Access Control in Industrial Internet of Things

Stavros Salonikias[1], Antonios Gouglidis[2], Ioannis Mavridis[1], Dimitris Gritzalis[3]

[1]Dept. of Applied Informatics, School of Information Sciences, University of Macedonia, 156 Egnatia str., 54636, Thessaloniki, Greece, {salonikias, mavridis}@uom.gr

[2]School of Computing and Communications, InfoLab21, Lancaster University, Lancaster, United Kingdom, LA1 4WA, a.gouglidis@lancaster.ac.uk

[3]Athens University of Economics & Business, 76 Patission Ave., Athens, 104 34, Greece, dgrit@aueb.gr

Abstract

The Industrial Internet of Things (IIoT) is an ecosystem that consists of -- among others -- various networked sensors and actuators, achieving mainly advancements related with lowering production costs and providing workflow flexibility. Introducing access control in such environments is considered to be challenging, mainly due to the variety of technologies and protocols in IIoT devices and networks. Thus, various access control models and mechanisms should be examined, as well as the additional access control requirements posed by these industrial environments. To achieve these aims, we elaborate on existing state-of-the-art access control models and architectures and investigate access control requirements in IIoT, respectively. These steps provide valuable indications on what type of an access control model and architecture may be beneficial for application in the IIoT. We describe an access control architecture capable of achieving access control in IIoT using a layered approach and based on existing virtualization concepts (e.g., the cloud). Furthermore, we provide information on the functionality of the individual access control related components, as well as where these should be placed in the overall architecture. Considering this research area to be challenging, we finally discuss open issues and anticipate these directions to provide interesting multi-disciplinary insights in both industry and academia.

Keywords: Industrial Internet of Things, IIoT, Access Control, Models, Mechanisms, Architectures, Requirements, Fog, Mist, Information Security

1. Introduction

The Internet of Things (IoT) is a term widely used to describe the existence of an ecosystem where pervasive and ubiquitous computing technologies are used to provide connectivity to physical things and make them part of a network where people, devices and things coexist and interact. IoT was greatly benefited from the development of underlying technologies in wireless and mobile networks, which in turn enabled the evolution of both the cloud and Wireless Sensor Networks (WSNs). WSNs provide things with sensors and actuators that are used to sense and produce, as well as consume data and interact with the environment. Advances in the IoT domain are so rapid that although the estimation of 50 billion devices in 2020 seems optimistic, the trend is inambiguous and a number of 20-30 billion seems feasible [19]. IoT is currently used in a number of domains, such as smart homes, smart cities, medical applications and the industry.

Over time, there have been some significant advances in technology that were acknowledged as milestones for the industry development, even characterizing the whole era: In the nineteenth century, steam provided the means for machine development and made the first industrial era possible. Afterwards, the significant development, that started the second era, was the deployment of electricity and its impact in the industry. The third era was characterized by the adoption of Information and Communication Technologies (ICT) that allowed for the development of Programmable Logic Controllers (PLCs) and Supervisory Control and Data Acquisition (SCADA) systems. Today, we witness the transition into the fourth industrial era, that is aided by the integration of a whole ecosystem of networked sensors and actuators into every aspect of the production stage. This integration between legacy industrial information systems and IoT, was initially described by the Industrie 4.0 initiative, mainly developed in Germany to provide competitive advantages by lowering production cost and providing workflow flexibility [25]. The outcome of the aforementioned integration is known as the Industrial Internet of Things (IIoT).

As in many emerging technologies, the adaption of ICT technologies in IIoT introduced issues with regards to standardization and security. Thus, a number of commercial entities have created the Industrial Internet Consortium (IIC) which has published a number of publicly accessible white papers on architecture and security [13]. In ICT, as well as in the IoT, information security (i.e., confidentiality, integrity and availability) is of major concern. However, in IIoT, additional concepts should be taken into consideration regarding the applicability in the application environment as well as the need for safety. Controlling access to resources for ecosystem stakeholders is crucial to fulfill both targets.

Introducing access control in IIoT is considered to be a challenging task stemming from the diversity that characterizes these industrial environments. The diversity is mainly introduced by the great variety of technologies and protocols supported by the IIoT devices and networks. Access control in Cyber-Physical

Systems (CPSs) has been examined in [18], where access control models are compared and a set of requirements is examined. Yet, we anticipate that further investigation may be required to cope with access control challenges in IIoT. The aim of this chapter is to provide additional information about the most promising access control models for IIoT, examine access control mechanisms able to support the described models and propose an access control architecture for IIoT based on virtualization technologies.

Specifically, in the following of this chapter, in Section 2, we provide background information on IIoT architectural trends, which are necessary to gain visibility to the ecosystem and extract access control requirements. In Section 3, major families of access control models and mechanisms are extensively presented. Access control approaches proposed in the literature for application in IIoT is examined in Section 4. The various components that constitute an access control architecture for IIoT are investigated in Section 5. Finally, in Section 6, we briefly elaborate on a set of open issues with regards to access control and IIoT and provide concluding remarks.

2. Background

The IoT is defined by the pervasive presence of things that are uniquely identified and are able to interact among them and with the rest of the network [3]. Initially introduced by Radio-Frequency Identification (RFID) tagging to provide Electronic Product Code (EPC), today IoT includes a number of heterogenous devices inter-connected using various protocols and technologies to provide the most efficient means of connectivity and interoperation.

Specifically, IoT describes a network of objects that may collect and share data in an autonomous manner and without requiring assistance by humans. Examples of such objects are considered to be various type of sensors that monitor and measure the temperature or humidity of the environment, the acceleration or position of an object, etc. The application scenarios of IoT are considered to be numerous, ranging from smart appliances (e.g., smart lighting and heating devices) to fitness devices (e.g., Fitbit).

The International Telecommunications Union (ITU) has released ITU-T Y.2060 [29], which is a recommendation that provides an overview of IoT. According to the recommendation, IoT adds a third axis in the already existing “anytime” and “anyplace” communication, that could be even provided by legacy ICT systems. The new axis is called “anything” and represents communication not only between computer devices, but also between human to human, human to thing and thing to thing. Things are objects that exist in the physical world and can be sensed and identified. The identification can be performed utilizing virtual entities which can exist without the presence of the physical ones.

