Dependable and Secure TMO Scheme

Jungin Kim  
Department of Computer Science  
University of Texas at Dallas  
jungin.kim@student.utdallas.edu

Bhavani Thuraisingham  
Department of Computer Science  
University of Texas at Dallas  
bhavani.thuraisingham@utdallas.edu

Abstract

In a real-time distributed computing environment, security is critical to protect the system from unauthorized access especially since such systems are being used in time critical applications. Access Control mechanisms have been introduced during the last several decades and have offered a basic and powerful means for enforcing security. In this paper, we examine the concepts of the TMO (Time Triggered Message Triggered Object) scheme that provides guaranteed real-time services in a distributed object computing environment. We also examine access control mechanisms; such as the traditional model, the RBAC (Role-based Access Control) model and the UCON (Usage Control) model. The main contribution of this paper is applying the traditional, RBAC and UCON models to the TMO scheme in order to provide a secure real-time distributed environment.

1. Introduction

Confidentiality, integrity and availability are important security measures. That is, information has to be secured and should be protected from unauthorized modification. Furthermore, information has to be available when needed. Computing environments have been migrating toward the distributed real-time object computing paradigm with the real-time servicing requirements for various practical applications. Many real-time applications, such as military defense systems, intelligent transportation systems, emergency management systems and medical systems, have to meet timing constraints and execute in a distributed environment. However, if security is not properly enforced, information may be exposed to unauthorized users, destroyed or changed in unauthorized ways. This results in information loss, theft and denial of service. Many security issues have been studied and resolved in areas such as databases, operating systems and networks, but security mechanisms in distributed real-time systems are yet to be studied in depth. Some preliminary research has been reported in [1], [2], [3]. However much research needs to be carried out in this important area.

Systems developed and designed without security suffer from leakage of sensitive information, as well as from incorrect information. To achieve a dependable real-time system, we need to enforce security mechanisms such as access control, authentication and identification. Confidential data should not get into the wrong hands, which means we need to incorporate access control mechanisms such as RBAC and UCON into the systems. Also every access request from outside should be authenticated correctly as to who the users are. In this paper, we discuss security for dependable real-time systems and focus on access control issues. We utilize the TMO (time triggered message triggered object) model as the underlying model for real-time computing. TMO which was developed by Kane Kim et al at the University of California at Irvine (UCI) is a useful modeling scheme for real-time systems. In section 2 we disrobe TMO. In Section 3 we discuss authorization models such as RBAC and UCON developed by Ravi Sandhu et al at George Mason University (GMU). We discuss how to incorporate such security schemes into TMO with examples in Section 4. We discuss authentication issues for TMO in Section 5. Section 6 discusses some possible threats such as denial of service in real-time computing systems. Future work such as access control in other distributed systems including, CORBA will be discussed in section 7. Lastly section 8 summaries and concludes the paper.

2. TMO

This section provides an overview of the TMO scheme for guaranteeing real-time service in distributed systems. The TMO object scheme we discuss in this paper was developed at the Dream Laboratory at UCI [4].

The TMO object is defined as a real-time object which has real-time property in itself and this scheme
is an extension of the concept of the existing service object. The distinct difference between an object model and the TMO scheme is that the TMO object model has additional an method, SpM (Spontaneous Method) that can be spontaneously triggered by the defined time values in an object.

The TMO model supports a distributed computing environment. TMO object can be distributed among multiple nodes and communicate with each other using RPC (Remote Procedure Call). To increase concurrency, client nodes of TMO can make non-blocking requests to server nodes of TMO. SpM and SvM (service method) are clearly separated in that an SpM is a method which is triggered by the internal real-time clock. That is, an SpM is a time-triggered method. On the other hand, an SvM is a method which is triggered by the external service request from another TMO. SvM is a message triggered method because this external service request occurs through a message.

