Access Control Models for Virtual Object Communication in Cloud-Enabled IoT

Asma Alshehri and Ravi Sandhu
Institute for Cyber Security & Department of Computer Science
University of Texas at San Antonio, One UTSA Circle, San Antonio, TX 78249, USA
Email: nmt366@my.utsa.edu, ravi.sandhu@utsa.edu

***Abstract*—The Internet of Things (IoT) is the latest evolution of the Internet, encompassing an enormous number of connected physical "things." The access-control oriented (ACO) architecture was recently proposed for cloud-enabled IoT, with virtual objects (VOs) and cloud services in the middle layers. A central aspect of ACO is to control communication among VOs. This paper develops operational and administrative access control models for this purpose, assuming topic-based publish-subscribe interaction among VOs. Operational models are developed using (i) access control lists for topics and capabilities for virtual objects and (ii) attribute-based access control, and it is argued that role-based access control is not suitable for this purpose. Administrative models for these two operational models are developed using (i) access control lists, (ii) role-based access control, and (iii) attribute-based access control. A use case illustrates the details of these access control models for VO communication, and their differences. An assessment of these models with respect to security and privacy preserving objectives of IoT is also provided.**

***Keywords*-Security; Access Control; Internet of Things; Devices; Virtual Objects; ACL; RBAC: ABAC;**

I. Introduction

The concept of the Internet of things (IoT) originated from the evolution of wireless communication systems over the last few decades. The technologies of sensing, networking, software architectures, information management, data analytics, and visualization all converge in IoT. IoT gives rise to new security challenges that call for a significant revision of current security solutions, including access control systems. In prior work we have developed an access-control oriented architecture (ACO) [1] for cloud-enabled IoT, comprising four layers: an object layer, a virtual object (VO) layer, a cloud services layer, and an application layer (see Section II-C). ACO recognizes the need for communication control within each layer and across adjacent layers, coupled with the need for data access control at the cloud services and application layers. In this paper we focus on developing access control models for VO communications, within the developed ACO framework.

Virtual objects can communicate in various ways. The current common method is topic-based publish-subscribe (see Section II-B). For instance, a virtual object is called a device shadow in Amazon Web Service (AWS) IoT. The device shadows service uses reserved MQTT [2] topics to permit applications and things to get, update, or delete the state information for a device [3].

The traditional access control models are access control lists (ACLs), capability lists, and role-based access control (RBAC). Attribute-based access control (ABAC) is receiving current interest as a more general model that encompasses the benefits of prior traditional models, as well as brings new features suitable for dynamic and open environments such as IoT. In this paper, we develop access control models for VO communication in two layers: operational models and administrative models, assuming topic-based publish-subscribe interaction among VOs. Operational models are developed using (i) ACLs for topics and capabilities for virtual objects, and (ii) ABAC. It is argued that RBAC is not suitable for this purpose. Administrative models for these two operational models are developed using (i) ACLs, (ii) RBAC and (iii) ABAC. A use case illustrates the details of these access control models for VO communication, and their differences. To the best of our knowledge, this is the first paper to address control of VO communication for IoT.

The rest of the paper is organized as follows. First, we review related work of access control models for IoT, background on publish/subscribe, and the developed ACO architecture for IoT in Section II. A use case about sensing the speed of cars, and flagging those above the limit, is introduced in Section III. In Section IV, we propose and discuss appropriate operational access control models for virtual object communication. Section V discusses administrative models for this purpose. Assessments of our models with respect to the IoT security and privacy preserving objectives proposed in [4] are discussed in Section VI. Finally, we offer our conclusions in Section VII.

II. Related Work and Background

A. Related Work

Several access control models for IoT have been proposed to address security and privacy issues, as surveyed in Ouaddah et al [4]. Using capability-based access control (CAC) model for IoT has been proposed because entities hold granted rights that support different levels of granularity with possibility of delegation, while similar functionality is not feasible with ACLs. However, the main two major

drawbacks of using the capability approach are propagation and revocation [5]. Mahalle et al [6] propose the identity authentication and capability-based access control (IACAC) model, where devices use an access point and the CAC model to be connected to each other. Moreover, in [7], the capability-based access control system (CapBAC) is used in controlling access to services and information. The authors described use cases and argue that CapBAC supports rights delegation, least privileges access principle, more fine-grained access control, fewer security issues, and fewer issues related to complexity and dynamics of subject's identities than ACLs, RBAC and ABAC. In [8], a simple-efficient mutual authentication and secure key establishment based on ECC, which has much lower storage and communication overheads, is proposed for the perception (object) layer of the IoT. The ABAC-based authorization method has been recently adopted for access control.

B. The Publish and Subscribe Communication Paradigm

The publish/subscribe communication interaction scheme is suitable for large-scale distributed interactions such as the IoT. It lets subscribers indicate their interest in topics (events) services that manage and deliver data generated by publishers. In other words, producers publish data on a software bus (topic/event service), and consumers subscribe to the information they are interested in receiving from that bus (topic/event).

The publish/subscribe paradigm has various implementation styles [9], [10], primarily topic-based and content-based. The topic-based scheme is similar to the notion of groups where consumers (subscribers) become members of a topic (a group), and producers (publishers) publish data to a topic. All subscribers to that topic are informed about the published data. In contrast, the content-based approach introduces a subscription scheme based on the published content. Here, subscribers specify filters, which define constraints based on the name-value pairs of the published properties (content) and use single or combined basic comparison operators (=, $<$, $\leq$, $>$, $\geq$) to identify events of interest [9], [10].