Due to IoT great success and adoption rates, IoT technologies are also embraced by the industry and introduced in industrial environments as a means to improve operational efficiency [5]. Therefore, IIoT, "IoT Version 4.0" or "Manufacturing IoT" are expressions frequently used to denote the use of IoT for industrial purposes. By the end of 2020, it is estimated that more than 10 billion devices will account for the IIoT and represent the 57% of IoT spending [19].
IoT has already been a part of everyday life, including, but not limited to, smart cities, healthcare, agriculture, leisure (smart homes), construction, intelligent transportation systems, etc. There are many initiatives aim to exploit IoT in industrial environments, such as smart factories, Industrial Internet, Factories of the Future, etc. [25]. Although IoT and underlying technologies are well established and evolving constantly, adoption in the industry is a challenging task considering both the different environment and the fact that there are already well-established ICT systems in place (e.g., Distributed Control Systems (DCS) and SCADA systems) that control and monitor production process.

Industrie 4.0 is an initiative to support manufacturing in optimizing production efficiency and increase product quality. The initiative's underlying concept is to integrate IoT into legacy production field industrial information systems, thus being able to create a new concept, the IIoT. IIoT is enabled by the advances on Machine to Machine (M2M) communication, network efficiency and simplicity induced by 4G and 5G development and of protocols like 6LoWPAN and LoRaWAN and faces all challenges that exist in the IoT, such as resource constrained devices, heterogeneity, limited connectivity, etc. In the industrial environment an important factor is also the requirement for safety [25]. Although safety is not directly concerned with information security, being a key objective in IIoT operation, it must be taken under consideration to prevent accidents that could potentially threat the integrity of humans and machinery, as well as the availability of services. Access control models do not take safety under consideration as an inherent design feature, so safety provision should be considered, if possible, when creating access control policies.

2.1 IIoT Architecture

In March 2014, AT&T, Cisco, General Electric, Intel, and IBM co-founded the Industrial Internet Consortium (IIC) with the aim to promote the growth of IIoT. IIC has released version 1.8 of the IIoT Reference Architecture [16] where an IIoT analysis define four different viewpoints, i.e., business, usage, functional and implementation viewpoints. In this chapter we are mainly concerned with the implementation viewpoint where technological aspects can be revealed and examined.

With regards to the implementation viewpoint, IIC defines a three-tier architecture, namely, the edge, platform and enterprise tiers. The edge tier is

where data collection is performed from industrial and other end devices such as vehicles, machinery, workstations, automations, and all other sensors representing "things" in the industrial area. Data collected from the edge tier is sent to the platform tier, which is the medium between data collection and data exploitation, with the latter taking place to the enterprise tier (upper tier). Nodes residing on these tiers are inter and intra connected using different kinds of networks. These include the proximity network, which connects assets within the edge tier, the access network that connects the edge to the platform tier, and finally the service network that connects the platform with the enterprise tier.

The edge tier includes all the ICT components that are located in the production space. Example of such components are sensors, actuators and all other legacy devices and CPSs. The evolution of IoT led to the multiplication of the number of edge nodes that are characterised by physical limitations on computing and energy resources. The platform tier includes all the necessary processing that is required for edge device provisioning and data consolidation before those are delivered to the enterprise tier where services are developed. IIC does not provide topology-related constraints so, in its simplest form, platform and enterprise tier can be physically either located in premises or be powered by the cloud. Considering the volume of collected data, the cloud can be an enabling computing paradigm since it may provide the best candidate for big data processing. Connecting edge nodes directly to the cloud though, can be challenging considering the resource restrictions of many edge devices and latency induced by logical distance. The latter can be a potential threat to service provisioning as well as to system safety since the delay induced can lead to delayed actions that may cause damage. To overcome this issue, fog computing can be used as a middle layer between the edge and the cloud, thus reducing both distance and latency.

Fog was initially proposed by Cisco Systems [4] to provide a location aware and low latency virtualized layer between the edge and the cloud, thus bringing services nearer to the actual stakeholder. A fog layer is populated by private, community, public or hybrid [14] fog nodes that process information from edge devices and communicate with the cloud when necessary. In the fog concept, all information processing is performed in the fog nodes and little or none in the edge devices. Nevertheless, since nowadays network fabric can also provide the means to integrate processing into the network itself, a new layer can be developed between the edge and the fog. This is created by low-resources microcontroller-based devices with low-resources and is known as the mist layer [14]. Mist nodes are actually embedded in the same environment with the edge device, providing more accurate context information and enabling processing at the edge of the network, which further reduce the overall latency, provide contextual accuracy, and reduce power requirements from end devices.

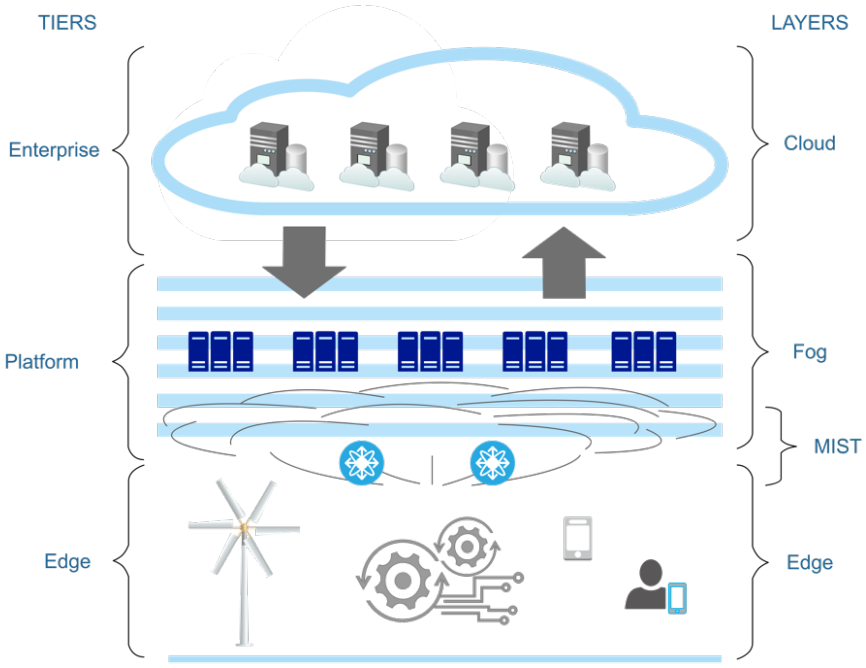

Fig. 1. IIoT Ecosystem

Although IIC edge tier is clearly matched to the edge layer, it is not so clear how to map the platform tier and the enterprise layer. Such a mapping usually depends on the specific application domain and topology. In Figure 1 an indicative mapping is depicted.

