Design of Secure CAMIN Application System based on Dependable and Secure TMO and RT-UCON

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

Increasingly the need for protecting information from unauthorized access has lead to more attention in the field of information security. Access control mechanisms have been in place for the last four decades and are a powerful tool utilized to ensure security. In a real-time distributed computing environment, systems have to meet timing constraints for time sensitive applications, such as obtaining financial quotes and operating command and control systems. However, real-time distributed computing environment security is yet to be investigated. Earlier work examined a Time Triggered Message Triggered Object (TMO) scheme that provides real-time services in a distributed computing environment secured by applying a Role-Based Access Control (RBAC). In this paper we describe the application of a sophisticated Usage Control (UCON) model for a TMO and subsequently design an application that utilizes a Coordinated Anti-Missile Interceptor Network (CAMIN) system. This application is a secure CAMIN based upon a secure TMO that applies a Real Time UCON (RT-UCON) in the CAMIN environment.

1. Introduction

Real-time applications have been developed in numerous fields such as multimedia, industrial automation, and military systems. Advancements in computing power have led computing environments to migrate toward the idea of a distributed computing paradigm from that of the grid paradigm as application have become more distributed. These real-time applications have to meet timing constraints along with other constraints such as, flexibility, accuracy, and security. For example, a command and control system such as an Air Borne Warning and Control System (AWACS) has to meet timing constraints when processing the track data and also ensure that the data is accessed for querying and updating by authorized processes [1].

Security controls are utilized to prevent unauthorized individuals from gaining access to the information. Important security requirements include confidentiality, integrity, and availability as well as privacy, fault tolerance and trust. These requirements must be balanced with the fact that information does need to be accessed and cannot be locked away. However, one must be mindful of the fact that applications without proper security controls may have disastrous results. For example, a malicious process could ensure that the real-time processing component of an AWACS misses its deadlines which could cause a catastrophe. Many security issues have been studied in computer systems and applications such as databases, operating systems, and networks, but security mechanisms in distributed real-time systems have yet to be studied in depth. Some research has been conducted in [1], [2], and [3], however much research remains to be carried out.

Current techniques are inadequate for building applications satisfying both real-time, dependability, and integrity and security requirements. The University of California at Irvine’s (UCI) DREAM Laboratory has developed an ideal scheme for a real-time object structure called a Time Triggered Message Triggered Object (TMO) that represents temporal behavior. However, a TMO does not include sufficient security features and therefore is not suitable in its current form for many command and control applications. Nevertheless, the flexibility and evolvability of a TMO makes it an ideal candidate for incorporating security. Therefore, our research has focused on incorporating security for the TMO scheme.

In our previous paper, we described the initial design of a Secure and Dependable TMO [4]. In that effort we applied a Role-Based Access Control (RBAC) as the authorization mechanism for a TMO scheme and briefly mentioned the Usage Control
(UCON) model for future research. In this paper, we focus on UCON as an access control model and discuss an application system design to demonstrate a dependable real-time system. In this effort, we utilize the CAMIN application system [5] which was developed at UCI’s DREAM laboratory in a simulator for real-time systems.

The organization of this paper is as follows. In section 2 we review both the TMO scheme and our previous work on developing a secure TMO. Next in section 3, we discuss a real-time UCON (RT-UCON) which is an extension of the UCON model developed by Ravi Sandhu et al at George Mason University for real-time applications. We also describe in detail our design of a secure CAMIN which is based on the basic CAMIN described in section 3. This also includes a description of the application interfaces implementing the RT-UCON mechanism for CAMIN. In section 4 we summarize this paper and discuss future directions.

2. Background

2.1. TMO

A TMO object is a real-time object which is a natural, syntactically minor, and semantically powerful extension of the conventional object [5, 6, 7]. Generally there are two different methods that are employed in a TMO. The Spontaneous Method (SpM) which is triggered by the defined time values in an object and the Service Method (SvM) that is triggered by an external service request from another method. The key difference between the SpM and the SvM is the fact that the SpM is triggered by the internal real-time clock while the SvM does not. Figure 1 illustrates the TMO object structure.

Essential Components of the TMO Scheme:
- ODS (Object Data Store): Storage of the properties and states of the TMO object.
- EAC (Environment Access Capability): List of gates to objects that provide 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. TMO object (Taken from [5])

2.2 RT-RBAC

Access control is a key mechanism central to information security. Access control restricts unauthorized users, grants access to authorized users, and limits what authorized users can do with objects they access. Many access control mechanisms have been developed such as Mandatory Access Control (MAC), Discretionary Access Control (DAC) and Role-Based Access Control (RBAC) [8,9]. RBAC, which is currently the industry standard implemented in many commercial databases and operating systems, was invented to support multiple users and applications and consists of roles and actions. Access to resources in the basic RBAC model is granted through roles which are then assigned to users who need permission. Roles typically represent organizational positions in real-life and are assigned to users who need permission in order to gain access to the resources.