C. Access Control Oriented (ACO) Architecture for IoT

The Access Control Oriented (ACO) Architecture for IoT was proposed in [1], consistent with various published IoT architectures [11]–[19]. ACO architecture comprises four layers: an object layer, a virtual object layer, a cloud services layer, and an application layer, as shown in Figure 1. We will briefly discuss each layer below.

The object layer comprises heterogeneous physical objects such as sensors, actuators, cameras, cellphones, etc. Users can directly communicate with objects by pressing a button, changing a device, powering on an object, etc. Also, objects can communicate directly to each other through communication technologies, or indirectly through virtual objects.

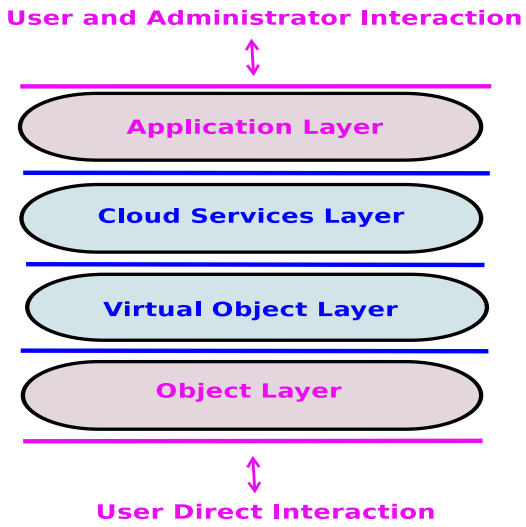

Figure 1. ACO Architecture for Cloud-Enabled IoT

A virtual object represents the persistent current status of a physical object, when the two are connected. When disconnected, a virtual object could represent the last received state, desired future state, or both. A virtual object can have a subset of a physical object's services, all of a physical object's services, or some of a physical object's services. Virtual to physical object association can be one-to-one, many-to-one, one-to-many, or many-to-one [1].

The use of virtual objects helps solve a variety of IoT issues such as scalability, heterogeneity, security and privacy, and identification [16]. Virtual objects in this layer can uniformly communicate with each other regardless of heterogeneity and locality in the object layer. It is important to control this communication by using access control mechanisms. It can also be beneficial to use multiple access control mechanisms in this regard [20].

The cloud services layer assists in storing and processing the collected data. This data can be used intelligently for smart monitoring and actuation, and it can be visualized in ways that are more meaningful for users. Thus, policymakers (or administrators) can utilize the visualized data to help them to modify or add policies that kept in the cloud, so the communication and access between applications and objects are managed through cloud services. In addition, multiple IoT clouds can also directly communicate with each other, ranging from only providing services and information at a local level (local cloud) to collaborating with other connected IoTs in order to share information at a broad level and pursue common goals.

The application layer is the topmost layer of the proposed ACO IoT architecture and offers an interface through which users can easily communicate with objects and visualize the analyzed information. Administrators can also interact with applications to generate policies or to update/add policies

based on the obtained information. Moreover, configuring and managing the communication of objects and virtual objects is organized by administrators through applications. General users and administrators can remotely communicate with IoT objects and virtual objects only through applications.

III. Use Case within ACO Architecture

We employ a use case of sensing speeding cars, illustrated in Figure 2, as a running example. A car is declared to be speeding if two sensors within a specified distance sense the speed to be over limit. Such cars will be reported by the camera along with a picture. Each car is assumed to have an RFID that enables sensors to identify cars. Objects in physical object layer (sensors and camera) generally have limited computational power and low storage. Also, issues such as scalability, heterogeneity, security and privacy, and identification can be handled easier within virtual object layer than object layer. Thus, we assume that the sensors and camera push collected data (e.g. RFID) to their virtual objects where more powerful computations could happen. For this use case, we assume communication between sensors can occur only within the virtual object layer, via publish/subscribe to topics, and they cannot communicate directly with each other.

The physical objects are sensors $S1, ..., Sn$ and a camera $C1$. Correspondingly, in the virtual object layer, there is a group of virtual sensors $VS1, ..., VSn$, one for each physical sensor, and one virtual camera $VC1$ for the physical camera. The physical sensors are linearly arranged along the highway. Similarly, their virtual sensors communicate in a linear sequence. We have topics $T1, ..., Tn$-1, where $T1$ enables communication from $VS1$ to $VS2$, and so on, through Tn-1 which enables communication from VSn-1 to VSn. Finally, topic Tn facilitates communication from the last virtual sensor VSn to the virtual camera $VC1$.

Physical sensors have the capability to sense speed and RFID of cars at the location where the sensor is located. Moreover, they have local storage and simple computation capability for the collected data to be refined and pushed to their virtual objects. The physical camera has the capability to sense RFID, current location, and take pictures. It has local storage, and local simple computation for the collected data to be refined and pushed to VC1. However, if the virtual objects are not connected to physical objects, the collected data will be kept temporarily in the local memory of physical objects. Eventually, the refined collected data will be pushed to the virtual objects and will be removed from local memory.