2.2 Access Control Requirements

Access control is essential in all systems that require to control and limit actions or operations that are performed by a user or process on a set of system resources [6]. An access control system is considered of three abstractions, namely, the access control policies, models, and mechanisms. Based on these abstractions, an access control system is made responsible for enforcing the access control policies and preventing them from subversion. Access control policies are characterized as high-level requirements that specify how and when a user, or a process, may access a resource. The access control policies are enforced through an access control mechanism, which is responsible for granting or denying access. An access control model is an abstract container of a collection of access control mechanism implementations, capable of preserving support for the reasoning of

the access control policies through a conceptual framework. Thus, access control models are bridging the abstraction gap between the policies and the mechanisms in an access control system.

In [24] an IoT enabled ecosystem utilizing the notion of fog computing in Intelligent Transportation Systems (ITSs) is presented. Considering the number of access control issues presented there, the following requirements can be extracted:

- **Context awareness:** Contextual information characterises the situation of an entity and the environment [1]. Context can influence access control decision and allow for policy creation that considers factors beyond subject's and object's identity. Having visibility into the context, access control policies can also be designed with an eye on safety on top of information security.
- **Inter-domain operation:** IIoT is deployed in multiple domains supporting operation of remote sectors under the same administration authority. Any access control solution should be able to support a coherent operation among different domains.
- **Privacy assurance:** Privacy is nowadays an important factor that needs to be considered in the deployment of every ICT solution (privacy by design). Since 2018, it is also a legal obligation in the European Union, defined by the General Data Protection Regulation (GDPR). An access control mechanism should be designed in a way that no private data should be ever disclosed.
- **Resource efficiency:** Most devices on the edge are designed to perform specific tasks and consume the less power possible. This limits available resources, both in terms of processing power and storage space, so any component designed to run on those, should take these limitations under consideration.
- **Manageability:** There should be a centralized way to create, store and enforce policies, that would not induce extra latency and could function over low-bandwidth networks, that may even sometimes become unavailable.
- **Accountability:** Auditing should be supported to provide respective stakeholders with the ability to monitor and reveal any violations or system misuse.

The list of the aforementioned requirements is not exhaustive, but instead it operates as stepping stones in choosing a more appropriate authorization scheme. In the following, we provide more information about families of access control models and frameworks towards their investigation in the context of IIoT environments.

3. Access Control Approaches

Although there is an abundance of access control models that could be applicable in IIoT environments, we elaborate in the following on major access control

family of models. This results in avoiding replication of information among models having their root on the same model family and help to describe the main characteristics offered by these models. Specifically, we provide information about the role-based, attribute-based, capability-based and usage control family of models.

3.1 Role-Based Access Control

In Role-Based Access Control (RBAC) [15], access to resources of a system is based on user and role assignment to roles, which have predefined permissions associated with them. RBAC may support several principles, e.g., least privilege, separation of duties and separation of administrative functions, which makes it preferable for use in organizational environments.
The core RBAC model is composed of five static elements, namely, *users*, *roles*, and *permissions*, with the latter being composed of *operations* applied on *objects*. With regards to relationships among elements, *roles* are assigned to *users* and *permissions* are assigned to *roles*. These types of relations may be of many-to-many, i.e., one user can be assigned to many roles and many users can be assigned to a single role. The same applies for role to permission assignments. Negative permissions are not supported in RBAC.

RBAC has two different phases, i.e., the design and run-time. During the design phase, a system administrator can define a number of assignments between the elements in the computer system. At the run-time phase, the assignments in the system are enforced by the model as it is specified by the security policy, which was prescribed during the design phase. Run-time enforcements are instantiated through the concept of sessions. The latter distinguishes RBAC from other group-based mechanisms. During a session, roles for a subset of users are allowed to be activated. This means that a user could be assigned various roles during the design phase, but these roles do not need to be activated always or simultaneously. Using the latter mechanism, RBAC provides support for the principle of least privilege. A number of constraints may be also enforceable during a session.

Apart from the core model, RBAC supports also hierarchies between roles. This mechanism provides great flexibility when it comes to the management of the policies. Specifically, permissions that are assigned to a role can easily be inherited to another role, without the need to reassign the same permissions to the latter. For example, we assume two roles R1 and R2 and two permission sets PR1=(P1,P2) and PR2=(P3,P4), which are initially assigned to roles R1 and R2, respectively. If role R1 inherits role R2, it means that all of R2's permissions are available through R1. The available permissions to role R1 are expressed by the union of permissions on sets PR1 and PR2. When hierarchies are represented in graphs, the immediate inheritance relation is shown as →. The head of the arrow or arc defines both the permissions and user membership inheritance. In the

previously example, we have R1 → R2. User membership refers to the assignment of users to roles in a hierarchy. In such a case, users are authorized to access all the permissions assigned to roles either directly or through inheritance relationships. Yet, another functionality that is provided in hierarchical RBAC is the support of general and limited role hierarchies. General hierarchies comprise the most common cases in role inheritance, and they are depicted as partial order sets. However, in more restrictive environments the requirement for supporting limited hierarchies may arise. This involves usually the existence of either a single immediate ascendant or descendant role in the hierarchy tree structure.