In the TMO scheme, each TMO object can be a subject and an object. It is assigned to a specific role and permission is granted only to certain roles. With these ideas, we proposed a dependable and secure real-time system with RBAC [4]. This implementation of
RBAC in a TMO can be carried out effectively, since we can think intuitively that every TMO object is equivalent to the subject and each subject can be assigned roles. Furthermore, RBAC is also suitable to be enforced with the main concept of a TMO, which is spontaneous behavior. 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 or a set of fixed operations that invoke the TMO scheme.

Traditional RBAC is a sever side centralized model and in a distributed system this may not be adequate because information can be accessed by multiple systems. An additional problem is that real-time systems must not violate timing constraints that are essential to its operation. To avoid the latter problem, we can use the RBAC model [10] however in doing these role assignments and temporal behaviors are separated. To rectify this problem, the inventors of RBAC developed the UCON model which already includes the temporal feature as one of its core components in light of this fact. Thus allowing it to apply access rights as an attribute of an object and define temporal behaviors with a condition in the UCON model. After having done this we can consider an access control mechanism which has a temporal behavior from both the client and the server side. As a result of this insight, we have chosen to use UCON for our real-time model and have named the resulting model RT-UCON.

RT-UCON can satisfy the essential features of access control and provide more powerful and flexible mechanisms describing timing conditions, continuity, and mutability which are the core components of UCON. We now discuss the RT-UCON that we have designed in section 3 together with our design of secure CAMIN.

3. Secure CAMIN

A secure CAMIN requires both RT-UCON and the CAMIN, therefore, we will discuss the underpinnings of RT-UCON and the CAMIN first before we discuss the secure CAMIN.

3.1 RT-UCON

Traditional access control models consider static authorization decisions based on the users level of authorization on the target objects. A centralized reference monitor is utilized that checks each user’s access and permission is later granted according to the policies that are being implemented at the time of the access request.

The UCON model utilizes several parts of the traditional models, such as the security level classifications of MAC and DAC and the role authorizations of RBAC [11]. These are adapted in the UCON model using attributes which are attached to subjects and objects. Figure 2 illustrates the core UCON model. In this model, access is not a permanent decision, but a process lasting for some duration with related actions. Furthermore, during an access period actions can result in changes to subjects or objects attributes. The UCON model has the following components: subject, object, rights, condition, and obligations. The basis for authorization consists of subjects, objects, and access rights which arise from the traditional models. The UCON model extends the traditional models’ methods 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 a system clock, system location, and system mode.

![Figure 2. UCON model](image-url)

The UCON model when compared to the traditional models is distinguished by having continuity of access decisions, mutability of subjects, and object attributes. There are three phases for an access decision in the UCON model; before usage, ongoing usage, and after usage. In the UCON model, access decisions can be checked repeatedly while information is being accessed and revoked if some conditions are not satisfied or changed. The mutability and continuity of attributes makes the UCON model very powerful and provides a seamless security administration policy. Also, the UCON model is useful to specify dynamic constraints and change the amount of access a user has, increasing or decreasing access time. Thus, an access decision is not a single function for subjects or objects but rather several ongoing decisions made while user is attempting to obtain information.
These basic authorization components of UCON and its characteristics of mutability and continuity can be easily adapted for access control in the TMO scheme. The mutability aspect of UCON can be used to control the scope of accessing TMO object by increasing or decreasing mutable attributes.

The conditional aspect of UCON can also be useful for a TMO by restricting access. A TMO object can give permission or be denied permission during a particular time period, which is an extension from the idea of operations found in the RBAC model. This feature may be used within SpM to restrict behaviors or access from others. For example, SpM can block the access from outside by changing attributes, detecting any abnormal status, or announcing to all other TMOs that the system has gone into maintenance. A condition can also control resource usage to limit the number of time access is allowed or limit the scope of authorization. Through the entire system life or specific time period, conditions can be varied and security policies can be implemented flexibly and dynamically for the TMO scheme. This aspect of UCON is very beneficial and efficient for the TMO scheme when it is combined with mutability. Thus, access rights are included in the TMO object, but conditions like temporal constraints are implemented with an Application Program Interface (API) and are called by objects.

