2016 IEEE 2nd International Conference on Collaboration and Internet Computing

Access Control Models for Cloud-Enabled Internet of Things: A Proposed Architecture and Research Agenda

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 concept and deployment of Internet of Things (IoT) has continued to develop momentum over recent years. Several different layered architectures for IoT have been proposed, although there is no consensus yet on a widely accepted architecture. In general, the proposed IoT architectures comprise three main components: an object layer, one or more middle layers, and an application layer. The main difference in detail is in the middle layers. Some include a cloud services layer for managing IoT things. Some propose virtual objects as digital counterparts for physical IoT objects. Sometimes both cloud services and virtual objects are included.**

In this paper, we take a first step toward our eventual goal of developing an authoritative family of access control models for a cloud-enabled Internet of Things. Our proposed access-control oriented architecture comprises four layers: an object layer, a virtual object layer, a cloud services layer, and an application layer. This 4-layer architecture serves as a framework to build access control models for a cloud-enabled IoT. Within this architecture, we present illustrative examples that highlight some IoT access control issues leading to a discussion of needed access control research. We identify the need for communication control within each layer and across adjacent layers (particularly in the lower layers), coupled with the need for data access control (particularly in the cloud services and application layers).

I. Introduction

With the development of wireless communication systems over the last few decades, the concept of Internet of Things (IoT) has emerged and has recently attracted increasing attention of governments, companies, and academia. The IoT is an extension of network technology, where the basic core of communication is the Internet. The promising IoT paradigm integrates many widely dispersed, mobile, abundant, heterogeneous objects, such as sensors and actuators that collect data from an environment and in turn act upon it. Many industries have initiated major projects even in the absence of widely accepted architectures for IoT. Thus, there is a crucial need to develop consensus architectures for the future IoT.

There have been various proposals for IoT architecture in the research literature. The proposed IoT architectures can be divided into three main layers: an object layer, one or more middle layers, and an application layer. The main difference in detail between them is in the middle layers. Some architectures abstracted the middle layers to only one layer [1], while others have two or more middle layers [2]–[5].

Integrating the cloud as a central entity is suggested in various IoT architecture [2]–[6]. The IoT can gain advantage from the powerful capabilities and resources of the cloud to offset its technological constraints. The IoT encompasses pervasive and heterogeneous objects that produce big non-structured or semi-structured data. IoT objects have limited computational power and low storage. Offering virtually unlimited computational capabilities, low-cost, on-demand storage capacity, and ubiquitous resources usable from everywhere, the cloud is the most convenient and cost-effective solution to deal with IoT technological constraints [7]–[10].

Moreover, several research papers have suggested incorporating an object abstraction layer as an essential part of IoT architecture. Atzoori et al. [5] argue for such a layer to unite access to the heterogeneous devices in the object layer. Evangelos et al. [11] suggest the benefit of exposing the capabilities and services of objects to the upper layers through such abstractions. A similar definition as [11] is given in [12] with the name of 'virtual object layer'. A virtual object is also described in [13] as comprised of both current and historical information about a specific physical object. A virtual object is called a device shadow in Amazon Web Service (AWS) IoT, which persists the final and desired future status of each device, even when the device is offline. The potential benefits of using virtual objects are discussed in depth in [11], with respect to IoT issues such as scalability, heterogeneity, security and privacy, and identification.

In the present paper, we take a first step toward our eventual goal of developing an authoritative family of access control models for a cloud-enabled Internet of things. We build upon previously published IoT architectures, which are all roughly divided into three layers: an object layer, one or more middle layers, and an application layer. In the different approaches the middle layer is divided into sub-layers differently. Since several papers discuss the advantages of using the cloud and virtual objects to solve IoT issues, our proposed access-control oriented (ACO) architecture supports using them in the middle of object and application layers. As a result, our proposed architecture is divided into four layers: an object layer, a virtual object layer, a cloud layer, and an application layer. This 4-layer architecture will be our guide to build access control models for a cloud-enabled Internet of Things. Within this architecture, we present several illustrative examples that expose some IoT access control issues. This leads us to discuss needed access control research to address these issues.

The rest of the paper is organized as follows. First, we review proposed IoT architectures from the research literature in Section II. In Section III, we propose a cloud-based IoT

978-1-5090-4607-2/16 $31.00 © 2016 IEEE
DOI 10.1109/CIC.2016.79

architecture and its characteristics. Illustrative examples are discussed in Section IV. A research agenda for access control based on our proposed architecture is discussed in Section V. Finally, we conclude this paper in Section VI.

II. Background on IoT Architectures

Various IoT architectures have been proposed, and these are divided into different layers. The general construction proposed in most IoT architectures includes three basic layers: an object layer, one or more middle layers, and an application layer [1]–[5]. In all the papers that we reviewed, an object layer and an application layer exist. Although the functionalities of the object and application layers might vary in their detail, in general they are quite similar. On the other hand, the middle layers vary in terms of the number of sub-layers and the proposed technologies. We will discuss each layer in the general IoT architecture as well as its entities and functionalities.

A. The Object Layer

The main task of the object layer (e.g., perception layer [4], [14] or hardware layer [15]) is to identify objects [4], collect data from the physical environment [2], [3], and reconstruct a broad perception of the data. This task is accomplished by using objects (devices) such as sensors that can query location, humidity, temperature, motion, etc. [2].