Essential components of TMO are the ODS (Object Data Store): Storage of properties and states of the TMO object; EAC (Environment Access Capability): List of gates to objects to providing efficient call-paths to remote object methods, logical communication channels, and I/O device interfaces; AAC (Autonomous Activation Condition): Activation condition for an SpM, which defines the time window for the execution of that SpM; SpM (Spontaneous Method): A time triggered method which runs in real-time periodically; SvM (Service Method): A message triggered method which responds to external service requests.

Figure 1 shows the TMO object structure and the following paragraph describes TMO object name and their roles. An ODS is a common object data store being accessed or shared by multiple SpMs and SvMs, but both methods can not access it simultaneously. When SpM and SvM access the ODS at the same time, the SpM’s priority is higher than that of SvM’s priority. This can be accomplished by BCC (Basic Concurrency Constraints). BCC guarantees the avoidance conflicting of access between SpM and SvM. That is, BCC is a set of rules to prevent potential conflict between SpMs and SvMs. If SpMs and SvMs attempt to access the same object in the ODS, then SvMs are delayed from executing until there is no conflict. That is, SvMs can be run only when there is no conflict between SpMs and SvMs. The TMO model supports time windows, defined by an AAC that is located in the first clause of the SpM in order to guarantee that deadlines are met. The EAC is responsible for the interface call of the communication channels and the I/O devices. The SpM and the SvM are the list of methods, which are clearly separated from the existing object. The TMO object can have several SpMs and SvMs. [4, 5, 6].

3. Access Control

3.1. Overview

This section explains the overview of access control and introduces traditional access control models, RBAC (Role-Based Access Control) and UCON (Usage Control) model. Note that RBAC and UCON have been developed at GMU. Then we discuss how to incorporate the above access control mechanisms for authorization schemes for the TMO model and then explain our design as it can be used in TMO successfully in section 4.

As stated in Section 1, access control is a key mechanism to ensure information confidentiality and integrity and is central to information security. Access control restricts unauthorized users, grants access to authorized users and limits what authorized users can do with the system. However, access control itself is not a complete solution. A system can be secured when access control is combined with other mechanisms such as identification and authentication. Access control is a different mechanism to authentication [12]. Access control limits the usage on the object that is requested from authenticated users or subjects. Access control determines how subject’s access to objects is controlled. On the other hand, authentication is the process of determining who you are while authorization determines what you are allowed to do with the system.
3.2. Mandatory and Discretionary Access Control

Mandatory access control (MAC) [8] is an access policy supported for systems that process sensitive data. Systems providing MAC must assign clearance labels such as top secret, secret, confidential and unclassified to all subjects and all objects in the system. All access decisions are made by the system and statically enforced. Another popular access control system is discretionary access control (DAC) [8]. DAC is an access policy that restricts access to objects based on the identity of users or groups to which they belong.

Access Matrix. The access matrix is a conceptual model that specifies the rights that each subject possesses for each object. However, in a large system the access matrix size will be large and most of cells are likely to be empty. The space requirements of the matrix prohibit the actual use of this model in a computer system.

Access Control List. A popular security mechanism that has its origins in operating systems is the Access Control List (ACL). An ACL is a lower-level mechanism that contains the names of subjects that are authorized to access the object to which it refers, as well as specific permissions that are granted to each authorized subject. It specifies the access rights of a subject on an object. Each object is associated with an ACL, indicating for each subject in the system the accesses that the subject is authorized to execute on the object. This approach corresponds to storing the access matrix by columns. Thus, when a subject wants to access an object, the system has to search for an entry for the subject in the ACL.

3.3. Role-Based Access Control (RBAC)

In modern computing environments, computer systems serve multiple users and require multiple applications. Access control needs to ensure that only authorized users are given access to certain data or resources. To support this paradigm, the RBAC model was developed [9, 11, 12].