The scenario of communication among virtual objects starts with $VS1$, which publishes a suspicious RFIDs list of over-limit cars, received from $S1$, to $VS2$ through $T1$. $VS2$ also receives a suspicious RFIDs list from $S2$. $VS2$ compares these two suspicious RFIDs lists. RFIDs that occur

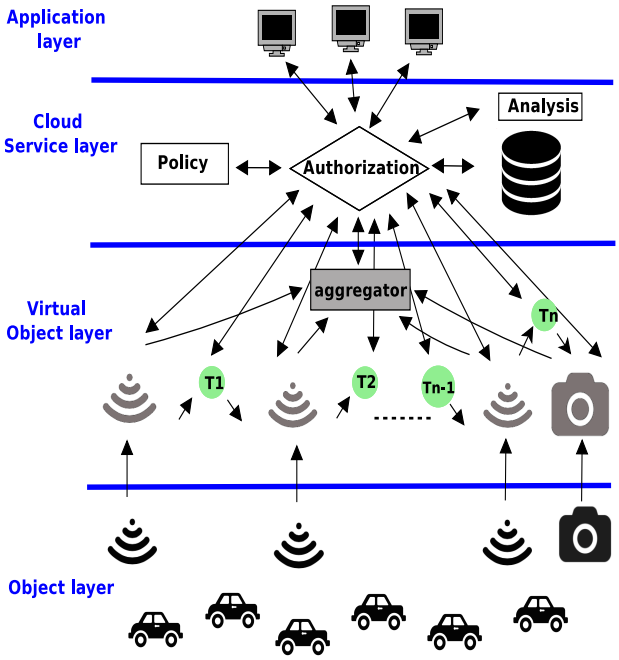

Figure 2. Sensing speeding cars within ACO Architecture

on both lists are added to a SavePic list located on $VS2$ as well as pushed to an aggregator, which is responsible to consolidate all incoming data from VSs and VC1 and push it to storage in the cloud services layer. RFIDs which occur on only one of the lists, are consolidated in a suspicious RFIDs list at $VS2$. Then, $VS2$ publishes the SavePic list and the suspicious RFIDs list to the next virtual sensor $VS3$. Other sensors and virtual sensors perform similar steps. $VC1$ compares the RFIDs (along with pictures) coming from $C1$ with RFIDs on the SavePic list. The matched RFIDs will be pushed, along with the taken pictures to the aggregator. Note that a picture is taken of over-limit car by the camera $C1$ and communicated to $VC1$, but only pictures of cars that are in the SavePic list are sent to the aggregator and communicated outside the VO layer. The other pictures are discarded. This shows the privacy benefit of separating the VO layer with transient information from the cloud services layer with persistent information.

Collected data cannot be accumulated in a device or a virtual object for long time. For example, when sensors are not connected to their virtual objects, collected RFIDs should not be saved in sensors longer than the time that an over-limit car can take to get to the next sensor. Similarly, a virtual object needs to publish the suspicious list frequently to the next virtual object to compare the suspicious list from the virtual object with the suspicious RFID list from its sensor. Matched RFIDs should be removed from the two lists and added to SavePic list, while the rest of RFIDs in the two suspicious lists will be combined as one suspicious list and published to the next virtual object. At the virtual

camera, the SavePic list should not be accumulated for long time. RFIDs that do not have a matched picture coming form the camera also need to be pushed up to the aggregator and then to the cloud.

IV. Operational Access Control for VO Communication

We develop access control models for VO communication in two layers. The operational model specifies controls regarding which VO to VO communications are authorized via topics. The administration of these controls is specified by the administrative models. This separation of operational and administrative models was first introduced in role-based access control, where operational models were defined in [21], [22] and administrative models in [23].

In this section we develop two operational models: ACLs and capabilities-based access control, and attribute-based access control. Publish/subscribe schemes typically employ message brokers (MBs) [9] (also called event brokers [10]) that route messages from publisher to subscribers for topics. After subscribers register (by sending a subscribe request) with a message broker of a topic, a published message to that topic will be forwarded by the message broker to all subscribers. The operational authorizations specify which VOs are allowed to publish to which topics, and likewise which VOs can subscribe to which topics. These authorizations determine the permitted pattern of communication in the VO layer, and thereby indirectly in the object layer.

The operational access control models address the following questions. Which VOs are allowed to publish or send a subscription request to a topic's MB? Which VOs should a topics MB forward data to? Which MBs should VOs publish to or send a subscription request to? Which MBs should VOs receive data from? These lead to the following related questions. Where should the publish and subscribe controls be located? On the topic side, virtual object side, or both?

The operational models recognize two sets of entities: virtual objects (VO) and topics (T), and a set of rights R={p,s} denoting publish and subscribe respectively. VOs are active entities that can publish data to topics, and receive data from topics they are subscribed to. Each topic has an associated MB, which responds to subscribe requests from VOs, accepts data published to the topic by a VO and forwards this data to the topic's subscribers. The right for the forward operation is represented in the singleton set F={Forward}. Note that these entities are very different in nature from the usual user/subject and resource/object entities in access control models [22], [24].

A. ACL and Capability Based (ACL-Cap) Operational Model

The ACL-Cap model incorporates ACLs for topics and capability lists (Cap) for VOs, as illustrated in Figure 3. These lists are maintained by administrators, as will be discussed in Section V. The ACL of a topic comprises a list of VOs, along with a publish or subscribe right for each VO. The capability list of a VO similarly comprises a list of topics with the publish or subscribe right for each topic. The capability list informs the VO as to which topics it can publish or subscribe, obviating the need for additional logic for this purpose. Similarly, the ACL of a topic informs the topic's MB as to which VOs can publish or subscribe to it. Since the VOs and topic MBs are fully automated, this dual ACL-Cap approach is more convenient and secure relative to ACL-only or capability-only approaches. This dual scheme allows unauthorized operations to be denied at the earliest possible moment, instead of deferring the decisions till later.