All papers agree that the primary entity of this layer is sensors and actuators. Some papers describe this layer as consisting of wireless sensor networks (e.g., cluster of sensors [16]), where sensors are the main physical objects that collect data. Other proposals add additional entities to this layer, such as actuators [2], [15], RFID tags [3], devices (e.g., cameras and cellphones) [12], and networks of devices [14].

The IoT relies on a pervasive and heterogeneous set of objects that produce big non-structured or semi-structured data [7]. These objects generally have limited computational power and low storage. Since IoT technology is a rich producer of big data [17], which is collected by constrained objects, the collected data needs to be transferred to a more capable layer through secure channels to provide added functionality. Moreover, with a large set of heterogeneous objects which have different operating conditions, functionalities, resolutions, etc. [12], providing seamless integration of these devices is a huge challenge, and this issue may hinder object interoperability and slow down the improvement of a unified reference model for the IoT [18].

B. The Middle Layers

The main goal of middle layers is to successfully convey the collected data from object layer to a remote destination [1], [14]. Many proposed IoT architectures describe the middle layers as only one layer. A transmission layer (gateway) proposed in [14] is responsible for gathering/sending data, packaging data, exchanging data, parsing/dispatching commands, and logging events between the application and object layer. All data is saved in the application layer in a database. A network layer is proposed in [1], [19], [20] as a middle layer; it is responsible for intelligently processing the massive amount of collected data.

While the transmission layer in [14] and the network layer in [1] are the single middle layer in the above architecture, other IoT architectures have proposed two or three layers between the application and the object layers. The proposed architecture in [21] consists of two layers in middle, the network layer and the service layer. The network layer connects everything together to share information, and it aggregates information from existing IT infrastructures such as power grids and healthcare systems. The service layer includes service discovery, service composition, trustworthiness management, and service APIs. The IoT architecture in [22] also introduced the network and middleware layer in the middle. The network layer transmits information to the middleware layer, which has service management, link to the database, information processing, automatic decision, and a ubiquitous computation unit that can be placed in the cloud.

Separating tasks between a network and a service/middleware layer [21], [22] is more robust than loading the network layer [1], [19], [20] with so many tasks. The service/middleware layer includes important tasks such as processing received data, managing services, making decisions, and computing tasks. Several papers suggest integrating with the cloud to support tasks in this layer [7]–[10], [22].

The main functionality of the middleware layer is providing a common set of device functionalities [2], [7], [11]. The middleware layer can also be divided into sub-layers. It is divided into two sub-layers in [2], which are the object abstraction and the service management layers. The service management layer pairs services of objects with requests for them, processes received data, and makes decisions, while the object abstraction layer transmits data collected by objects to the service management layer. Cloud computing and data management processes are implemented at the object abstraction layer. Other papers have proposed a middleware layer that is divided into three sub-layers: an object abstraction, a service management, and a service composition layer [5], [11]. The service composition layer offers the functionalities for the composition of single services, which are represented at the service management layer.

An approach to integrating cloud computing as a middle layer in the IoT architecture is proposed by Gubbi et al. [15], where the IoT architecture includes three layers: a wireless sensor networks layer, a cloud computing (middle) layer, and an application layer. Integrating cloud in the middle offers various functionalities to support a middleware layer. Gubbi et al. and other researchers [7]–[10] have discussed the integration of cloud computing with IoT.

The object abstraction layer is discussed in many papers, although they offer slightly different definitions. In [5], this layer consolidates access to the heterogeneous devices in the object layer, while in [11], it allows physical objects in the object layer to deliver their capabilities and features to the

upper layers. It is also called virtual object layer in [12]. Virtual object which is defined in [13] comprises both current and historical information about a specific physical object. However, Amazon called virtual objects as device shadows which persist both last and desired future status of each device even when the device is offline. The advantages of representing virtual object (e.g., digital counterpart of physical objects) as major component of IoT are discussed in depth in [18].

C. The Application Layer

Application layer is the topmost layer of the IoT architecture, and it delivers services and system functionalities to the end users. The application layer presents the information to the final users through merged and analyzed data. This layer exploits the functionalities of the middle layer and provides user-friendly applications of the IoT. Using applications is an easy way to remotely communicate to objects (devices) and present their information. The final received information from the middle layer can be used to create models, graphs, and flowcharts, which can support decision-making [2], [4], [5].

III. Access Control Oriented Architecture for IoT

From the reviewed papers, we conclude that there is a need for integrating the cloud with the IoT architecture and a benefit to using virtual objects as a counterpart of physical objects. Therefore, we propose an IoT architecture that emphasizes enabling cloud computing to support middleware and service management functionalities. Our architecture is designed to assist in proposing access control (AC) models for IoT, and thus we call it an AC-oriented (ACO) architecture for the IoT. We will show the details of this architecture and examples of it in this section.

Our ACO architecture is designed to be roughly close to the general architecture of the IoT that is divided into three layers: the application layer, one or more middle layers, and the object layer. Therefore, we kept the two basic layers (the application and object layers) that exist in all of the reviewed IoT architectures. However, the middle is divided into two layers: a virtual object layer and a cloud services layer. Therefore, our architecture basically includes four main layers: an object layer, a virtual object layer, a cloud services layer, and an application layer. Figure 1 represents the layering of ACO architecture for the IoT where object layer appears at the bottom and application layer at the top. Each layer has its components and functionalities. We discuss each layer below.

A. The Object Layer

This layer is similar to most of the object layers that we reviewed. The main task of this layer is to collect data from physical environment and to construct a broad overview of the data to send it to the upper level (virtual object layer) or to other objects. This layer includes heterogeneous types of objects such as sensors, actuators, and cameras, which form one cluster or multiple clusters of objects.