RBAC is also capable of supporting constraints through static and dynamic separation of duty relationships. The main objective in both types of constraints is to preserve the security of the system and prevent it from being compromised. Constraints are usually used to deliver business requirements. Static separation of duty relationships copes with the enforcement of conflict of interest policies. For example, let R1 and R2 be two conflicting roles and user U1 assigned to role R1. By enforcing a static separation of duty constraint between roles R1 and R2, RBAC prohibits the assignment of user U1 with role R2 since the two roles are conflicting. This type of constraints is defined and enforced in RBAC during the design phase. In the presence of a role hierarchy, the static separation of duty constrains are enforced in the same way for all the directly assigned and inherited roles. Dynamic separation of duty relationships handles conflict of interest policies in the context of a session. In this case, the user has a set of roles activated. A dynamic separation of duty relationship is described during the design time, but it is enforced during run-time – in the context of a session – to prevent the simultaneous activation of two or more conflicting roles. In case of role hierarchies, a similar mechanism to static separation constraints is applied, but constraints are enforced only on the set of activated roles.

3.2 Capability-Based Access Control

Capability-based access control (CapBAC) is based on the concept of capabilities [27], which are known to be communicable and unforgeable tokens of authority. A capability contains entries for the resources that a subject has granted access to. Thus, in a similar way to access control lists, an access control matrix is considered that may include *subjects*, *objects*, and *permissions*. In CapBAC, *permissions* are assigned with *subjects*, and thus support one-to-many relationships between *subjects* and *objects*. *Subjects and objects* refer to the users and resources of a system (in a similar way to RBAC). *Permissions* are authorized operations that can be performed by a *subject* on an *object*.

CapBAC support also *delegation* and *revocation* mechanisms for capabilities. These are required to delegate access (indirectly) to other *subjects* and revoke access, respectively. Usually, capabilities are issued in the context of a Simple

Public Key Infrastructure (SPKI) to cope with delegation of authorizations from one subject to another [23]. Such solutions may be applicable in multi-domain federated environments.

3.3 Usage Control

A representative usage control approach is UCON [22], which is based on a modern conceptual framework. The UCON conceptual framework encompasses traditional access control, trust management and digital rights management for the protection of digital resources. Nonetheless, functionalities such as administration and delegation are still absent. UCON has introduced a number of novelties compared to both RBAC and other attribute-based models, like its support for mutable attributes and continuity of access decision. Research has also been conducted regarding its usage in collaborative systems [30].

UCON is formed of eight components, namely, subjects, subject attributes, objects, object attributes, rights, authorizations, obligations and conditions. The notion of subjects and objects as well as the association with their attributes is straightforward. A subject can be an entity in a system and its definition, as well as its representation, is given by a number of properties or capabilities in the associated subject's attributes. For instance, role hierarchies similar to RBAC can be formed through the use of subject attributes. In regard to objects, they also represent a set of entities in a system. Each object can be associated with object attributes. Subjects can hold rights on objects. Through these rights, a subject can be granted access or usage of an object. This type of attributes can serve, for example, in the classification of the associated objects, by representing classes, security labels and so on and so forth. It is worth mentioning that both subject and object attributes can be mutable. This means that the values of the attributes can be modified as a result of an access. To the contrary, when an attribute is characterized as immutable, its value can be modified only by an administrative action and not by its user activity.

UCON is characterized by a number of novelties, stemming mainly from the rest of its components. The component of rights represents a number of privileges that can be held and exercised from a subject to an object. In a similar way to RBAC's roles, the UCON conceptual framework supports hierarchies among rights. Note that rights are not set a priori, but they are determined during the access. The access decision is given from a usage function by considering the following factors of subject and object attributes, authorizations, obligations and conditions. Authorizations in UCON are functional predicates, whose evaluation is used for taking decisions, namely if access to a subject is granted to an object. In a same manner to the usage function, the evaluation of authorizations is based on subject and object attributes, requested rights and a set of authorization rules. Authorizations can be characterized as pre-authorizations or ongoing-

authorizations. The *pre* prefix refers timely before the requested right and the ongoing prefix during the time span of access.

Furthermore, obligations in UCON are used to capture the requirements that must be met from a subject requesting the usage of an object. These are expressed as functional predicates and, as already mentioned, they are used in the evaluation of access both in the usage function as well as with authorizations. Obligations are also divided into *pre* obligations and *ongoing* obligations. The former is used usually for the retrieval of history information and the latter to check whether the requested requirement is fulfilled during the time span of access. Finally, conditions in UCON are used to capture factors that are accrued from the environment of the system. The semantic difference between conditions and other variables, namely authorization and obligation, is that the former cannot be mutable since there is no direct semantic association with subjects.

3.4 Attribute-Based Access Control

Attribute-Based Access Control (ABAC) has gained a significant attention due to the development of distributed systems and networks, such as the Internet, and is considered to be a logical access control methodology [11]. In contrast to RBAC, a standardized ABAC definition is still missing, and thus several have been proposed. However, a set of guidelines are provided by NIST in [11]. ABAC can provide access decisions based on the evaluation of attribute values, policy rules and environment conditions, depending on the particular ABAC definition. One virtue of ABAC compared to other models is that its policies are expressed in terms of attributes without prior knowledge of the subjects and objects in the system. Moreover, subjects and objects in a system may be assigned with attribute values without prior knowledge of policy details. This does greatly simplify authorization management.

