mapping of Usecase and Assets

Fig helps to visualize the relation between various use cases and assets. E.g. admin can

perform Remove user and to carry out that use case actor needs delete permission on User table in

database. This privilege require to modify data in system so it is represented with solid a line. User can login in to the system for login use case user require select privilege on user only user id field of user table. User need not update data in system it only read permission therefore it is denoted by dashed line. Similarly various use cases can be mapped to various assets in system.

Once mapping is done it can be easily used to create the Access Control List of the system as follow:

Admin	User{delete,insert,update}, permissions{insert , delete}
user	user{select},device{update,read}, fan{setspeed}

#### IV. CONCLUSION

Access Control is one of the important aspects of any software. Improper access control can lead to various vulnerabilities in software system. Access control can be defined using one or more access control models. To define robust access control one need to have proper knowledge of entire system such as who are the actors in the system? What are tasks that each actor can perform? What are various resources in system? Having this knowledge can help define and identify roles and privileges required to them. Unified modeling Language can be used to identify these factors in system before head actual development and helping developer design and implement system according requirements while considering them. UML can be extended to determine the various actors, use cases and resources in system and establish relation among them. This can help to implement access control

with ease and without leaving any gap in security of system.

#### REFERENCES

- [1] P. B. Ambhore, B. B. Meshram and V. B. Waghmare, "A Implementation of Object Oriented Database Security," 5th ACIS International Conference on Software Engineering Research, Management & Applications (SERA 2007), Busan, 2007, pp. 359-365.
- [2] Yixin Jiang, Chuang Lin, Hao Yin and Zhangxi Tan, "Security analysis of mandatory access control model," 2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No.04CH37583), The Hague, 2004, pp. 5013-5018 vol.6.
- [3] R. S. Sandhu, E. J. Coyne, H. L. Feinstein and C. E. Youman, "Role-based access control models," in Computer, vol. 29, no. 2, pp. 38-47, Feb. 1996.
- [4] Coyne, Edward J., Timothy R. Weil, and Rick Kuhn. "Role engineering: methods and standards." IT Professional 13.6 (2011): 54-57.
- [5] Stuart Steiner, Daniel Conte de Leon, and Ananth A. Jillepalli. 2018. Hardening web applications using a least privilege DBMS access model. In Proceedings of the Fifth Cybersecurity Symposium (CyberSec '18). ACM, New York, NY, USA, Article 4, 6 pages
- [6] Stuart Steiner, Daniel Conte de Leon, and Ananth A. Jillepalli. 2018. Hardening web applications using a least privilege DBMS access model. In Proceedings of the Fifth Cybersecurity Symposium (CyberSec '18). ACM, New York, NY, USA, Article 4, 6 pages
- [7] J. Jurjens, Secure Systems Development with UML. Berlin, Germany:Springer, 2005.
- [8] T. Lodderstedt, D. Basin, and J. Doser, "SecureUML: A UML-based modeling language for model-driven security," in Proc. 5th Int. Conf. UML Unified Model. Language, 2002
- [9] L. Røstad, "An extended misuse case notation: Including vulnerabilities and the insider threat," in Proc. 12th Working Conf. Requirements Eng.: Found. Softw. Quality, 2006.
- [10] Sylvia Osborn. 1997. Mandatory access control and role-based access control revisited. In Proceedings of the second ACM workshop on Role-based access control (RBAC '97). ACM, New York, NY, USA, 31-40
- [11] <https://www.techopedia.com/definition/4017/mandatory-access-control-mac>