Roles mean the positions of the users who are allowed to access resources and the extent to which these resources are accessed. A role can be defined as a set of actions and responsibilities associated with a particular activity. With RBAC, the system administrator creates roles and then assigns users to the role on the basis of their specific responsibilities and qualifications. A user can belong to many roles and a role can have many users. Similarly, a role can have much permission and the same permissions can be assigned to many roles. Figure 2 shows core RBAC elements and relationships between them.

![Figure 2. Core RBAC model](image)

The following defines role authorization. A subject can never have an active role that is not authorized for it.
1. All s: SUBJECTS, r: ROLES, o: OBJECTS, p: PERMISSION
2. r ∈ roles( s ) ∧ s ∈ subject( s ) ∧ s ∈ assigned( r )
3. access: ( s ∈ p ∧ o ) := BOOLEAN

That is, access (subject, permission, object) = 1 if subject s can access object o with permission p, 0 otherwise.

3.4. Usage Control (UCON)

Traditional access control models consider static authorization decisions based on the users permission on the target objects. There is a centralized reference monitor that checks each user’s access. Permission is granted once according to the decision which is determined by policies at the time of the access request; subject can access the object as many times as it wants to.

In the UCON model [10], an access is not a permanent decision, but a process lasting for some duration with related actions. Besides, actions during an access period result in changes to subjects or objects attributes. Usage control model has the following components: subject, object, rights, condition and obligations. The authorization which consists of subjects, objects and access rights is the basic to access control as in the traditional model. Figure 3 illustrates the core UCON model. The UCON model extends the general model of access control by including obligations and conditions. Obligations are actions that are performed by subjects or users before obtaining access rights. Conditions are system or environmental restrictions such as system clock, system location, system mode and so on.

The distinguishing aspects of UCON compared to traditional models are continuity of access decision and mutability of subject and object attributes. There are three phases to complete access decisions: before usage, ongoing usage and after usage. In UCON, access decision can be checked repeatedly during the access and revoked if some conditions are not satisfied or changed. The mutability and continuity of attributes make the UCON model very powerful and provide a seamless security administration policy. Also, UCON...
model is useful to specify dynamic constraints and consumable access increasing or decreasing access time. Access decision is not a single function for subjects or objects, they interact with each other depending on access from others.

![UCON model](image)

**Figure 3. UCON model**

### 4. Access Control for TMO

#### 4.1. Overview

Access control provides assurance that the TMO object accessing the other TMO objects has permission to access the related information. Different access control models have been developed to ensure that only authorized users who have proper permission can access resources. From the TMO perspective, we wish to ensure that TMO objects are, for example, authorized and hold proper permission to execute the other object, access the data of the other object and make changes to the TMO object.

These are the essential mechanisms for security, and to prevent the breach of information should be enforced at all the component levels of the TMO scheme. The goal of our work is to ensure security guarantees for entire objects through global environment rather than for the individual objects. We first discuss MAC and DAC for TMO. This is what we have called the traditional access control model. Next we discuss RBAC in detail for TMO. Finally we briefly discuss UCON for TMO.

#### 4.2. MAC and DAC for TMO

MAC mechanism controls access on the basis of classification of subject and object in the system. With MAC, each TMO object can be assigned to be either a subject or an object and a label called a security clearance such as Unclassified, Confidential, Secret and Top secret is also associated with the TMO. Based on the subject security clearance, access to an object by a subject is granted. As an example, TMO1 with a clearance level of secret could not access TMO2 with top secret label for the read operation. Some of these features can be useful in rigid environments. Also MAC can use a hierarchical scheme, which would be read down and write up [9]. This would be needed to prevent information leakage.

With DAC, TMO object can be viewed as owner based administration of access rights. In DAC, the owner of an object has discretionary authority over which TMO object can access the other TMO objects. Capabilities are accommodated in the TMO scheme. It is natural for a TMO object with these capabilities to enforce DAC efficiently. In a capability based system, access to protected objects is granted to the would-be accessing subjects possessing a capability for the object. The advantage of capabilities is that it is easy to review all accesses that are authorized for a given subject but difficult to review subjects that can access an object and revoke access to an object.