The ABAC model consists of the following six categories of elements: *Attributes*, *subjects*, *objects*, *operations*, *policies*, and *environmental conditions*. *Attributes* are characteristics of the *subject*, *object*, or *environment conditions*. *Attributes* may contain information given by a name-value pair, i.e., a tuple of the form: *(Name, Value)*. Both *subject* and *object* attributes are able to support the use of meta-attributes. The latter provides an additional index for referring to groups of subjects and objects per se. Hierarchies in ABAC are intrinsically supported via the meta-attribute functionality. This provides ABAC with the potential to express powerful hierarchies between elements of the same type. A *subject* is usually interpreted as being a user or process that issues access requests to perform operations on objects. Subjects can be assigned with one or more attributes. An *object* can be a system resource for which access is managed by the ABAC system. These could be devices, files, records, tables, processes, programs, networks, or domains containing or receiving information. It can be the resource

or requested entity, as well as any entity on which an operation may be performed by a subject including data, applications, services, devices, and networks. An *operation* is the execution of a function at the request of a subject upon an object. Example of operations include the read, write, edit, delete, copy, execute, and modify commands. A *policy* is the representation of rules or relationships that makes it possible to determine if a requested access should be allowed, given the values of the attributes of the subject, object, and possible environment conditions. An *environment condition* is an operational or situational context in which access requests occur. Environment conditions are detectable environment characteristics. Environment characteristics are independent of subject or object, and may include the current time, day of the week, location of a user, the current threat level, etc. The above definitions subsequently help in the provision of a reference model for ABAC and a formal specification of it.

In the following, a brief description of well-known ABAC frameworks is provided. Access control frameworks may provide useful guidelines when considering the implementation of an access control system. With regards to attribute-based approaches, the Extensible Access Control Markup Language (XACML) and the Next Generation Access Control (NGAC) appear to be the most prominent frameworks. Both provide operations to manage policies, evaluate decisions, enforce policies, etc. XACML and NGAG may facilitate the adoption of attribute-based approaches though the provision of specifications with regards to both functional operations and composition of components (e.g., policy decision point, policy enforcement point). In the following, we provide information on XACML and NGAC, so as to operate as a precursor when considering proposing access control systems applicable in IIoT environments.

3.4.1 Extensible Access Control Markup Language

XACML is an OASIS standard, currently in version 3.0, which provides a framework for deploying ABAC. To achieve this, XACML provides a data-flow model, named the XACML context, and a policy language model. The data-flow model describes the main functional components, e.g., Policy Enforcements Point (PEP), Policy Decision Point (PDP), Policy Authorization Point (PAP), etc. and interactions among them. These are used for accessing repositories – containing policies or attributes – and getting authorization decisions. The XACML context expresses access requests and responses using an XML schema, implemented by the PDP for authorization purposes. The policy language model is used for the specification of access control requirements using attributes in the context of three hierarchical components, i.e., rules, policies, and policy sets.

Apart from the main components, it is also interesting to refer to the terminology differences between the XACML standard and the guidelines on ABAC provided by NIST. It is apparent that despite some terms are expressed differently, both refer to the same concepts. In the following, we briefly refer to

this mapping, as identified in [12]. *Subjects* and *actions* refer to the same concept, in both XACML and ABAC. A *subject* refers to the entity that requests access, and an *action* refers to the performed operation on the requested entity. A *resource* in XACML is mapped to an *object* in ABAC – *resources* or *objects* are entities that a *subject* request to access. The *environment* in XACML is mapped to *environment condition* in ABAC – that is a dynamic factor, independent of *subjects* and *objects*. Lastly, while the term *element* is used in NIST's guidelines document to refer to *subjects*, *objects*, *actions*, and *environment conditions*, the term *category* is used in XACML instead to refer to *subjects*, *resources*, *actions*, and *environments*.

3.4.2 New-Generation Access Control (NGAC)

NGAC is a NIST initiative [12] for standardizing ABAC mechanism. It is able to express and enforce a wide range of policies. Defined in accordance to ABAC to meet its requirements, NGAC uses data/relations and attributes to express policies and deliver capabilities, respectively. It also provides a set of administrative operations and functions for configuring data and enforcing policies. In the following, we provide briefly information on NGAC as described in [12].

Access control data in NGAC includes *elements*, *containers*, and *relations*. An element may be a *user*, an *operation* or an *object*. These maps to ABAC's *subject*, *action*, and *object*, respectively. *User* and *object* attributes are supported through *containers*. The latter are used to administer and formulate attributes and policies. *Containers* are used to associate and group *elements* among them. Similarly, policy class containers are used to provide collection of policies. Attributes in NGAC are used in a similar way to ABAC – they represent characteristics of the *user* or *object*. For example, *user* attributes could express user roles, etc., while an *object's* attributes could express its stored data. A set of basic operations are provided by NGAC to interface with the data of objects, and administrative operations are responsible for the creation of data elements and relations.

Relations in NGAC are used to express access control policies. There is support for four different type of relations, i.e., assignments, associations, prohibitions and obligations. Assignments are used to define membership on containers. This is expressed through a tuple of the following form: *(a, b)* or equivalently $a \rightarrow b$. The semantics are that element *a* is assigned to element *b*.

Associations are used to derive privileges and are expressed as 3-tuples including a user attribute *ua*, a set of access writes *asr,* and a user or object attribute *at*. The latter association is written as *ua – ars – at* with the following semantics: Users in *ua* can execute the *ars* access rights on the policy elements referenced by *at*.

Prohibitions are used to derive privilege exceptions. Three types of prohibitions are supported, i.e., user-deny (*u_deny*), user attribute-deny (*ua_deny*), and process-deny (*p_deny*). Each prohibition is expressed using a 3-tuple including a

user *u*, a user attribute *ua*, and a process *p*, respectively, followed by an access right (*ars*) and a policy element (*pe*). A user-deny prohibition may be of the following form: *u_deny(u, ars, pe)*. The semantics of the latter prohibition is that user *u* cannot execute access rights in *ars* on policy elements in *pe*. In a similar manner, attribute-deny and process-deny are expressed as *ua_deny(a, ars, pe)* and *p_deny(p, ars, pe)*, respectively.

Lastly, obligations are used to dynamically alter an access state. Obligations are expressed as pairs of event patterns *ep* and a response *r* (i.e., sequence of administrative operations). The former consists of conditions, which when evaluated to true causes the response *r* to execute.

4. Access Control in IIoT

As stated already, access control can introduce the appropriate mechanisms in a system to restrict access of legitimate users or processes in it. IIoT can be characterized as a system of systems, and its emerging characteristics, such as automation, adaptation, high heterogeneity of devices, spatial diversity, etc. require revisiting the concept of access control. Although several works have been conducted in the context of IoT environments, access control in IIoT is still a relatively new area of research. In the rest of this section, we refer to the latest achievements in access control and IoT that appear to be prominent for application in an IIoT environment. Yet, we identify research works that has been already conducted in IIoT environments.