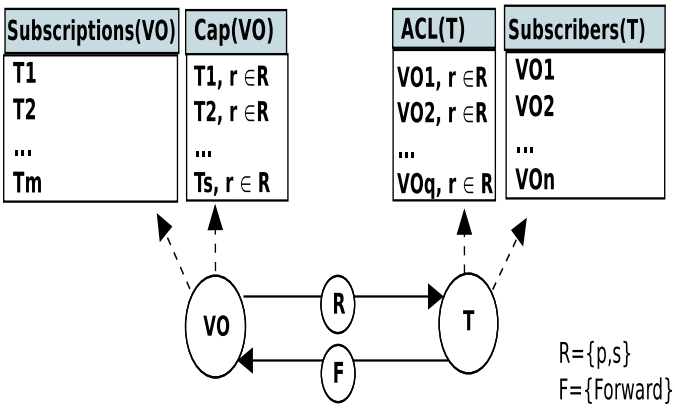

Figure 3. The ACL-Cap Model

Table I
ACL of Topics

$T1$	....	Tn-1	Tn
$VS1$, p	....	VSn-1, p	VSn, p
$VS2$, s	....	VSn, s	$VC1$, s

Table II
Capability List of VOs

$VS1$	....	VSm	$VC1$
$T1$, p	....	Tn, p	Tn, s
	....	Tn-1, s	

A particular VO can publish to topics for which it has a publish capability. The publish operation will succeed only if that topic's ACL has a corresponding entry for that VO with the publish right. The authorization rule for publish is therefore expressed as follows.

$$Auth\text{-}Publish(VO,T) \equiv (T,p) \in Cap(VO) \wedge (VO,p) \in ACL(T) \quad (1)$$

The subscribe operation is more complicated, in that the subscribe relationship needs to be established before published data is forwarded and received. This requires a request to

subscribe from a VO to a topic, and an accepting response from the topic's MB. Recognizing that this is a multi-step operation, we express the authorization rule for successful completion of subscribe as follows.

$$Auth\text{-}Subscribe(VO,T) \equiv (T,s) \in Cap(VO) \wedge (VO,s) \in ACL(T) \quad (2)$$

A successful subscribe operation adds the topic T to the VO's subscriber list, and the VO to the topic's subscriber list as shown in Figure 3. The actual forwarding of published data by a topic's MB to a VO is authorized as follows.

$$Auth\text{-}Forward(T,VO) \equiv VO \in Subscribers(T) \wedge T \in Subscriptions(VO) \quad (3)$$

Equations 1 and 2 respectively address the questions: which VOs are allowed to publish or send a subscription request to a topic's MB? Equation 3 addresses the question as to which VOs a topic's MB can forward data to. Note that equation 1 can be partially checked at the publishing VO's side, thus preventing a rogue VO from indiscriminately attempting to publish to unauthorized topics (as would be possible in an ACL-only approach).

For the use case defined in Section III, we have $VO = \{VS1, .., VSn, VC1\}$ and $T = \{T1, .., Tn-1, Tn\}$, with the ACLs and capability lists given in Table I and Table II.

B. ABAC Operational Model

Next we develop an ABAC operational model, illustrated in Figure 4. The entities in this model are the set VO of virtual objects and the set T of topics with rights R={p,s} and F={Forward}, as before. We have a set of attributes, VOA for virtual object attributes and TA for topic attributes, as follows.

$$VOA = \{VO\text{-}Publish, VO\text{-}Subscribe, VO\text{-}Subscriptions, VO\text{-}Location\}$$

$$TA = \{T\text{-}Publish, T\text{-}Subscribe, T\text{-}Subscribers, T\text{-}Location\}$$

The T-location and VO-location attributes are atomic valued and give the location of the corresponding topic and VO in appropriate units. The remaining attributes are set-valued. Values for VO-Publish, VO-Subscribe, and VO-Subscriptions are a subset of the topics T. Values for T-Publish, T-Subscribe, and T-Subscribers are a subset of the virtual objects VO. The following authorization rules express the same policy as in Section IV-A.

$$Auth\text{-}Publish(VO,T) \equiv T \in VO\text{-}Publish(VO) \wedge VO \in T\text{-}Publish(T) \quad (4)$$

$$Auth\text{-}Subscribe(VO,T) \equiv T \in VO\text{-}Subscribe(VO) \wedge VO \in T\text{-}Subscribe(T) \quad (5)$$

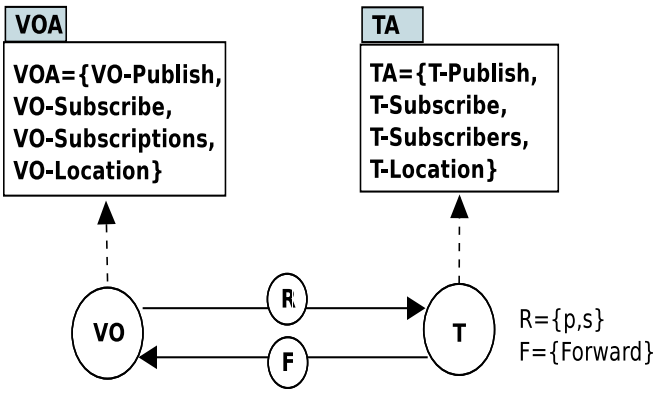

Figure 4. ABAC Operational Model

$$Auth\text{-}Forward(T,VO) \equiv T \in VO\text{-}Subscriptions(VO) \wedge VO \in T\text{-}Subscribers(T) \quad (6)$$

The attributes VO-Publish, VO-Subscribe, T-Publish and T-Subscribe are assigned by administrators. The VO-Subscriptions and T-Subscribers attributes are assigned as a consequence of establishing the subscribe relationship as discussed in Section IV-A. The T-location and VO-location attributes are enhancements to the use case in the ABAC model. We assume that VO-Location is automatically assigned to be the location received from the physical sensor. The T-location attribute is assigned by an administrator. We can conjunctively add the following condition to each of the three equations above.

$$T\text{-}Location(T) \approx VO\text{-}Location(VO) \quad (7)$$

This will further constrain the pattern of communication amongst the VOs by taking their location into account. In particular, if sensors are moved a significant distance the authorized communication will be disrupted. Small movements will be accommodated due to the approximate matching in this condition.