With ACL, it is straightforward to determine which modes of access the subjects are currently authorized for that object. Also it is straightforward to revoke all accesses to an object by replacing the existing ACL with an empty one. Another way to make this efficient compared to an access matrix is that the control lists need not be large, if a group of subjects with common access to the object are attached to the object. This group management was proposed in the TMO Object group model [7]. The creation and management of groups should be strictly controlled, such as being a member of a group and accessing objects by members.

A group is often used as an entry on the ACL for management to describe a collection of individual subjects. We may consider that a role is equivalent to a group. A role can represent a collection of users, and a user can be a member of multiple roles. However, roles and groups have different semantics in an access control model and different usage in their environment because groups are implementation-specific. The properties of a group can be changed from one implementation to another.

Therefore, from the distributed environment perspective, managing user permissions through global roles is more efficient than through the individual groups of many operating systems and applications. Central to RBAC is the concept of role relations. In addition to user and permission assignment relations,
the RBAC model includes user and permission inheritance relations and various static and dynamic constraint relations. We discuss RBAC in the next section.

4.3. RBAC for TMO

Permissions, in the basic RBAC model, to resources are granted to roles and then these roles are assigned to users who need permission. Roles typically represent organizational roles in real-life and roles are assigned to the users who need permission according to their roles. This model is implemented in many commercial DBMSs and operating systems and we can figure out the relationship in these systems intuitively. In the TMO scheme, each TMO object can be a subject and object at the same time and it is assigned to a specific role and permission is granted to roles. For example, with the CAMIN application environment described in the TMO papers [4], roles are assigned implicitly such as Theater, command post and alien at design time of the application. Theater may have read and write access to the object data store and invoke SvM for all other objects in entire system like command posts, and command posts may have read rights to Theater but not have one to change data within it. Some objects may have a right to execute other objects but no access rights to the resources itself. However, we can assign specific roles to each TMO for the entire system, and not individually. For example, Theater can be an administrator in the CAMIN application, each command post in land and sea can be a user and Alien can be a guest or have some other role.

Roles and their permissions may be changed occasionally over time or not at all, whereas subjects — that is, objects in the TMO scheme — may change regularly. Thus granting permission to the role is more efficient than granting it to each user. Also removing permission can be convenient. All of these operations can be done without impact on the application. The RBAC model would give simplicity to its implementation, thus it needs initial configurations in application or we can invent an administration tool to manage roles for the system, which can give more advantages to the RBAC in its maintenance and scalability. To accomplish our goal, and that is to develop a secure TMO, we need to incorporate this RBAC into the TMO to make it a dependable system. Figure 4 illustrates enforcing RBAC on TOM objects.

Whenever each SvM, which is a passive method, is invoked by SpM in another TMO, it checks the id and matches the role of the incoming TMO object and makes a decision whether the access can be allowed or not. So the authorization process needs to be carried out before further actions on an object at the entry to the SvM method.

There are application interface classes, basic role classes and each user defined role classes. The conditions for each role can be defined in each role class. Applications and role classes can interact with each other through the application interface class. The only thing needed in application is to call this interface with the conveying TMO id as a parameter, and then the interface class returns true or false to the application. The goal of an authorization is a decision for a “yes” or “no” answer as to whether a subject can get the permission to the object. Each role object and the permission to resources need to be defined at design time. Figure 5 illustrates this approach.

Data in ODSS have the permissions. After a decision for the request, SvM performs corresponding actions to the data based on the permission or privilege in ODSS. These policies such as permission or privilege are based on the actions that a subject is allowed to perform within the context of the entire system. Also actions can be defined in each role class, but they may depend on applications.
One of the merits of RBAC is the concept of operation, which may be able to do anything depending on the application in which it is implemented. This operation can be a complex instruction. For example, Command post can only update data to the Theater from 10 AM to 6 PM or the Theater can allow access to itself from another object during a specific time period. (This idea of operations is expanded in the UCON model. We will discuss UCON for TMO briefly in the next section.) However, the operations may be a fixed set such as read, write, update and invoke in the TMO scheme. The main purpose of RBAC is to associate operations with roles without impact of the behavior of applications.