An extensive review of access control model and frameworks for IoT is conducted by A. Ouaddah et. al., in [21]. The survey includes approaches proposed within a period of five years, starting of 2011. These approaches can be potentially applicable in IoT/IIoT environments. An interesting outcome of the survey is the compilation of a taxonomy for both access control models and frameworks. An abundance of access control models has been included, yet all of them have been grouped in representative families of models/categories, e.g., ABAC, RBAC, usage control, CapBAC, organizational-based access control models, etc. In the following, we briefly elaborate on individual models that appear to be omitted in [21] and elaborate on generic frameworks – potentially applicable in IIoT environments.

An RBAC model has been proposed in [9], which is applicable in collaborative multi-domain systems. The proposed model (domRBAC) supports all the components of the standard role-based model (ANSI INCITS 359-2004), including support for the core RBAC, hierarchical RBAC, static and dynamic separation of duties. Furthermore, domRBAC is able to enforce access control under secure interoperation, a prerequisite in multi-domain environments.

In [17] an RBAC model is proposed for application in IIoT, considering them to be multi-domain collaborative environments. Specifically, the requirements

under investigation include these of resource sharing and process collaboration. The authors define RBAC policies as an authorization route optimization problem and provide a solution by proposing an algorithm for solving it. Although the proposed solution may provide optimal solutions, its performance may be restrictive in some cases, as stated by the authors (e.g., assuming excessive amounts of devices and roles). It is provided merely as an administrative tool and lacks automation, i.e., it is not applicable in a policy decision point.

An ABAC model is formally defined in [8] in adherence with NIST's recommendations in [11]. The model's main elements that can take part in the authorization process and a description of its main administrative operations and review functions are provided. ABAC approaches intrinsically support highly distributed environments due to context information conveyed through attribute values.

The UseCON model [7, 10] is a next-generation model based on the concept usage control. UseCON is able to support complex and more expressive policies com-pared to existing usage-based approaches (e.g., UCON). Although it is not explicitly defined in the context of IoT/IIoT, its main characteristics, such as continuity of decision and attribute mutability, may render it applicable in industrial environments. Although an implementation of the model is missing, formal proofs have been provided with regards to its internal functions.

Independently of the access control model and its supported policies, a set of functional components are required for an access control mechanism to be instantiated in the context of an access control system architecture. The telecommunication standardization sector of ITU provides in X.812 a recommendation of a security framework, which defines among other the main functions required in open systems to support access control services and mechanisms [28]. Thus, based on X.812 the main functions may include: An initiator (e.g., a user or process), a target (i.e., the resource access is required upon), an Access control Enforcement Function (AEF), and an Access control Decision Function (ADF). The latter is responsible for access control decision making. The decisions are made based on information applied by the access control policy rules, the context in which the access request is made, and Access control Decision information (ADI). ADI is part of the Access Control Information (ACI) function, which includes all the information used for access control purposes, including contextual information. Lastly, the responsibility of AEF is to enforce the decision taken from the ADF.

Following the core idea of X.812, existing access control frameworks as the XACML and the NGAC provide their own set of functions to support X.812 functionality. XACML main functions are a PEP, a PDP, a Policy Information Point (PIP), a PAP and a context handler (CH). Further information about the operations supported by the individual functions is provided in OASIS XACML standard documentation [20].

In a similar manner, NGAC provides its own functional architecture, too. Its main functional components are: At least one PEP; at least one PDP; zero or one

Event Processing Point (EPP); one PAP; one PIP; and one or more Resource Access Points (RAPs). Further information about the operation of the individual functions in NGAC is provided in [12].

It is worth mentioning that although both XACML and NGAC frameworks share some functionality, yet they differ. For example, the PAP, PDP, and PIP appear to provide slightly different functionality in each framework. Differences apply also when it comes to their access decision process, which is logic based in XACML and enumerated in NGAC [12].

5. Components Placement

From the above it is evident that considering an access control architecture for application in an IIoT environment requires a carefully investigation of all its functional components. This will provide – depending on the applied framework (e.g., XACML, NGAC) – indications on where to place each of the functional components in respect to the layers, as depicted in Figure 1. The placement is not just an arbitrary architectural decision since it affects both the functionality and the efficiency of the applied framework in the specific context.

The cloud is an important element in the development of IIoT. It provides a unified, ubiquitous platform for data sharing and can support various applications in the context of IIoT. Alsheri et al. [2] propose a cloud-enabled architecture for access control deployment in IoT. That architecture includes a layered environment that consists of the object layer, the application layer and the in-between middle layer(s). Specifically, the object layer includes the things residing on the edge, whereas the middle layer includes the virtual object and cloud services layers. The virtual objects layer is an abstraction used to provide the constant presence of things including both current and historical information [26].

The cloud services layer provides resources to objects, and finally, the application layer offers an interface to communicate with the objects. In such an approach, the access control decision making is provided by a PDP placed in the cloud layer and the enforcement of access control decisions is performed by a PEP placed on the object layer. Access control administration is performed in the administration layer.

In [24] an ABAC specific deployment is proposed where cloud, fog and edge layers are used for the various components of access control system. Access control administration is provided by a PAP, which is located on the cloud along with a PIP that stores subject, object and system attributes. PDPs are in turn located in various fog nodes and interact with the PAP and PIP in the cloud. Finally, PEP is performed on the edge layer. Integrating PEP on the edge is a challenging task considering all resource limitations and the heterogeneity of objects that renders the consistent enforcement deployment to be a challenging task.

PAP is the term used by ABAC models to describe the entity that is used to create and manage policies of an access control system. Deploying PAP in the cloud makes it available enterprise-wide and it eases any consideration regarding policy exchange between remote federations, provided that all required mechanisms (e.g., authentication) are in place. The same applies to any other model implementation when it comes to policy administration.
PDP on the other hand provides time-critical services since their use is to reach into access control decisions. Access control decision making requires on the one hand resources to allow for quick processing of policies and on the other hand low latency to communicate the decision to enforcement points instantly upon making. Placing PDP on the cloud may not be the most efficient architectural decision, mainly due to the distance between stakeholders and the cloud itself. Extending cloud near the edge though, which is the case when exploiting fog computing, lowers this distance and makes the fog layer the prevalent candidate to host PDPs.