Note that a single ABAC authorization rule incorporates topic and virtual object attributes. In this respect equations 4, 5 and 6, are respectively similar to equations 1, 2 and 3. However, ABAC allows incorporation of additional attributes such as in equation 7, whereas the ACL-Cap model is limited to the ACL and Cap lists as the only permitted "attributes."

C. RBAC Limitations

In closing this section we discuss some limitations of RBAC in context of IoT VO communications. RBAC was invented with the notion of assigning users to roles, through which users acquire permissions primarily to perform operations on target objects. Virtual objects and topics do not fit this paradigm very cleanly. Virtual objects are active entities in regard to publish and subscribe operations, while

they are targets for forward operations. Similarly, topics are targets with respect to publish and actors with respect to forward and accepting subscribe requests. The active aspects of virtual objects and topics can be accommodated in RBAC by assigning these entities to mutually exclusive sets of roles. With respect to equations 1 and 2, the first part of the equations (i.e., $(T, p) \in Cap(VO)$ and $(T, s) \in Cap(VO)$) could be represented by permission assignment of topic permissions to the VO's role. The second part (i.e., $(VO, p) \in ACL(T)$ and $(VO, s) \in ACL(T)$) could similarly be represented by permission assignment of virtual objects permissions to T's role. But this splits the equations into separate roles, which must thereby both be considered when access decisions are made. This consideration of roles of both actor and target requires major extension to conventional RBAC [22].

V. Administrative Access Control for VO Communication

In this section, we develop three administrative access control models to control VO communication, respectively using ACL, RBAC and ABAC approaches. An administrative model is an essential complement to the operational models described earlier. At the same time the structure of an administrative model is not tightly coupled with that of the operational model, as will demonstrate. We use the terms admins to mean users who are authorized to control VO communication, by adjusting configuration of the operational model. For simplicity, we assume admins of topics are same as admins of related VOs.

For the ACL-Cap operational model we have two main administrative questions: Who is allowed to add or delete (VO,p) or (VO,s) from ACL of T? Who is allowed to add or delete (T,p) or (T,s) from Capability list of VO? Slightly different administrative questions arise for the ABAC operational model: Who is allowed to assign or delete values to/from attributes of T? Who is allowed to assign or delete values to/from attributes of VO?

A. Administrative ACL Model

In addition to the operational model for our use case, the administrative ACL model introduces a set of admin users (A) and admin permissions (AP) as follows.

$$A = \{U1, .., Um\text{-}1, Um\}$$
$$AP = \{Own, Control\}$$

The administrative ACL model has one ACL for each T and VO as shown in Figure 5. Own and Control are similar in authorizing modifications to ACLs, Capability lists, and administered attributes of topics and virtual objects as appropriate for the underlying operational model. The difference between Own and Control is that Own authorizes the admin user to grant Own or Control over the topic or virtual object to other admin users, while Control does not.

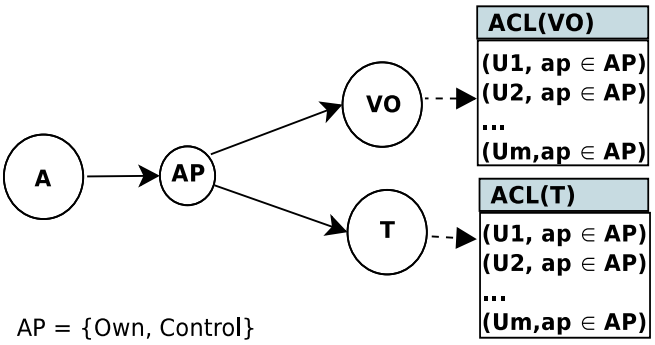

Figure 5. Administrative ACL

Table III
All Admins have Own Permission for all VO and T

T1, VS1 Admins	T2, VS2 Admins	.....	Tn, VSn Admins	VC1 Admins
(U1, Own)	(U1, Own)	.....	(U1, Own)	(U1, Own)
....	....	.....	....	....
(Um, Own)	(Um, Own)	.....	(Um, Own)	(Um, Own)

Table IV
Only U1 has Own Permission

T1, VS1 Admins	T2, VS2 Admins	.....	Tn, VSn Admins	VC1 Admins
(U1, Own)	(U1, Own)	.....	(U1, Own)	(U1, Own)
(U2, Control)	(U2, Control)	.....	(U2, Control)	(U4, Control)
(U3, Control)	(U3, Control)	.....	(U3, Control)	

A particular admin user U can control T or VO only if (U, ap) is respectively in the ACL of T or VO, where ap is Own or Control. We express the authorization rule for U to control T or VO as follow.

$$Auth\text{-}Control(U, T) \equiv (U, ap) \in ACL(T) \quad (8)$$

$$Auth\text{-}Control(U, VO) \equiv (U, ap) \in ACL(VO) \quad (9)$$

1) Administrative ACL Model for Operational ACL-Cap: This model is shown in two different configurations in Tables III and IV. In Table III all admin users have the Own permission for all topics and virtual objects, while in Table IV only U1 does. Presumably U1 has granted U2 and U3 control over topics T1 to Tn-1, and virtual sensors VS1 to VSn. Control over VC1 is granted to admin U4.