This implementation of RBAC in TMO can be carried out smoothly. Furthermore, RBAC is also quite suitable to the key concepts of TMO, which are spontaneous behaviors. However, in a distributed system, centralized access control mechanisms may not be adequate because information is distributed and can be accessed by various systems. Thus, we need to consider access control from both the client and the server side. The traditional RBAC model is a server side mechanism while UCON is the client and server side mechanism. We discuss UCON in next section.

4.4. UCON for TMO

The UCON model encompasses the traditional model (e.g., MAC/DAC) and RBAC [9]. Security level of MAC and DAC and role of RBAC is adapted in UCON using attributes which are attached to subjects and objects. The basic authorization components of UCON and its mutability and continuity can be easily adapted for access control to the TMO scheme. Mutability aspect of UCON can be used to control the scope of accessing TMO object by increasing or decreasing mutable attributes such as access time, access range and access count.

The condition aspect of UCON also can be useful for TMO by means of restricting the access. TMO object can give permission or be denied during a particular time period, which is an extension from the idea of operations in the RBAC model. This feature may be used within SpM to restrict its behaviors or access from others. For example, SpM can block the access from outside by changing its attributes or announce to all other objects when the system goes into maintenance or detect any abnormal status. Condition also can control resource usage to limit the number of access or scope. Through the entire system life or specific time period, conditions can be varied and security policies can be accommodated flexibly and dynamically to the TMO scheme. This aspect of UCON is very beneficial and efficient for the TMO scheme when it is correlated with mutability.

The concept of obligations also can give a useful way for access decisions. Obligations are active actions required to be performed on the subject side before accessing the object, so it can be used to screen any TMO object before the subject acquires the access permission. Global timing synchronization can be one example of obligation. To communicate with each other, all TMO objects must have the same global time to prevent abnormal behavior among TMO objects or to guarantee timing assurance. The attributes of an object also can be controlled or referenced by obligation aspect as well. If an object does not allow using its resources associated with the attributes by other objects with respect to obligation, then access may be denied because it may allude to malicious objects. To accomplish the above features, TMO classes need to be extended to add attributes and methods for conditions, obligations and mutability of the UCON model. We may need to devise various methods to handle the components of the UCON model for TMO applications. UCON for TMO will be elaborated in a future paper.

5. Authentication for TMO

Authentication is the process of determining who you are, while authorization determines what you are allowed to do. We can design a dependable TMO when both of these features are combined together. From the TMO perspective, authentication mechanisms ensure that only legitimate external objects can gain access to the object that they wish to access. Poor authentication would allow malicious objects to gain access to the system and disclose critical and sensitive information. Traditionally, authentication systems are based on a username and password to access computer systems through networks; however, it is a weak authentication policy, moreover it is not applicable in the distributed environment of TMO.

In a distributed system, users are not necessarily registered at the node they are accessing an object. The question here is, how to prove each accessing objects’ identity for each service and how objects prove their identity to subjects even though the TMO gate and RMMC provide an interface between the TMO objects? Let us assume we use password authentication in TMO. Every object needs to store the password for the objects it accesses. Also, it may have different passwords for TMO objects. An individual object having multiple passwords is not practical. It would be better that we use shared secure key or public key for the TMO environment. Also, any authenticated
message like a password if applicable to the data should be delivered in a secure way through the network. We need to consider data encryption and distribution in TMO along with authentication mechanisms to determine whether the received key is correct. Unfortunately, however, it seems that there is no way to solve the above problems in the TMO scheme at this moment. Therefore, we may need stronger authentication policies and mechanisms for TMO apart from those for the operating systems on which TMO applications run. This presents a significant challenge to us.