The concept of obligations can also give a useful way for determining access decisions in the TMO scheme. Obligations are active actions that are required to be performed on the subject side before access to the object is allowed. Thus, it can be used to screen any TMO object before the subject acquires access permission. Global timing synchronization is one example of an obligation. To communicate with each other, all TMO objects must have the same global time to prevent abnormal behavior and to guarantee timing assurance. The attributes of an object can also be controlled or referenced by obligation aspects as well. If an object is not allowed to use its resources associated with the attributes given to it by other objects with respect to that obligation, then access may be denied because it may allude to malicious objects. To accomplish this TMO classes need to be extended to add the attributes and methods for conditions, obligations, and mutability of the UCON model. Further, we may need to devise various methods to handle the components of the UCON model for TMO applications. We discuss them in section 3.4 and 3.5.

3.2 CAMIN

The CAMIN [5] is a simulator used to illustrate the idea of a TMO scheme. It shows the real-time environment and requirement specification through multiple design processes to develop and test a real-time application. The application scenario of a CAMIN is as follows:

The environment has three parts, sky, land, and sea. There are flying objects which are either threatening and non-threatening and target objects in the sea which are moving around to avoid being destroyed by the hostile flying objects. Also, there are target objects on land, however they cannot move around. Our mission is to defend target objects both in the sea and on the land from the hostile objects in the sky. Figure 3 illustrates the CAMIN application environment.

![CAMIN application environment](Taken from [5])

There are four main objects in the CAMIN as shown in Figure 4. The Theater TMO is an application environment, which describes sensors and interceptor launchers in land and keeps all the information from other objects such as command post on land, command ships in sea, and all other flying objects. This information is kept in Object Data Storage (ODS) of the Theater and updated periodically by SpM in the Theater TMO. Besides the Theater, CAMIN has two more components, enemy and natural forces, which are clients for Theater and represented by an Alien TMO. The Alien TMO introduces hostile reentry vehicles and non-threatening flying objects into the Theater. The command post TMO and command ship TMO are the two other main objects in CAMIN. Both of them have a control system which consists of Radar Data Queue (RDQ) to store radar data received, Flying Object Tracking (FOT) to store information needed for tracking flying objects, and Intercept Plan Data Structure (IPDS) to store information needed to control the interceptor launcher. Every object corresponds to a
TMO object and the whole Theater is a network of all TMO objects.

The following is a simple CAMIN scenario: The spot check result generated by the FOT goes to the RDQ as radar data and the RDQ checks the recent result to determine where to send the data. Then the FOT sends a copy of each radar request to the RDQ. There are two SvMs in the RDQ. One receives information on flying objects in the Theater from radar and the other receives copies of spot check radar requests from FOT. Also, there is an SpM which periodically sends radar data along with the ID numbers of the requests to FOT. FOT analyzes the radar return data and determines whether or not the detected flying object is dangerous. If not, FOT simply tracks it for a short time and then forgets about it. If the flying object is considered dangerous, FOT asks the IPDS to generate a schedule for intercepting the flying object and to destroy it. Major FOT computations are handled by SpM, whereas SvMs are designed mainly to receive information and deposit it in the appropriate object data store segments. When it receives an intercept order, the IPDS calculates the interceptor launching time and the direction in which the interceptor should be launched. This information is sent either to a fighter airplane or land-based launcher. In general, fighter airplanes are first ordered to fire; land-based interceptors are only fired if the air launched intercept fails or is likely to fail. If the ground-launched intercept attempt fails or if a flying object bypasses the zone interceptable by ground launched interceptors and moves toward the command ship, the Command Post TMO passes the observation data to the Command Ship TMO.

![Figure 4. Specification of main components (Taken from [5])](image)

### 3.3 Secure CAMIN

We have discussed RT-UCON as an access control mechanism and the TMO application of a CAMIN in sections 3.1 and 3.2. In this section we discuss our secure CAMIN application system. If there is no security mechanism in the CAMIN, we cannot guarantee whether or not the information in a TMO objects is secure. Some TMO objects may have incorrect data and some malicious objects may send wrong information to the other objects to distract the system. For example, if there is a hostile object on land and it sends corrupt information to the radar or to the land launcher we may miss the hostile object or may not detect the enemy in the sky. In that situation we would not be able to defend target objects and consequently would fail in our mission. Stated another way, we may encounter a dangerous situation without the proper security mechanisms in place. The goal is to defend the target objects safely so that we ensure all information is correct and only authorized objects communicate according to the access control policies.