2) Administrative ACL Model for Operational ABAC: The ACL administrative model does not change structurally, but the meaning of Own and Control are adapted to the ABAC operational model. The difference between Own and Control remains as discussed above, and only impacts the administrative ACLs. For operational ABAC the Control

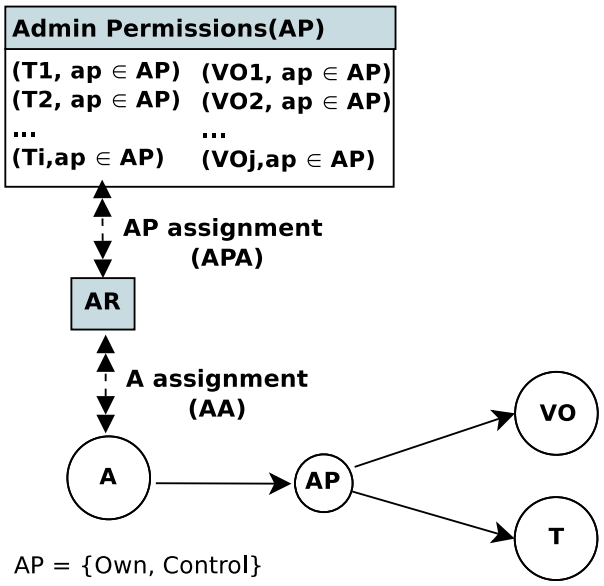

Figure 6. Administrative RBAC

permission over a topic or virtual object authorizes the admin to correspondingly modify topic or virtual object attributes, that are administrable. These are VO-Publish, VO-Subscribe, T-Publish, T-Subscribe and T-Location in our use case. VO-Subscriptions, T-Subscribers and VO-Location are automatically assigned and not administered by admins.

In both cases above, the administrative ACL model has one ACL for each topic and each virtual object. Thus, with large sizes of VO and T this will be difficult to maintain.

B. Administrative RBAC Model

The administrative RBAC model for our use case continues to use the set of admin users $A = \{U1, .., Um\text{-}1, Um\}$ and admin permissions $AP = \{Own, Control\}$, introduced in Section V-A. Additionally, it introduces a set of administrative roles (AR) and admin permissions (AP) as follows.

$$AR = \{AR1, .., ARs\}$$
$$AP = (VO \times AP) \cup (T \times AP)$$

A particular U can control a topic or virtual object only if U has admin assignment (AA) with some administrative role AR_1 where AR_1 has admin permission assignment (APA) with that virtual object or topic, as shown in Figure 6.

The administrative RBAC model is much easier to maintain than administrative ACL, due to well-known advantages of RBAC over per-topic and per-VO ACLs. The number of administrative roles that need to be managed is reduced to one for the configuration of Table III as shown in Figure 7, and to three for the configuration of Table IV as shown in Figure 8. These are constants numbers as opposed to the linear increase in ACLs with increase in topics and virtual objects.

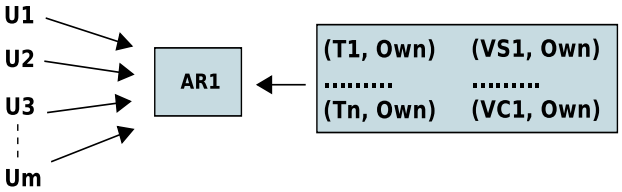

Figure 7. Administrative RBAC: Reflects Table III

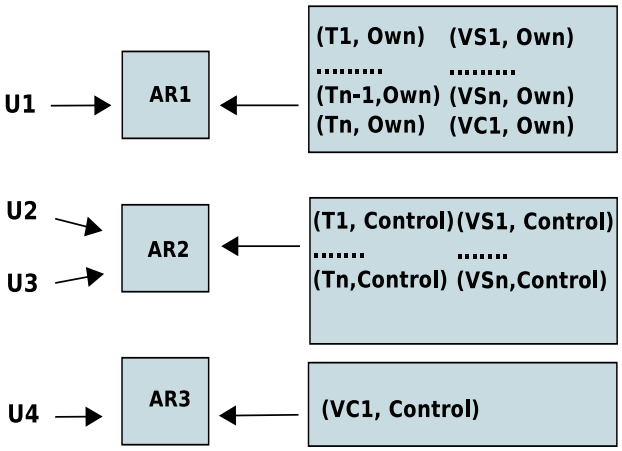

Figure 8. Administrative RBAC: Reflects Table IV

C. Administrative ABAC Model

The administrative ABAC model for our use case continues to use the set of admin users $A = \{U1, .., Um\text{-}1, Um\}$ and admin permissions $AP = \{Own, Control\}$, introduced in Section V-A. It also introduces administrative attributes for topics (TAA), VOs (VOAA), and users (UAA), as follows.

$$TAA = \{T\text{-}Location, T\text{-}Department\}$$
$$VOAA = \{VO\text{-}Type, VO\text{-}Location, VO\text{-}Department\}$$
$$UAA = \{U\text{-}Type, U\text{-}Location, U\text{-}Department\}$$

Note that these reuse the operational attribute introduced in the operational ABAC model of Section IV-B for Location, and add additional administrative attributes Type and Departmen. These administrative attributes are atomic valued. The range of the Type and Department attributes are some small number of enumerated items in each case. Figure 9 shows TAA, VOAA, and UAA being used to authorize AP for A. The interpretation of the Own and Control permissions for the two operational models is as discussed in Section V-A.