6. Possible Threats to the TMO System

There are many threats in computer systems such as information compromise, integrity violation by virus, denial of service (DoS), eavesdropping (obtaining copies of messages without authority), masquerading (using the identity of another principal without authority), tampering (altering the contents of messages), replaying (storing intercepted messages and sending them at a later time) [15]. Among them, the most common and critical method of attack today is a denial of service (DoS) and distributed denial of service (DDoS) attacks. We need techniques to detect such attacks and also develop solutions for defense and tracing [14]. The purpose of a DOS attack is to deny the users access to resources by means of destruction of information, resource consumption and network bandwidth consumption. Basically the TMO applications run on conventional operating systems such as Windows family and Linux, which have lots of vulnerabilities and are threatened by malicious attackers continuously. Thus the basic threats to the TMO can be any existing attacks. TMO application itself may have unauthorized information access problem without proper security mechanisms.

The TMO scheme has several security vulnerabilities as follows: Network communication is vulnerable to interception and tamper. Attackers can tamper physically replacing or damaging nodes spread far apart. User authentication of the distributed system is performed via network transmission of authentication messages, thus malicious attackers can masquerade as the user simply by eavesdropping the authentication message, consequently an existing connection between two endpoints can be disrupted. An Internet hacker also can gain access rights to the TMO applications acting as a server and by IP sniffing or as a Trojan horse on the client object to capture the key or password and obtain sensitive information. Since the system is distributed, inconsistency may exist among different nodes where TMO objects reside, which can give intruders the opportunity to compromise security.

As their announcement propagates among TMO objects, the network routes more traffic in their direction. This can form a black hole within the network. Even an authorized user can masquerade as someone else, and therefore can obtain the access to whatever that user is authorized to do. In a distributed system, a user may delegate his rights to other objects, so they can act on their behalf. This adds the threat of rights being delegated too widely, again causing the threat of unauthorized access.

Note that malicious code in the systems can make the TMO miss the deadlines. This is a very serious problem. One possible solution is to ensure that only trusted code can modify the timing constraints specified in the TMO system. When designing secure systems, the objective is to keep the trusted portions of the system as small as possible. Therefore we need to determine exactly which parts of TMO have to be trusted [1].

Another problem with developing secure real-time systems is that access control checks for MAC, DAC, RABC and UCON are time consuming operations. This could cause the deadlines to be missed. This is also a serious problem. However violating the access control rules is a security problem. Essentially we need flexible policies for time critical applications [1].

7. Summary and Conclusion

Various modern real-time systems including military defense systems, intelligent transportation systems, emergency management systems and medical systems need to access, store, transact, manipulate, and communicate sensitive information requiring a timeliness services. Therefore ensuring security is a serious concern in the design of such systems. We have discussed the TMO scheme, which is a useful real-time system model, and discussed some security models for TMO. Specifically we have examined MAC, DAC, RBAC and UCON for TMO.

Traditional access control mechanisms provide a general approach for implementing access control in computing systems for several decades, but they are inadequate for distributed environments. Therefore, we reviewed the RBAC and UCON models and examined them for TMO. Both RBAC and UCON are powerful models. We showed with examples how they can be applied for TMO. Access control mechanisms along with authentications are strongly required in TMO scheme to make it a dependable system. Next we discussed some possible threats for TMO such as DDoS. Finally we discussed some directions such as
CORBA and its access control policies as a reference model.

We proposed a security design of RBAC and UCON model for TMO by incorporating them into the TMO scheme. However, access control and authentications by themselves cannot be the basis for strong security solutions for TMO. They should be combined together to provide a sound structure for the secure TMO scheme. We need further investigation to apply access control and authentication effectively for TMO applications that will take advantage of all the features provided by TMO. We believe that implementing such features will increase the dependability and ensure information security for the TMO scheme. Some future directions of dependable objects are to examine some other models such as CORBA (Common Object Request Broker Architecture) [13]. Our preliminary investigation of this effort is reported in [16].

8. References