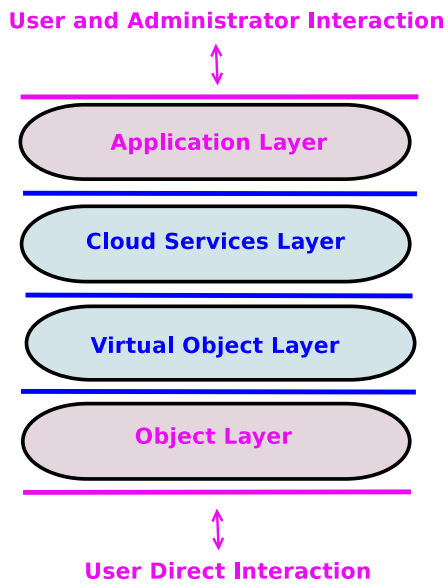

Fig. 1. ACO Architecture for the Cloud-Enabled IoT

Objects at this level basically push collected data to upper layers, such as data collected from sensors. However, objects at this layer also can receive information from other objects or from higher layers. For example, a light bulb needs to receive a command to be turned off or on. Thus, data at object layer could be output of objects or input to them.

The object layer is on the bottom of the IoT architecture. Users can directly communicate with objects by pressing a button, changing a device, powering on an object, etc. Objects in this layer can communicate with other objects directly through communication technologies such as Bluetooth, Wireless, ZigBee, 6LoWPAN, ISA 100, WirelessHart/802.15.4, 18000-7, and LTE [2]. They can also communicate to their virtual objects (digital counterparts) through the Internet. In both communication directions, there is a need to authenticate the communication possibly using technologies such as PKI or digital certificates.

The physically connected objects could be intentionally or unintentionally turned off or on. At the same time, the input/output data of physical objects could be needed/reached any time. Thus, knowing the status of objects in IoT architecture is required. One way to do it is to have virtual objects of physical objects. In addition, most of the physical objects will have limited computational power and low storage, and can only implement simple computational tasks and save limited data. Since IoT relies on vast sets of collected data, these physical objects need to depend on another party to execute intensive computational tasks, as well as to voluminous collected data. This party will be the cloud services layer, which will be described in part C.

B. The Virtual Object Layer

In the virtual object layer, virtual objects (digital counterparts) can present a persistent current status of objects if both

are connected. In case the virtual and physical objects are not connected, virtual objects could also present a desired future status, the last received status of a physical object, or both the future and last received status. Virtual objects deliver the services and capabilities of physical objects to users. Virtual objects can have a subset of physical objects' services, all of the physical objects' services, or one of physical objects' services. In our model, we will assume virtual objects only for physical objects. There is no digital counterpart for users (although that may be appropriate as the architecture and functionality evolve).

Using virtual objects solves IoT issues such as scalability, heterogeneity, security and privacy, and identification. Thus, the virtual objects in this layer can uniformly communicate with each other regardless of heterogeneity and locality in the object layer. This communication needs to be controlled by appropriate access control mechanisms, such as RBAC [23], ABAC [24], or ReBAC [25]. Studying the benefit of using multiple access control mechanisms is also possible [26].

Virtual objects can be associated with physical objects in various ways. The simplest is to represent one virtual object with one physical object (if any) that has one or many services, thus leading to a one (or less)-to-one association [27]. With an object that has many services, there is also the possibility of representing one virtual object for each service, thus leading to a one-to-many association. For example, a smartphone could represent all of its services through a single virtual object (one-to-one), or it could have separate virtual objects one for each available service, e.g. one for location sensing and one for temperature sensing, thus resulting in a one-to-many association [18], [28]. Another way would be to represent a set of physical objects with one virtual object, for instance, to manage them more efficiently with less resource consumption than having a distributed implementation (many-to-one) [18], [29], or to collect the information of single service from various physical objects (many-to-one) [18], [30]. Thus, the combination of all different kinds of associations will lead to many-to-many association [31].

C. The Cloud Services Layer

This layer is built to assist most of the functionalities related to the service/middleware layer. With an expected 50 billion smart objects in existence by 2020, attention must be focused on developing the means to access, store, and process the huge amount of data collected by these objects. Thus, this layer assists in storing and processing the big collected data. The saved data in this layer can also 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 are kept in the cloud, so the communication and access between applications and objects are managed through the cloud. The cloud services layer also assists in the intensive computational tasks that cannot be handled by the constrained objects. Thus, the cloud services layer supports the computation, visualization, and analysis of stored data in the cloud.

In addition to managing the communication with applications and objects, clouds can also communicate with each other, ranging from only providing services and information at a local level to collaborating with other connected IoTs in order to share information at a broad level and pursue common goals. Hence, multi-cloud communication can occur at this layer. As we mentioned above, controlling accessing to data and entities communications can be controlled by suitable access control mechanisms such as RBAC [23], ABAC [24], and ReBAC [25].

D. The Application Layer

The application layer is the top most 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. For example, a user who is out of her home can use an application to send a turn off command to remote light bulbs located at her home. Applications communication with any entity should be controlled and authorized by using appropriate access control techniques.

IV. Illustrative Example

The proposed ACO architecture for IoT in the previous section is illustrated in more concrete terms in Figure 2 in context of a simple example.