The authorization to use the Control permission with respect to virtual objects or topics is specified as follows.

$$Auth\text{-}Control(U, VO) \equiv$$
$$(U\text{-}Type(U) = Own \vee U\text{-}Type(U) = Control) \wedge$$
$$U\text{-}Department(U) = VO\text{-}Department(VO) \wedge$$
$$(VO\text{-}type = sensor \vee VO\text{-}type = camera) \wedge$$
$$U\text{-}location \approx VO\text{-}Location(VO)$$

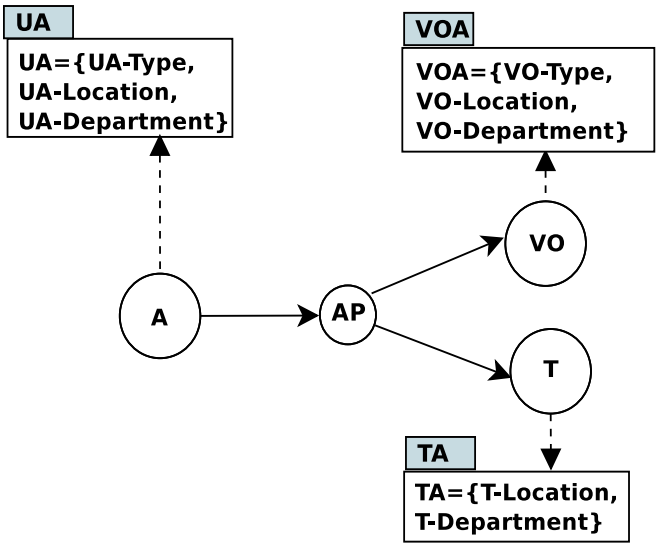

Figure 9. Administrative ABAC

$$Auth\text{-}Control(U,T) \equiv$$
$$(U\text{-}Type(U) = Own \vee U\text{-}Type(U) = Control) \wedge$$
$$U\text{-}Department(U) = T\text{-}Department(T) \wedge$$
$$U\text{-}location = T\text{-}Location(T)$$

These representative equations provide the $Control$ permission to a user who has Own or Control type for a VO if they are in the same Department and approximate Location, provided the VO is of type sensor or camera. The $Control$ permission to a user who has Own or Control type for a topic T is provided if they are in the same Department and exact Location (recall Location of a topic is an administered attribute), provided the VO is of type sensor or camera. In ABAC these rules can be easily modified or refined, e.g., we could have separate rules for sensors and cameras. ABAC is flexible, scalable and adaptable because it abstracts identity, role, and resources information of ACL and RBAC approaches into VO, topic and user attributes. Also, collected data (e.g. VO-Location) can be used as attributes values, which collaborate with other attributes to make a decision.

VI. Assessments With Respect to IoT Security and Privacy Objectives

Ouaddah et al [4] discuss security and privacy requirements for several IoT application domains, and classify these into six categories: privacy, technological constraints, social and economic aspects of the IoT, confidentiality and integrity, reliability and availability, and usability. Each category has various objectives. In the following, we will assess these objectives with respect to our ACO architecture, access control models, and our use case.

The privacy of users in our ACO architecture and access control models is generally maintained. The VO layer collects data from objects, so no third parties (such as other virtual objects) can get these data without rights (publish, subscribe) that are managed by admin (admin-driven permissions [4]). Once a right is obtained, VOs can share their data with each other without any intervention (decentralization [4]) or observation by a third party (a user, the Cloud, etc). Our use case identifies over-speed cars and tracks them using RFIDs (pseudonymity [4]), and the collected data are kept in virtual object layer until a decision is made. When a speeding decision is made, pictures of the cars (the cars' license plates are disclosed) are persistently saved along with their RFIDs and shared with the Cloud service layer. Otherwise they are discarded (privacy). Moreover, our ACO architecture helps users to control their collected data by keeping their data in VO layer or pushing anonymous sub-data to the Cloud service layer. Furthermore, access control models help users to control their own data by setting rights for VO communication (user-driven [4]). However, in our use case, the driver of a car has no control over the collected data about their car speed. We also needed to link the specific actions of the same car to track their speed, so the RFID is maintained and a picture is taken if needed.

IoT area has significant technological constraints. Our ACO architecture allows pervasive heterogeneous objects (sensors, camera, etc). Once they connect with their VOs, those VOs can communicate through the publish/subscribe scheme despite their heterogeneity. Our access control models for VO communication achieve most of the complex computation within the VO layer, and the physical objects layer is typically collecting data (sensors) and doing very simple computations (if a car is over the speed limit, then push its RFID to VSj).

Our ACO architecture is designed to allow for Cloud collaboration among different organizations. Further, communication among VOs within different organizations is possible using our access control models. For example, collaboration among the ACL of topics and the capability list of VOs models can support interoperability and cooperation by communicating directly (publish to current Cap(VOs), and accepting publish commands from current ACL(T)). In addition, by using ABAC, attributes of VO and T can be shared among the Cloud service layer and decisions made. Moreover, ABAC supports making decisions by using assigned attributes and the surrounding contextual attributes (sensor location, time, etc.) about an object, the environment, or a user (i.e., context awareness [4]) .

The ABAC operational model has a higher level of granularity than the ACL-Cap model, where the specification of the access control rules has more flexibility and incorporates more information about objects, users, and the environment. A VO's access to a topic can be easily revoked by deleting the VO from the ACL of a topic T, a topic T from the capability list of a VO, a VO from T-Publish/T-Subscribe, or a VO from VO-Publish/VO-Subscribe. Finally, admin users

with their own permission can grant or delegate their granted permission to other users.

Once a right r is granted to VO or T, an access control decision is made regardless of the connectivity of resource owner (i.e., offline mode [4]). The availability time to publish/subscribe to T is ready once ACL of T is checked. That might take little bit more time with ABAC since both of VO and T attributes need to be checked.

The ACL of T and the capability list of VO is easily managed and modified by authorized users. However, because of the need of context awareness in access control decisions, ABAC facilitates managing access control authorization (Auth-Publish, Auth-Subscribe) by combining various attributes that are related to an object (e.g. Location), the environment (e.g. Time), or the user (e.g. Department).