We designed the simulation model which we call Secure CAMIN based on the CAMIN and RT-UCON. Figure 5 illustrates the Secure CAMIN environment which consists of the basic CAMIN application and additional objects that are malicious trying to access the main components such as the command post or land launchers. We apply security mechanisms in the
RT-UCON to the basic CAMIN environment. Application of the Secure CAMIN is as follows. When malicious objects access any of the main TMO objects, these TMO objects check whether they are legitimate objects and decide their access based on the authorization policies and security levels. To test this idea we add several TMO objects to the scenario, such as malicious command post and radar and a couple of additional land launchers and fighters. Figure 5 shows the extended secure CAMIN environment.

Figure 5. Secure CAMIN environments

A malicious command post on land will send the wrong information to the land launcher and fighters in sky. But the land launcher and fighter should know whether the data is correct or not when it checks the authorization information. If the information coming from the outside object is incorrect after checking the access control from the malicious object, then they reject access and do not accept that information. Also, command posts may receive incorrect data from malicious radar stations on land, but the command post can detect this by checking the authorization rules. In addition, we can assume there are several land launchers on land and that they are switched off by some events such as maintenance, replacement, or accidents. In such a situation, these objects need to block access from the objects attempting to gain access during a specific time period and announce it to the command post as correct information and to stop the land launchers from receiving the message. If hostile objects do get that information, they may abuse which may result in a catastrophic situation. We can decide these timing conditions in the design phase, but for flexibility the land launcher needs to publish the information when the command post attempts to accesses the information. This can be implemented by continuity and mutability of RT-UCON along with timing constraints. Also, objects may extended from the command center such as FOT, RDQ and IPDS, and these must have access constraints because they are the essential objects in the CAMIN application.

Whenever an object receives an access decision request from the outside, it asks the access control TMO object to decide the access rights. We will introduce the access control TMO object in next section. This object is an essential aspect for the secure CAMIN application. Therefore, the point of a secure CAMIN is that every object in the application environment contains access rights that are used when it accesses the other objects. The other side checks the authorization rules and then decides whether to allow the access or not and some objects use temporal constraints to limit access during specific time period. Utilizing the design above we can ensure security of the CAMIN system since we restrict access according to the policies. To implement this idea we must also utilize the access control TMO object and application interface which is discussed in section 3.4.

3.4 Access control TMO object

The access control process is implemented with through a separated TMO object. The main role of this object is to check access right and maintain access policies in the system. Each TMO object makes an access requests to access the control TMO object which is placed at the same node so that server object may accesses it directly. Figure 6 illustrates the structure of an extended object for access control.

![Access Control TMO Diagram](image)

Figure 6. The structure of access control TMO

The TMO object for access control can be included inside each TMO object but a more efficient implementation is to utilize a separate TMO object for convenience, flexibility and easy to design and maintain. Also, there is no network communication between a general TMO object and an access control TMO object because they are placed at the same node.
This ensures that there is no packet loss and communication delay between them.

However, the access decision process needs to be implemented in the server TMO object if the timing constraints of the application are critical. In this case, the separate access decision TMO object may not be used in critical real-time applications. If the server TMO object accesses the access decision TMO object, the result is a slower communication time due to the extra step that is being used in the process. This kind of process may result in missing a deadline. However, real life trade-offs have to be made between the speed at which the process work and the level of security that the system operates with. Sometimes a more flexible and easier design may be used that is able to maintain and implement the access decision in another object. Therefore, the developer should decide whether to use the access decision TMO object as a separate object from the server object or use it in the same object.

The access decision TMO object refers to the access policies in ODS used to decide if access should be granted when the server object asks to access information for the client object as illustrated in figure 8. The access policy contains appropriate access constraints for each client object and the access decision TMO object consists of three members in a general TMO application object: ODS, SpM, and SvM.

- ODS: It is used to store only access policies such as access level and to allow client lists. The access policy in this ODS utilizes SpM and SvM to decide the access level of the client object and is maintained by the SpM periodically. Additionally, there are static and dynamic access policies. Which dynamic policy is used is determined the static policies that are implemented in the SpM and the static policies should not be changed during the life of the application.
- SpM: It maintains the access policies in ODS. Access policies stored in ODS can be changed according to the environmental conditions which are the static access control policies maintained in the SpM. Also the SpM in this object can change the ODS data of the server object for any access conditions when an access decision TMO object is implemented separately.
- SvM: It handles access decision request of the client object. The client TMO application object should get an access decision from this method and SvM returns that access decision result based on access policies in the ODS. However, this can be an optional method if access control is implemented in the server object.

