POLITECNICO DI TORINO
Repository ISTITUZIONALE

Toward attribute-based access control policy in industrial networked systems

Original
Toward attribute-based access control policy in industrial networked systems / Cheminod, Manuel; Durante, Luca; Valenza, Fulvio; Valenzano, Adriano. - ELETTRONICO. - 2018-:(2018), pp. 1-9. (Intervento presentato al convegno 14th IEEE International Workshop on Factory Communication Systems, WFCS 2018 tenutosi a ita nel 2018) [10.1109/WFCS.2018.8402339].

Availability:
This version is available at: 11583/2724583 since: 2021-01-28T13:39:15Z

Publisher:
Institute of Electrical and Electronics Engineers Inc.

Published
DOI:10.1109/WFCS.2018.8402339

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright
IEEE postprint/Author's Accepted Manuscript

©2018 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collecting works, for resale or lists, or reuse of any copyrighted component of this work in other works.

(Article begins on next page)

12 October 2024

Toward Attribute-Based Access Control Policy in Industrial Networked Systems

Manuel Cheminod, Luca Durante, Fulvio Valenza, Adriano Valenzano
National Research Council of Italy (CNR–IEIIT), Corso Duca degli Abruzzi 24, I-10129 Torino, Italy
Emails: {manuel.cheminod, luca.durante, fulvio.valenza, adriano.valenzano}@ieiit.cnr.it

***Abstract*—The definition of a correct Access Control Policy is a fundamental step in the design of a secure information system. However, the complexity of modern systems makes critical the choice upon which model to use for such definition. This is becoming particularly true for Industrial Networked Systems, where a correct access control policy must cover all the different and ever evolving interactions between all of its heterogeneous sub-systems at different levels of the production process. In this paper, with the support of an example of a typical industrial system, we highlight the limitations of the well known and widely used Role Based Access Control policy model and we propose an alternative model, built on the ideas of the Attribute Based Access Control model, showing how it can be leveraged to easily define complex access control policies in Industrial Networked Systems. We provide also a preliminary analysis on the kind of conflicts or anomalies that such expressive model can introduce.**

I. Introduction

The design and management of access control policies in modern industrial networked systems is a complex task. In fact, the number of possible interactions between users and system resources is large and continuously growing as INS systems are evolving towards distributed and flexible systems following the recent trends in the industrial world (Industry 4.0).

A correct access control policy must define clearly what kind of accesses each user has to each system resource. It is crucial for this task to select the most appropriate model for the definition of the access control policy. One of the most popular and widely used model is the Role Based Access Control (RBAC) one [1], [2]. This model relies on three main concepts: user, role, permission. Sets of permissions are granted to roles, stating what operations can be performed on which objects. Users are then assigned to one or more role, getting the related permissions. This approach conveniently avoids the impractical task of granting permissions to users in a one by one fashion.

Although much appreciated, this model has showed some limitations [3] that reduce its effectiveness when dealing with large and complex systems.

This work was partially supported by Regione Piemonte and the Ministry of Education, University, and Research of Italy in the POR FESR 2014/2020 framework, Call "Piattaforma tecnologica Fabbrica Intelligente", Project "Human centered Manufacturing Systems" (application number 312-36).

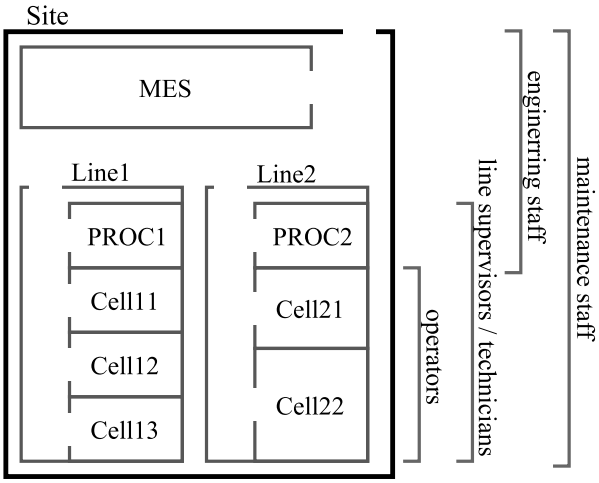

Fig. 1. Physical structure of the industrial plant and user groups activities distribution.

Several extensions have been proposed in literature, focusing on different aspects (e.g.: position, time) but the most investigated and promising model for access control is the Attribute Based Access Control (ABAC) one [4]. This model relies on the identification of several attributes (related to the user, the object, and the environment) and on their evaluation when an access request is triggered.

This is a flexible model as the set of attributes to evaluate can be chosen specifically for the considered system. However, there is not yet a clear and accepted complete definition of an ABAC model for industrial systems.

In this paper we propose a model for the definition of access control policies in industrial systems, following the principles behind ABAC. We identify a set of attributes and we define their evaluation for the identification of permissions to grant to different sets of users.

To motivate our proposal we provide an example of a typical industrial system architecture and we show how a standard RBAC based approach for the definition of access control policies is not efficient enough. Then, as the main contribution of our work, we overcome the highlighted shortcomings by proposing and using a new access control model that is flexible and able to simplify the access control policy design process.

Furthermore, we provide a classification of the potential anomalies and conflicts that can occur in the definition of a

TABLE I
SUMMARY OF OBJECTS AND RELATED OPERATIONS AND PURPOSES.