A. Multi-Value Switch Use Case

In our example, we have a multi-value switch that can change the color of a light bulb to red, blue, or green. Users communicate directly (and physically) with the multi-value switch to turn on the light bulb with a specific color. We will discuss each layer as follows.

1) The Object Layer: Although our ACO architecture in general allows objects to communicate directly to each other, we don't allow that in our example for simplicity. Therefore, the multi-value switch and the light bulb (objects) do not communicate with each other directly at this layer. Both of the multi-value switch and the light bulb connect to the Internet via secure channels to communicate with their virtual objects. Thus, the only communication allowed is with the virtual object layer.

Figure 2 shows one multi-value switch (object) that enables a color changing service. In other words, we have a physical object that has one service. Therefore, there is one virtual object that can associate with each multi-value switch, leading to a one-to-one association with the virtual object. Also, there is only one light bulb that receives a command to change its

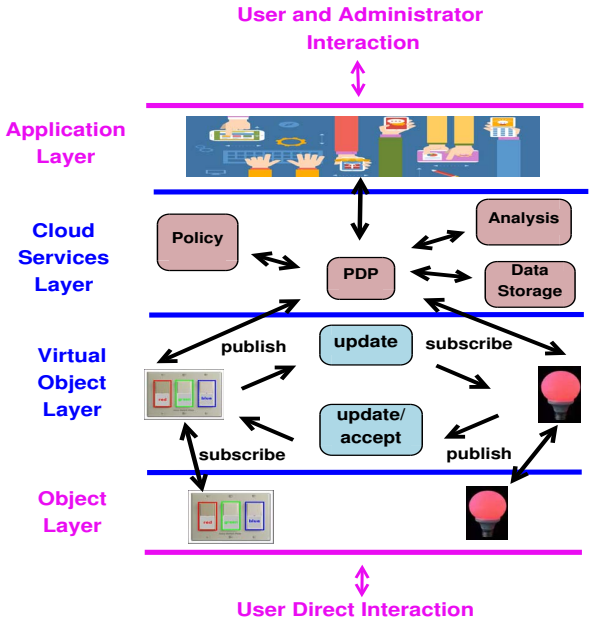

Fig. 2. Multi-Value Switch Use Case for the ACO Architecture

color, and for that light bulb there is one associated virtual object.

Users can directly interact with the light bulb and the multi-value switch by powering them on or off, changing them, or moving them, etc. Also, users can interact with multi-value switch by pressing a color. In our example, the collected data is only coming from users' action. When users press a color in the multi-value switch, the command is sent to virtual objects. The light bulb also needs to communicate with its virtual object to receive the new color; otherwise the color will stay the same. Over time, collected commands and received colors from the multi-value switch generate data that can be logged and saved.

2) The Virtual Object Layer: The virtual multi-value switch and virtual light bulb (virtual objects) store information about their physical objects. The virtual multi-value switch saves the current pressed command in the multi-value switch if they both are connected, and it will save the last received command in case if they are not connected. Similarly, the virtual light bulb will maintain the current color of the light bulb if they both are connected, and it will also save the last received color (the future color of the light bulb once it is connected) in case they are not connected. The current, past, and future status can be presented via list of attributes (e.g.'current-status', 'past-status', and 'future-status') that are saved in the virtual objects.

The two virtual objects communicate in three different ways. They communicate with their physical objects. They also communicate with each other directly at this layer. One familiar communication model between virtual objects is publish/subscribe [32]. Our simple use case has two topics: 'update' and 'update/accept'. The virtual multi-value switch publishes to 'update' topic, and the virtual light bulb subscribes to 'update' topic and thus receives any published command to change the state of the color; and vice versa with 'update/accept' topic. Finally, virtual objects can communicate with the cloud services layer to log and save the sending commands and the received colors, store the number of disconnections with physical objects, share attributes with the policy decision point (PDP), and receive authorized topics to publish or subscribe.

3) The Cloud Services Layer: This layer supports cloud services such as compute, storage and analysis of stored data. As shown in Figure 2, the cloud services layer has data storage that saves all the collected data (as discussed above). This data can be analyzed and visualized to decision makers to understand, for example, the difference between the number of sent commands from multi-value switch and the number of received commands to light-bulb.

The policy component stores rules that allow virtual objects to publish/subscribe to 'update' or 'update/accept' topics. In our example, the light bulb is allowed to publish to 'update/accept' topic but not to publish to 'update' topic. Policy rules are constructed and managed by administrators who communicate to this layer through applications. The PDP communicates with policy and data storage components, and with virtual objects to retrieve required information (e.g. roles and attributes) for making a decision [33]. For instance, it decides whether or not users can communicate to virtual objects and thus objects themselves.

4) The Application Layer: This layer includes an application to view the analyzed and visualized saved information, such as past statuses of the multi-value switch and the light bulb. The application allows the owner of the multi-value switch and light bulb to control communication between virtual objects, users' access to the saved data, and communication between applications and objects by constructing policies that are used by the PDP, which control various kinds of communication.

B. Multi-Value Switch Use Case Enhancements

Our use case showed a very simple scenaorio that has only two objects. Each object has one-to-one association with its virtual objects. This example can be enhanced in several ways such as adding multi-value switches and virtual objects, allowing direct communication between switches and light bulbs, or permitting collaborative multi-clouds, etc. Some examples of the enhancements are discussed as below.