An access decision TMO object provides the functions that support access control.

Figure 7 shows the overview of the TMO application model after implementing the access decision TMO object.

![Figure 7. Basic structure of Access decision](image)

### 3.5. Application Interfaces

To develop secure applications using the TMO scheme, we need new application interfaces to implement the access policies and to help developers and to support the functions explained in the previous sections. We discuss them in detail in this section.

**Access decision process**

Basically access of a subject to an object is allowed at the server side and the server object (SvM in TMO application) checks the access permission based on the access policies and decides to grant or deny access. Each object in the TMO application has a security descriptor and access policies that can be enforced by the application. The basic access allowing function returns a Boolean value of TRUE or FALSE to indicate whether or not the request was allowed or not.

The access decision functions are as follows:

```c
bool Access_Control (char AccessID; int AccessType; in name, in attributes list, out TRUE or FALSE);
```

Both client objects (SpM in TMO application) and server objects have the access type, access rights, and other information as an attribute of that object because they are associated with the attributes in the subject and an object in the UCON model. Thus, we can specify these characteristics in an object: int Rights; char AccessID; int AccessType;

To implement these functions, we need a separated access decision TMO object and also need to send the client TMO name and attributes optionally to the class. Because the TMO uses the global symbolic name, the server object can check access permission and decide the access. Then the server object asks access decision.
TMO object at the entry when client object accesses it with the name and attributes associated with client object. To accomplish these operations, both the SpM and the SvM in a current TMO object needs to be modified to process access decisions. The SpM in the client object should send its name and attributes when it accesses the server object. The SvM in the server object communicates with the access decision object to determine access control. The server object utilizes the following code to get the proper rights and to decide access permission. Figure 8 shows the access decision process between TMO objects.

```c
void get_rights(
    in object_name,
    in object_attributes_list,
    out rights = appropriate rights (read | write
| execute | NULL),
);
```

![Access Decision process](image)

**Figure 8. Access Decision process**

Temporal constraints and environmental conditions

The RT-UCON model has useful functions for access decisions which are not performed once but performed several times during continuous activities. Also, the environmental conditions can be changed over time in the system allowing access policies to be applied along with these constraints. This can be done through the utilization of the SpM access decision object. It sets the value of ODS and changes the environmental condition based on the access policy. The SpM in an access decision process can change the properties in the ODS for access constraints based on access policies as explained in the previous section. We need to design and implement this functionality in the future. For example, set_access_time (to restrict the access time), resume_access, block_access, and set_access_count among others have to be implemented. Basically, the environmental conditions are different for each application. Therefore, to implement continuity and mutability features of the RT-UCON they should be designed in the design phase.

4. Summary and Directions

We have designed a model called the RT-UCON for the development of dependable and secure real-time applications based on the TMO scheme as the access control mechanism. This application consists of the basic CAMIN and the RT-UCON model. The CAMIN application needs to ensure that all the information delivered and exchanged between objects is correct and secure because it contains sensitive information which requires accurate and timely transmission. Thus, ensuring security is an important issue in the design of such a system. We have designed both RT-RBAC and RT-UCON for the TMO to incorporate access control models into the CAMIN application. We believe that RT-UCON provides a more powerful and flexible model. It was shown how this model can be applied to a TMO scheme in our example. However, further investigation is needed to apply RT-UCON effectively for a CAMIN application that will take advantage of all the features provided by RT-UCON and timing features provided by CAMIN. We believe that implementing a secure CAMIN will give us confidence in the dependability security of a real-time system.

There are many opportunities for further research to simulate the idea of secure real-time application. We have discussed RT-UCON and design of secure CAMIN in previous sections. We also discussed the structure of access decision TMO objects and their interfaces. For future work, we need to design the security APIs and to subsequently implement a secure CAMIN application. Real-time applications must meet timing constraints which requires us to verify and validate that our design satisfies both the timing constraints and the security constraints at the same time. Further, it is understood that there may be a trade-off between timing and security. Thus, to satisfy all requirements, we need to consider flexible policies when adding security mechanisms into the real-time TMO applications.

**ACKNOWLEDGEMENTS:** We thank Prof. Kane Kim and his TMO Team at the University of California, Irvine, for their help, support and encouragement of our Secure TMO research. We especially thank Mr. Moon C. Kim from the CAMIN project for the information he provided to us. We also thank the Texas Enterprise Funds and Prof. Robert Helms, the Dean of Erik Jonsson School of Engineering and Computer Science for supporting this research.

5. References

Transactions on Knowledge and Data Engineering, January 1999.