VII. Conclusion

In this paper, we used our ACO architecture to propose the ACL-Cap and ABAC operational models to control virtual object communication. We noted the unsuitability of conventional RBAC for this purpose. We illustrated the operational models by means of a use case involving sensors, camera and speeding cars. We further developed ACL, RBAC, and ABAC administrative access control models in context of this use case, and identified the advantages offered by progressing to more sophisticated models in this regard. Finally, assessment of security and privacy objectives for IoT, as identifed by Ouaddah et al [4], are discussed in context of our ACO architecture, access control models, and our use case.

Acknowledgment

This research is partially supported by NSF Grants CNS-1111925, CNS-1423481, CNS-1538418, and DoD ARL Grant W911NF-15-1-0518.

References

[1] A. Alshehri and R. Sandhu, "Access control models for cloud-enabled internet of things: A proposed architecture and research agenda," in *the 2nd IEEE International Conference on Collaboration and Internet Computing (CIC)*. IEEE, 2016, pp. 530–538.

[2] OASIS, *MQTT Version 3.1.1*, October 29 2014.

[3] Amazon Web Services. (2017, April) How the AWS IoT Platform Works - AWS. [Online]. Available: https://aws.amazon.com/iot/how-it-works/

[4] A. Ouaddah, H. Mousannif, A. A. Elkalam, and A. A. Ouahman, "Access control in the internet of things: Big challenges and new opportunities," *Computer Networks*, vol. 112, pp. 237–262, 2017.

[5] L. Gong, "A secure identity-based capability system," in *1989 IEEE Symposium on Security and Privacy*. IEEE, 1989, pp. 56–63.

[6] P. N. Mahalle and et all, "Identity authentication and capability based access control (iacac) for the internet of things," *Journal of Cyber Security and Mobility*, vol. 1, no. 4, pp. 309–348, 2013.

[7] S. Gusmeroli, S. Piccione, and D. Rotondi, "A capability-based security approach to manage access control in the internet of things," *Mathematical and Computer Modelling*, vol. 58, no. 5, pp. 1189–1205, 2013.

[8] N. Ye, Y. Zhu, R.-C. Wang, and Q.-m. Lin, "An efficient authentication and access control scheme for perception layer of internet of things," *Applied Math. & Info. Sciences*, vol. 9, no. 4, pp. 1617–1624, 2014.

[9] J. Bacon, D. M. Eyers, J. Singh, and P. R. Pietzuch, "Access control in publish/subscribe systems," in *the Second International Conference on Distributed Event-Based Systems*. ACM, 2008, pp. 23–34.

[10] P. T. Eugster and et all, "The many faces of publish/subscribe," *ACM computing surveys (CSUR)*, vol. 35, no. 2, pp. 114–131, 2003.

[11] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: A survey on enabling technologies, protocols, and applications," *IEEE Comm. Surv. & Tut.*, vol. 17, no. 4, pp. 2347–2376, 2015.

[12] A. Botta, W. De Donato, V. Persico, and A. Pescapé, "On the integration of cloud computing and internet of things," in *IEEE Int. Conf. on Future Internet of Things and Cloud (FiCloud)*, 2014, pp. 23–30.

[13] L. Da Xu, W. He, and S. Li, "Internet of things in industries: A survey," *IEEE Trans. on Indust. Informatics*, vol. 10, no. 4, pp. 2233–2243, 2014.

[14] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (IoT): A vision, architectural elements, and future directions," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[15] R. Khan, S. U. Khan, R. Zaheer, and S. Khan, "Future internet: the internet of things architecture, possible applications and key challenges," in *10th IEEE Int. Conf. on Frontiers of IT*, 2012, pp. 257–260.

[16] M. Nitti, V. Pilloni, G. Colistra, and L. Atzori, "The virtual object as a major element of the internet of things: a survey," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 2, pp. 1228–1240, 2015.

[17] P. Parwekar, "From internet of things towards cloud of things," in *2nd IEEE Int. Conf. on Comp. and Comm. Tech.*, 2011, pp. 329–333.

[18] B. P. Rao, P. Saluia, N. Sharma, A. Mittal, and S. V. Sharma, "Cloud computing for internet of things and sensing based applications," in *Sixth IEEE Int. Conf. on Sensing Technology (ICST)*, 2012, pp. 374–380.

[19] R. Roman, J. Zhou, and J. Lopez, "On the features and challenges of security and privacy in distributed internet of things," *Computer Networks*, vol. 57, no. 10, pp. 2266–2279, 2013.

[20] A. Alshehri and R. Sandhu, "On the relationship between finite domain ABAM and PreUCON$_{\mathrm{A}}$," in *Int. Conf. on Network and System Security*, 2016, pp. 333–346.

[21] D. F. Ferraiolo, R. Sandhu, S. Gavrila, D. R. Kuhn, and R. Chandramouli, "Proposed nist standard for role-based access control," *ACM Transactions on Information and System Security (TISSEC)*, vol. 4, no. 3, pp. 224–274, 2001.

[22] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman, "Role-based access control models," *IEEE Computer*, vol. 29, no. 2, pp. 38–47, 1996.

[23] R. Sandhu, V. Bhamidipati, and Q. Munawer, "The arbac97 model for role-based administration of roles," *ACM Transactions on Information and System Security (TISSEC)*, vol. 2, no. 1, pp. 105–135, 1999.

[24] X. Jin, R. Krishnan, and R. Sandhu, "A unified attribute-based access control model covering dac, mac and rbac," in *IFIP Annual Conference on Data and Applications Security and Privacy*. Springer, 2012, pp. 41–55.