As the number of rooms increase, more light bulbs are needed, and thus using one multi-value switch can control all of them efficiently. Introducing more light bulbs that connect to one virtual light bulb leads to a many-to-one association on the light bulbs side, which helps to manage them more efficiently, while there is a one-to-one association on the multi-value switch side. Figure 3(a) shows how one multi-value switch can control many light bulbs. However, how to control each room with different color is not clear with a many-to-one association on the light bulb side.

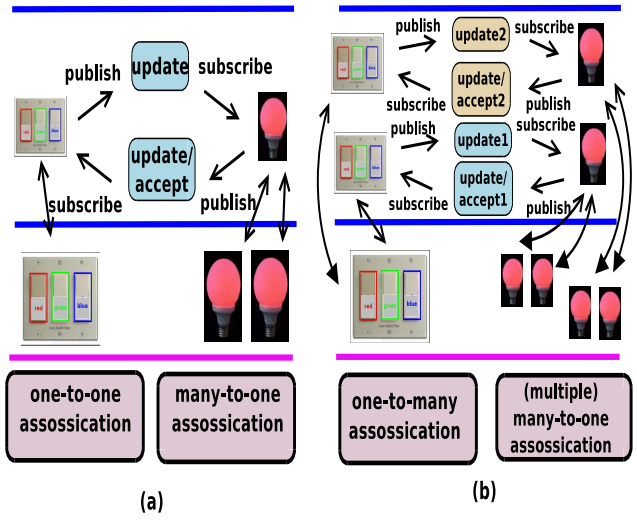

Fig. 3. Different kinds of objects and virtual objects associations

On the other hand, looking to control many rooms separately will result in increasing the number of multi-value switches, virtual multi-value switches, and virtual light bulbs. Designing a smart multi-value switch that considers how many times the red, the green, or the blue button has been pressed can result in a one-to-many association. In other words, this smart multi-value switch is associated to many virtual objects, rather than having multiple multi-value switches, each of them is associated with one virtual object. Figure 3(b) shows that with two groups of light bulbs (two rooms), each group associates with a different virtual light bulb (multiple many-to-one associations). The smart multi-value switch will be associated with two virtual multi-value switches. The first virtual multi-value switch is for the first group of light bulbs, and the other one is for the second group. Hence, two many-to-one associations are on the light bulbs side, while one-to-many association is on the smart multi-value switch side, which decreases the cost of having many multi-value switches.

In our simple case example, there is a one-to-one association on both the light bulb and the multi-value switch. One virtual multi-value switch is communicating with one virtual light bulb by pushing to 'update1' topic (this update is only for specific virtual object(s)). As a result, there are only two topics to publish and subscribe: 'update1' and 'update/accept' topics. However, Figure 3(b) shows more topics since we are looking to control two separate groups of light bulbs. Thus the 'update1' topic is for the first group and 'update2' is for the second one. In this case, adding a third group of light bulbs to be controlled separately will increase the number of topics.

As one advance enhancement is having groups of light bulbs and multi-value switches in one city, each group is for one neighborhood. The logged data, such as historical multi-value switch commands, is saved in the cloud. Another city that has a different cloud would like to communicate with the first city's cloud and retrieve the analyzed historical multi-value switch commands to discover the most required color in that city, for example, or to study the difference between the number of sent commands from multi-value switch and the number of received commands to light-bulb, and so on. This case shows why different clouds could communicate and collaborate within the cloud services layer in multi-cloud collaboration.

In the application layer, a smart phone could have an application that displays for users the current color, the past color, and the future color of a light bulb, as well as an illustrative graph that visualizes the number of times each color has been pressed so that users can understand what the most desired color has been. In addition to multi-value switch commands, an application could allow users to control the light bulb color remotely by pressing the required color and transmitting it within the cloud and the virtual object layer.

C. Controlling Communications and Data Access

Various access control models have been discussed such as attribute based access control model (ABAC) [24], relationship based access control model (ReBAC) [25], and role based access control model (RBAC) [23]. Access control models such as these can be employed to control communications between entities and controlling accessing to data.

In our simple use case, we can control virtual objects communication by adapting an appropriate access control model. ABAC model, for example, shows its capability for accommodating the need of the IoT in terms of the unlimited increase of objects. ABAC can be used to control communication between our two virtual objects: the virtual multi-value switch (VO1) and the virtual light bulb (VO2). Controlling which virtual objects are authorized to publish or subscribe to a specific topic is important here. In our case, the VO1 needs to be authorized to publish to the 'update' topic and subscribe to the 'update/accept' topic, while the VO2 needs to be authorized to subscribe to the 'update' topic and publish to the 'update/accept' topic.

For both of the two virtual objects, we have the following attributes: {Type1, Location, Current-color, Past-color, Future-color, Publish, Subscribe}. Each attribute has the following range: range(Type1) = {'apple switch', 'apple light bulb'}, range(Location) = {'home1'}, range(Current-color) = range(Past-color) = range(Future-color) = {'red', 'green', 'blue'}, and range(Publish) = range(Subscribe) = {'update', 'update/accept'}. We assume that Type1 and Location attributes' values are already assigned for both of the two virtual objects. Thus, a virtual object is allowed to either publish to 'update' topic if it is with Type1 'apple switch' and is located at 'home1', or it is allowed to subscribe to 'update' topic if it is with Type1 'apple light bulb' and is located at 'home1', and vice versa for the 'update/accept' topic. Figure 4 shows the authorization policy to publish or subscribe to 'update' topic and VO1 and VO2 attributes.