In the ABAC case, which is a suitable model in implementing context-aware access control mechanisms, thus mechanisms which use context to provide relevant information and/or services to the user, where relevancy depends on the user's task [1], required attributes need to be retrieved from PDP to perform access decision. This information is provided to PDP upon request to PIP. To achieve this, PIP should be both aware of all available attributes but also able to both retrieve and deliver attributes in real time without stalling the whole process. Since most attributes are domain-specific in the industrial environment, considering the uniqueness of each deployment, PIP needs to have visibility to the specific domain. To achieve this fog can be utilized and host an additional "local" PIP to provide cloud functionality in close vicinity to the stakeholders.

Apart from communication between PDP and PIP, the former needs to obtain the policies to consider. Having placed PAP on the cloud and PDP in the fog may induce latency or connectivity issues between those. However, given the benefits in policy management that cloud provides, it is a matter of context handler implementation to perform propagation of policies and disconnected decisions.

Access control decisions should be enforced from PEP. The enforcement usually happens in the edge, where stakeholders exist. Considering an industrial environment, main issues in this layer are resource limitation, device heterogeneity and proprietary communication methods. As a result, consistency in PEP deployment is hard to achieve. The mist layer, as introduced for IIoT deployment, can provide the area to deploy PEP.

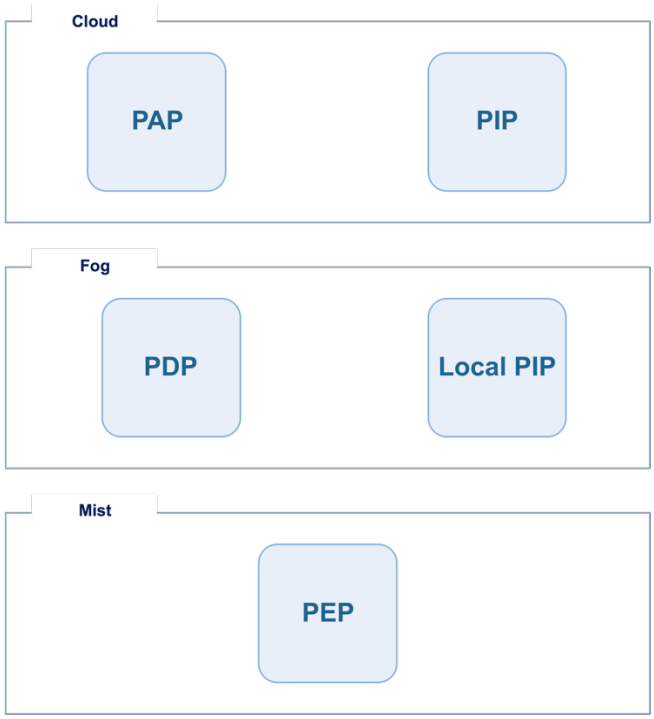

Fig. 2. Components placement

In the IoT reference by ITU [29], edge devices can communicate either directly with the upper layers or through a gateway node. Regarding PEP deployment, it can be either integrated with the device or with the gateway. Integrating PEP in a gateway enables support for joining proprietary or other devices that cannot be natively controlled. Moreover, access control in the industrial environment can heavily rely on mist implementation directly into the edge network fabric [14], thus potentially eliminating any latency or connectivity issues. An indicative component placement is presented in Figure 2.

6. Open Issues and Conclusion

There are still issues to promote further research in the deployment of access control in IIoT, some of which are presented in this section.

In ABAC, stakeholder and contextual attributes are evaluated in order to allow or deny access requests. In an environment like IIoT it is challenging to limit the scope of a domain into a specific area and control interactions with other domains. While RBAC models the definition of inter-domain policies requires to exchange

identities or roles [9], in ABAC based schemes there is a potential unlimited number of attributes that need to be exchanged.

Trust relationships between domains constituting federations, but also between federations, should be established. Moreover, although PIP placement in a domain's fog area is proposed as an effective approach (i.e., for retrieving attribute values), interconnectivity between PIPs and exchanging of attribute values is a matter of further research and analysis.

Communication between access control components should be optimized so that it can be secure and efficient. Working on this direction, communication protocols used in industrial environments, like Constraint Application Protocol (CoAP) or Message Queuing Telemetry Transport (MQTT), can be considered. In any case, communication between system components should be lightweight and reliable, but also ensure the confidentiality and integrity of the exchanged information.

Safety is not directly relevant to computer security. Nevertheless, IIoT is deployed in domains and environments (e.g., factories, warehouses, hospitals, roads) where human life is at risk and may be threatened of undesirable access control decisions derived of misconfigured policies or invalid attribute values. Safeguards, possibly based on machine learning techniques, should be included to protect against system failures or misconfiguration.

Industrial applications heavily rely on system availability. It is a critical factor that should be considered and therefore access control implementation should never threaten it. It is a matter of research to provide safeguards to ensure business continuity in case of access control system failure.

IIoT triggers the fourth industrial revolution. It improves visibility to the context and allows for the deployment of new innovative applications. Nevertheless, industrial systems should be protected against malicious access to ensure business continuity and smooth operation. Access control should be considered and implemented based on the selected model or framework. Thus, a lot of work still needs to be done in terms of formal specification, validation and verification of access control implementations for IIoT.

Access control appears to be a challenging research topic in the context of IIoT. In this chapter, we elaborated on the concept of IIoT and on access control models and frameworks that may be applicable in it. We anticipate these directions to provide interesting multi-disciplinary insights in both industry and academia, and to stimulate further research in this important field of study.