Historical sent commands (HSC) from VO1 and historical changed colors (HCC) of VO2 can be logged in the cloud storage. In that case, access control techniques are

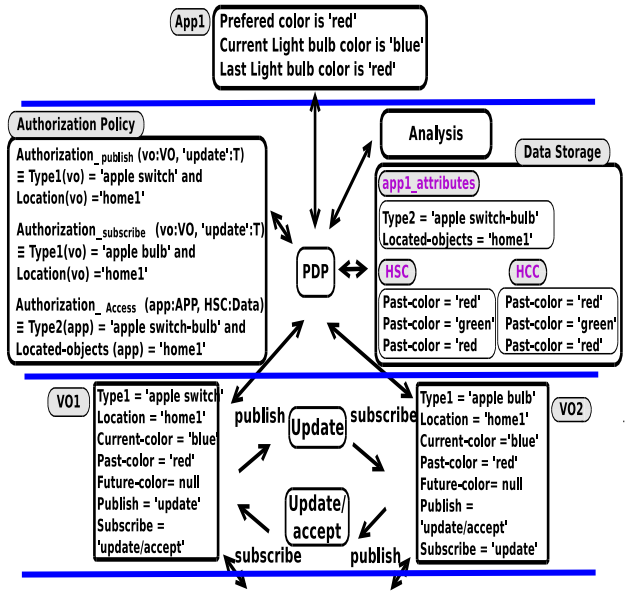

Fig. 4. Using ABAC to control virtual objects commuincations

needed to control accessing to historical data. For example, an application (App1), which represents information about historical sent commands and received colors, needs to access data storage to get that information. This application has the following attributes: {Type2, Located-objects}, and their ranges are as follows: range(Type2) = {'apple switch-bulb'}, range(Located-objects) = {'home1', 'home2'}. We assume that the application is already identified and the Type2 and Located-objects attributes values are already assigned for the application. By using ABAC model, applications can access historical sent commands and historical changed colors only if they have the following attributes values: Type2 = 'apple switch-bulb' and Located-objects = 'home1'. Figure 4 shows the authorization policy that allow an application to access historical data and application saved attributes. Also, it shows information that can be presented via the application (App1).

D. Object Life Cycle Issues

Looking to object layer in our simple use case, we have two objects that need to be designed with at least basic requirements of Internet of Things objects. For example, multi-value switches that do not connect to the Internet and communicate with virtual objects are not eligible to be placed with the Internet of Things objects. Thus, objects need to be designed and built to communicate.

Objects need to hold identifiers so as to be recognized once they connect to the Internet. With object identification, each object could be mapped to their virtual objects and authorized for communication with virtual objects. In our example, we have the virtual multi-value switch and the virtual light bulb. The decision of mapping light bulb to virtual multi-value switch or to virtual light bulb will need the light bulb identifier. Therefore, identifying objects is one important aspect of the object layer.

For each object there is at least one owner responsible for configuring that object, controlling its communication to other objects, and authorizing users and application to control/connect to this object. In other words, owners are the only users who can manage object policies. The multi-value switch in our example is permitted to send a command color only via owner authorization policy. As a result, we can say that ownership is significant for object security.

Designing objects to communicate with the assistance of identifiers and owner guidance leads to the secure deployment of these objects [34]. The secure communication of objects needs to be maintained periodically for these objects. The light bulb, for example, needs to be checked frequently for whether it is still permitted to communicate with a virtual light bulb or not. Additionally, objects or virtual objects that are not working any more or are not needed need to be changed or removed. Thus, ownership and policies of retired objects should be revoked for security purposes.

V. Research Agenda

Our use case reveals different possibilities of communications among entities in each layer and in different layers. As a result of these communications, the collected data flows among entities in various layers. From our ACO architecture and illustrative examples, we recognize two major issues that need to be controlled: communications among entities, and data that flows through these communications. Figure 5 represents the general two main recognized issues and entities in each of them.

A. Controlling Communications

Access control models have been frequently used to control data access. On the other hand, enforcing access controls to determine what kind of communication or traffic is allowed onto the network has been less frequently discussed by researchers although widely used in practice in firewalls [35]. A firewalls is a decision and enforcement point that grants or rejects any communication flow through it.

In general, communications between entities at different layers are possible in different directions. In our ACO architecture, objects can communicate directly with each other at object layer. Many protocols have been proposed for networked devices communications, such as Bluetooth and WiFi [36], [37]. In addition, objects can communicate with their virtual objects with different associations. These kinds of communications introduce questions such as, which objects are authorized to communicate with a specific object? Which objects are allowed to access specific virtual objects? What are the necessary requirements for objects to authorize them to communicate? Is the collected data from objects going to participate in controlling the communication of an object? All these questions can be studied and solved by proposing access control models for all kinds of objects' communications.

Within the virtual object layer, various communication models are possible. In our simple use case, we described a publish/subscribe communication style. Questions arise such as, who is allowed to publish or subscribe to the 'update' and 'update/accept' topics? What kind of information is needed to permit virtual objects to subscribe or publish? Questions such as these should be reviewed and resolved by suitable control models for virtual objects communications.

Our ACO architecture integrates the cloud as a service management layer, which helps in solving issues with IoT technology. It exists in the middle between the top and the bottom of the ACO architecture, so it is a crucial communication point that top and bottom entities should both access to communicate with each other. How do virtual objects get permissions to access each other or to access the cloud? Are virtual objects permitted to communicate directly with cloud entities or not? And what are the conditions and requirements for this communication? Do clouds communicate to share their information with each other? Can virtual objects be controlled or accessed through different (remote) clouds? Such questions should be addressed and appropriate control provided.

The only way for users and administrators to remotely connect with IoT entities is across applications in the application layer that generally display analyzed collected data. Also, applications can be used to control objects by sending commands that go from applications to cloud and virtual objects layers to control objects. Based on our ACO architecture, such communications between clouds, between virtual objects and clouds, and between applications and clouds can occur directly; communications between applications and objects, for example, should be transmitted through clouds and virtual objects. Figure 5 shows general entities that can communicate directly or through other layers and need to be controlled.

Within the IoT, indirect communication could introduce vulnerabilities in term of using an authorized communication to get unauthorized communication. For example, an application is allowed to communicate with a virtual object that has many-to-one association (many objects to one virtual objects). In contrast, the application is not allowed to communicate with some of the associated objects. Thus, such this association could cause indirect authorized access. It is important to have access control models to be used to control entities communication. As an initial step toward understanding controlling communication, an example of using ABAC to control a publish/subscribe communication style is discussed above in Section IV-C.

B. Controlling Access to Data

The ACO architecture integrates heterogeneous objects that collect data from an environment. Data could be collected by an individual object, such as the multi-value switch from our simple use case or a wearable FitBit device for one person. There could also be sub-data related to entities of the IoT, such as information about objects, virtual objects, and application. All sub-data and individual collected data can be accumulated and shared with others. Since our ACO architecture integrates

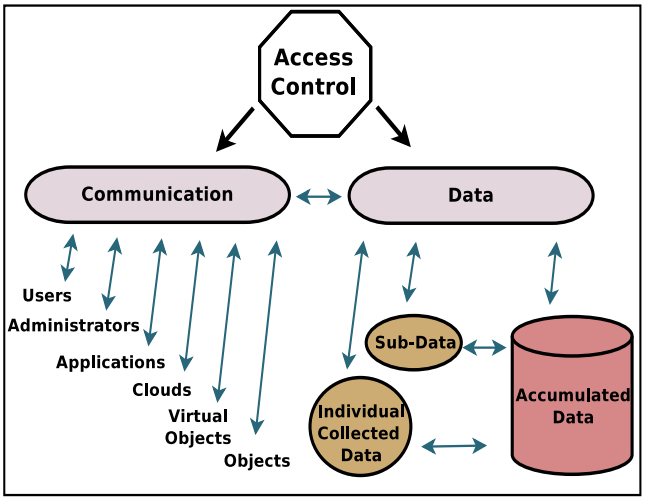

Fig. 5. Recognized access control issues from ACO architecture

the cloud, accumulated data is saved in the cloud's storage. Figure 5 shows that data is result of communication, and also communication could be established to retrieve data. Thus, the relation between data and communication is bidirectional. It also shows that individual collected data and by object(s) and the sub-data are a subset of all accumulated data. Differentiating between who is allow to access the individual collected, sub-data, or accumulated data is necessary.

Data security should be applied at every stage of the data lifecycle because it is vulnerable from the moment of transferring it from the owner's data storage until it is deleted from cloud storage. According to [38], the data lifecycle is divided into seven stages: data generation, transfer, use, share, storage, archival, and destruction.

In every stage of data lifecycle, the confidentiality and integrity of this data is important. There are many questions raised regarding data security and privacy. Can an object, a virtual object, or an application access data partially or entirely? If so, can they retrieve data directly or across other entities? Can accumulated data in one cloud accessed by remote clouds, objects, applications, etc.? What is accumulated data used for? These and similar questions present themselves when dealing these issues.

C. General Issues

The virtual object layer includes virtual objects for physical objects. However, it is important to address issues such as whether virtual objects exist first or physical objects? Should both virtual objects and physical objects exist together or one of them could exist first? Can a virtual object appear without being a counterpart for any physical objects? These questions need be to be considered by studying and controlling the mapping between virtual objects and their physical objects.

The owner or the administrator of entities creates an access control rule to govern the set of allowable capabilities. For example, all virtual objects of wearable FitBit devices can

view the average of the collected data from all wearable FitBit devices. Since the number of devices and applications in the IoT organization is unlimited, it is important for owners or administrators to apply access control policy without prior knowledge of particular entities that might require access [33]. ABAC allow an owner to implement access control policy without changing policy when new entities join.

Administrators control entities through applications that exist in the application layer. Since administrator and users both access through applications, it is important to distinguish administrators from users. How administrators control communications between entities at same and different layers? What kinds of actions that are self-control? What kinds of actions need direct control from administrators? All these questions need to be considered.

VI. Conclusion

In this paper, we take a first step toward our eventual goal of evolving an authoritative family of access control models for cloud-enabled Internet of things. First, we developed IoT architecture which is divided into four layers: the object layer, the virtual object layer, the cloud layer, and the application layer. This architecture will be our reference to build access control models for cloud-enabled Internet of things. We discussed illustrative examples that highlight the needed access control models for IoT. From our examples, we discussed the research agenda that could be studied.

Acknowledgments

This research is partially supported by NSF Grant CNS-1111925 and CNS- 1423481, and by grant 35157-Q1323101 from Mitre through University of Maryland.

References

[1] Z. Yang, Y. Yue, Y. Yang, Y. Peng, X. Wang, and W. Liu, "Study and application on the architecture and key technologies for IoT," in *IEEE Int. Conf. on Multimedia Technology (ICMT)*, 2011, pp. 747–751.