References

[1] Abowd GD, Dey AK, Brown PJ, et al (1999) Towards a Better Understanding of Context and Context-Awareness. In: Gellersen H-W (ed) Handheld and Ubiquitous Computing. Springer Berlin Heidelberg, Berlin, Heidelberg, pp 304–307

[2] Alshehri A, Sandhu R (2016) Access Control Models for Cloud-Enabled Internet of Things: A Proposed Architecture and Research Agenda. IEEE, pp 530–538
[3] Atzori L, Iera A, Morabito G (2010) The Internet of Things: A survey. Computer Networks 54:2787–2805. doi: 10.1016/j.comnet.2010.05.010
[4] Bonomi F, Milito R, Zhu J, Addepalli S (2012) Fog computing and its role in the internet of things. ACM Press, p 13
[5] Daugherty P, Banerjee P, Negm W, Alter AE (2015) Driving unconventional growth through the industrial internet of things. Accenture (https://www accenture com/us-en/_acnmedia/Accenture/next-gen/reassembling-industry/pdf/Accenture-Driving-Unconventional-Growth-through-IIoT pdf Downloaded 15 July 2016)
[6] Ferraiolo DF, Kuhn DR, Chandramouli R (2003) Role-Based Access Control, Artech House. Inc, Norwood, MA
[7] Gouglidis A, Grompanopoulos C, Mavridou A (2018) Formal Verification of Usage Control Models: A Case Study of UseCON Using TLA+. In: International Workshop on Methods and Tools for Rigorous System Design
[8] Gouglidis A, Hu VC, Busby JS, Hutchison D (2017) Verification of Resilience Policies That Assist Attribute Based Access Control. In: Proceedings of the 2Nd ACM Workshop on Attribute-Based Access Control. ACM, New York, NY, USA, pp 43–52
[9] Gouglidis A, Mavridis I (2012) domRBAC: An access control model for modern collaborative systems. Computers & Security 31:540–556. doi: 10.1016/j.cose.2012.01.010
[10] Grompanopoulos C, Gouglidis A, Mavridis I (2013) A Use-Based Approach for Enhancing UCON. In: Jøsang A, Samarati P, Petrocchi M (eds) Security and Trust Management. Springer Berlin Heidelberg, Berlin, Heidelberg, pp 81–96
[11] Hu VC, Ferraiolo D, Kuhn R, et al (2014) Guide to Attribute Based Access Control (ABAC) Definition and Considerations. National Institute of Standards and Technology
[12] Hu VC, Kuhn DR, Ferraiolo DF (2015) Attribute-Based Access Control. Computer 48:85–88. doi: 10.1109/MC.2015.33
[13] Industrial Internet Consortium. Technical papers, publications, and white papers. https://www.iiconsortium.org/white-papers.htm. Accessed 1 June 2018.
[14] Iorga M, Feldman L, Barton R, et al (2018) Fog computing conceptual model. National Institute of Standards and Technology, Gaithersburg, MD
[15] Jin X, Krishnan R, Sandhu R (2012) A Unified Attribute-Based Access Control Model Covering DAC, MAC and RBAC. In: Cuppens-Boulahia N, Cuppens F, Garcia-Alfaro J (eds) Data and Applications Security and Privacy XXVI. Springer Berlin Heidelberg, Berlin, Heidelberg, pp 41–55
[16] Lin S-W, Miller B, Durand J, et al (2015) Industrial internet reference architecture. Industrial Internet Consortium (IIC), Tech Rep
[17] Liu Q, Zhang H, Wan J, Chen X (2017) An Access Control Model for Resource Sharing Based on the Role-Based Access Control Intended for Multi-Domain Manufacturing Internet of Things. IEEE Access 5:7001–7011. doi: 10.1109/ACCESS.2017.2693380
[18] Lopez, J. and Rubio, J.E., 2018. Access control for cyber-physical systems interconnected to the cloud. Computer Networks, 134, pp.46-54.
[19] Navarro-Ortiz J, Sendra S, Ameigeiras P, Lopez-Soler JM (2018) Integration of LoRaWAN and 4G/5G for the Industrial Internet of Things. IEEE Communications Magazine 56:60–67. doi: 10.1109/MCOM.2018.1700625
[20] OASIS Standard. eXtensible Access Control Markup Language (XACML) Version 3.0. http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.html. Accessed 1 June 2018.
[21] Ouaddah A, Mousannif H, Abou Elkalam A, Ait Ouahman A (2017) Access control in the Internet of Things: Big challenges and new opportunities. Computer Networks 112:237–262. doi: 10.1016/j.comnet.2016.11.007
[22] Park J, Sandhu R (2004) The UCON ABC usage control model. ACM Transactions on Information and System Security 7:128–174. doi: 10.1145/984334.984339

[23] Pesonen LIW, Eyers DM, Bacon J (2006) A capability-based access control architecture for multi-domain publish/subscribe systems. IEEE, pp 7 pp. – 228
[24] Salonikias S, Mavridis I, Gritzalis D (2016) Access Control Issues in Utilizing Fog Computing for Transport Infrastructure. In: Rome E, Theocharidou M, Wolthusen S (eds) Critical Information Infrastructures Security. Springer International Publishing, Cham, pp 15–26
[25] Serpanos D, Wolf M (2018) Internet-of-Things (IoT) Systems. Springer International Publishing, Cham
[26] Welbourne E, Battle L, Cole G, et al (2009) Building the Internet of Things Using RFID: The RFID Ecosystem Experience. IEEE Internet Computing 13:48–55. doi: 10.1109/MIC.2009.52
[27] Wilkes MV, Needham RM (1979) The Cambridge CAP Computer and Its Operating System. Elsevier
[28] X.812 : Information technology - Open Systems Interconnection - Security frameworks for open systems: Access control framework. https://www.itu.int/rec/T-REC-X.812/en. Accessed 1 Jun 2018
[29] Y.2060 : Overview of the Internet of things. https://www.itu.int/rec/T-REC-Y.2060-201206-I. Accessed 1 Jun 2018
[30] Zhang X, Nakae M, Covington MJ, Sandhu R (2008) Toward a Usage-Based Security Framework for Collaborative Computing Systems. ACM Transactions on Information and System Security 11:1–36. doi: 10.1145/1330295.1330298