[2] 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. Surveys & Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015.

[3] P. Porambage, M. Ylianttila, C. Schmitt, P. Kumar, A. Gurtov, and A. V. Vasilakos, "The quest for privacy in the internet of things," *IEEE Cloud Computing*, vol. 3, no. 2, pp. 36–45, 2016.

[4] M. Wu, T.-J. Lu, F.-Y. Ling, J. Sun, and H.-Y. Du, "Research on the architecture of internet of things," in *3rd IEEE Int. Conf. on Advanced Computer Theory and Engg. (ICACTE)*, vol. 5, 2010, pp. 484–487.

[5] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, Oct 2010.

[6] 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.

[7] 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.

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

[9] 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. Conference on Sensing Technology (ICST)*, 2012, pp. 374–380.

[10] A. Botta, W. de Donato, V. Persico, and A. Pescapé, "Integration of cloud computing and internet of things: a survey," *Future Generation Computer Systems*, vol. 56, pp. 684–700, 2016.

[11] K. Evangelos A, T. Nikolaos D, and B. Anthony C, "Integrating RFIDs and smart objects into a unified internet of things architecture," *Advances in Internet of Things*, vol. 2011, pp. 5–12.

[12] C. Sarkar, A. U. Nambi, R. V. Prasad, A. Rahim, R. Neisse, and G. Baldini, "DIAT: A scalable distributed architecture for IoT," *IEEE Internet of Things Journal*, vol. 2, no. 3, pp. 230–239, 2015.

[13] E. Welbourne, L. Battle, G. Cole, K. Gould, K. Rector, S. Raymer, M. Balazinska, and G. Borriello, "Building the internet of things using RFID: the RFID ecosystem experience."

[14] Q. Zhu, R. Wang, Q. Chen, Y. Liu, and W. Qin, "IoT gateway: Bridging wireless sensor networks into internet of things," in *8th IEEE/IFIP Int. Conf. on Embedded and Ubiquitous Comp.*, 2010, pp. 347–352.

[15] 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.

[16] J. Voas, "Networks of 'things'," *NIST Special Pub. 800-183, 2016*.

[17] C. Dobre and F. Xhafa, "Intelligent services for big data science," *Future Generation Computer Systems*, vol. 37, pp. 267–281, 2014.

[18] 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.

[19] X. Jia, Q. Feng, T. Fan, and Q. Lei, "RFID technology and its applications in internet of things (IoT)," in *2nd IEEE Int. Conf. on Consumer Elect., Comm. and Networks (CECNet)*, 2012, pp. 1282–1285.

[20] M. C. Domingo, "An overview of the internet of things for people with disabilities," *J. NW. & Comp. Appl.*, vol. 35, no. 2, pp. 584–596, 2012.

[21] 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.

[22] 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.

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

[24] X. Jin, R. Krishnan, and R. Sandhu, "A unified attribute-based access control model covering DAC, MAC and RBAC," in *IFIP Conf. on Data and Applications Security and Privacy*, 2012, pp. 41–55.

[25] Y. Cheng, J. Park, and R. Sandhu, "Relationship-based access control for online social networks: Beyond user-to-user relationships," in *IEEE Int. Conf. on Privacy, Security, Risk and Trust (PASSAT)*, 2012, pp. 646–655.

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

[27] M. Langheinrich, F. Mattern, K. Römer, and H. Vogt, "First steps towards an event-based infrastructure for smart things," in *Ubiquitous Computing Workshop (PACT)*, 2000.

[28] IoT-A. (2010, Sep) Internet of things - architecture. [Online]. Available: http://www.iot-a.eu/public

[29] K. Römer, T. Schoch, F. Mattern, and T. Dübendorfer, "Smart identification frameworks for ubiquitous computing applications," *Wireless Networks*, vol. 10, no. 6, pp. 689–700, 2004.

[30] SENSEI. (2008, Jan) SENSEI - Integrating the physical with the digital world of the network of the future. [Online]. Available: http://cordis.europa.eu/pub/fp7/ict/docs/future-networks/projects-sensei-ec-summary_en.pdf

[31] IoT-iCore. (2011, Oct) IoT-iCore: Empowering IoT through cognitive technologies. [Online]. Available: http://www.iot-a.eu/public

[32] Amazon Web Services. (2016, Oct) AWS IoT features. [Online]. Available: https://aws.amazon.com/iot/how-it-works/

[33] V. C. Hu, D. R. Kuhn, and D. F. Ferraiolo, "Attribute-based access control," *IEEE Computer*, vol. 48, no. 2, pp. 85–88, 2015.

[34] Infineon. Lifecycle management security at every step of the device lifecycle. [Online]. Available: http://www.infineon.com/cms/en/applications/smart-card-and-security/internet-of-things-security/life-cycle-management/

[35] Paloalto Networks. What is a firewall? [Online]. Available: https://www.paloaltonetworks.com/documentation/glossary/what-is-a-firewall

[36] P. McDermott-Wells, "What is Bluetooth?" *IEEE Potentials*, vol. 23, no. 5, pp. 33–35, 2004.

[37] E. Ferro and F. Potorti, "Bluetooth and Wi-Fi wireless protocols: a survey and a comparison," *IEEE Wireless Comm. 12(1), pp12-26.*

[38] D. Chen and H. Zhao, "Data security and privacy protection issues in cloud computing," *2012 IEEE International Conference on Computer Science and Electronics Engineering (ICCSEE).*