Type of device	Operations	Purpose
SCADA, database, servers, workstations, HMI	read, write	access to data values (e.g.: production statistics, database entries, procedures documentation)
	program, execute	change running configuration and execute procedures
	login	acquire local access on the device
PLCs. DCSs. robots, motors, sensors	read, write	access device data and parameters
	program, execute	load specific configuration and execute procedures
	diagnostic, test	access diagnostic data and indicators and start testing procedures
Room	enter, exit	enter and exit rooms and other physical environments
Cabinet	open, close	open or close cabinets containing devices

policy with our proposed model.

II. MOTIVATING EXAMPLE

We provide an example of a large distributed industrial plant, that will allow us to highlight the limits of the standard RBAC model and will support our proposal. The design of this synthetic example is inspired by typical industrial system structures. However, the "business" level, or Enterprise Resource Planning (ERP) level, responsible for high level management of resources and customers relations, is intentionally not included in our description to keep the example simple.

A. Physical structure and devices description

The industrial plant is geographically distributed over multiple sites, namely Turin, Milan and Palermo sites. For simplicity, all sites have the same physical structure, depicted in Fig. 1. Each site includes a Manufacturing Execution System ("MES" in figure) area and two production lines. At the MES level the production activities are scheduled and distributed among different production lines ("Line1", "Line2"). Each production line, in turn, prepares some partial products and includes a common process area ("PROC1","PROC2") that coordinates the execution of all the tasks of the production process between cells of production ("Cell11", "Cell12", "Cell13",...). Each cell executes the instructions provided by higher levels of the system (e.g.: recipes from MES), elaborates the ingredients and produces the partial products that are finally recombined at the line level.

Fig. 1 includes the physical connections between the different environments so that, for instance, physical access to a specific cell "Cell11" is possible only by first entering the site, then the production line building ("Line1") and finally accessing the cell passing through its specific gate.

Having provided the main structure of each site area, we can now describe what kind of devices are typically deployed in each zone starting from the production cells, which include devices, such as motors, sensors, robotic arms, production belts, able to manipulate, move and combine ingredients in order to compose partial products. The process area, instead, includes those devices that control the process execution, that is PLCs and DCSs. Moreover, in the process areas are deployed also workstations equipped with specific software that allow technicians and engineers to access device data and configurations. Process and cell areas usually include also many Human Machine interfaces (HMI) that expose field data and operations to the operators in those areas. Moreover, controller devices (PLCs) are often protected by some physical enclosure such as cabinets. Finally, the MES area includes high level devices that gather and elaborate production data and push commands (recipes) to the lower levels. These devices include SCADA, production servers, databases (e.g. historians) and engineering workstations.

Each object (either a device or a physical building) provides a specific set of operations. It is possible to identify the common operations by type of device. For instance a user can enter or exit a building, while he can open or close cabinets. For what concern devices in the line and MES areas, they usually provide some diagnostic and testing features, some programming operations for deploying specific configurations, and some specific operations to start and stop the execution of their tasks. Workstations, instead, provide login operations to gain logical accesses on them and to execute the installed applications.

We summarize the distribution of type of devices in the different areas of our example as in the following:

- MES: SCADA, databases, servers, workstations.
- PROC{1,2} (in each line): PLCs, DCSs, HMIs, workstations.
- Cells (in each line): robots, HMIs, motors, sensors. Notably, "Cell13" does not include any robot.

Moreover, the network infrastructure is not explicitly shown here but it is implicitly defined as interconnecting all resources within a site and all sites among each other.

Each device exposes one or more operations to interact with it. Operations can be either general ("read" some data) or very specific ("load configuration 2"). In our example we identify some general operations ("read","write","program", and so on) to represent all possible operations.

Tab. I summarizes theses operations by specifying their names and purposes for each type of considered device.

B. Groups of users

In any complex system with large staff and several types of tasks to be performed, it is a common practice to identify groups of users by means of some common characteristics.

TABLE II
HIGH LEVEL DESCRIPTION OF SECURITY REQUIREMENTS

User group	Description and main responsibilities	Required permissions and limitations
Turin/Milan/Palermo employees	Identify employees in different sites.	Can pass through the gates (enter) of their respective sites only.
Operators	control production process execution through interactions with HMIs in cells	Access read/write operations of HMIs in cells. Access limited to a specific cell within a production line.
Line supervisors	Can act as an operator and manages the execution of the production flow from the process areas.	Includes operator's permissions. Can also login and execute applications in workstations in process areas. Can execute procedures on PLCs and DCSs. Accesses limited to the process and cells of a specific line.
Technicians	perform preparations and set-up operations on devices in cells and process areas	Can access test, program, diagnostic operations of each devices in process and cell areas. Remote access limited to the specific production line they are accessing from.
Engineering staff	Manages the production at the process and MES levels.	Has full access to devices in the process and MES area. Program operations are restricted to devices within the MES of the same site he is operating from.
Maintenance staff	Perform periodic maintenance tasks and checks on various devices.	Must access diagnostic and test operations on devices throughout all the plant. Only physical tests operations are allowed.

As an example, we base our group selection criteria on two aspects: the site the employee belongs to, and the areas of responsibility he/she has in the plant.

We define the first obvious group that includes all the users as the *Employees* group. Naturally, other three groups related to the specific site each employee belongs to, are: *Turin employees*, *Milan employees*, *Palermo employees*. Of course, each user included in one of these groups is also included in the *Employees* one.

Then we identify groups of users by their areas of responsibility, that span over different or multiple levels of the production process (from the physical level up to the management one). In our example, the involved areas are those related to the physical operations, the process control, and the MES. These logical areas are here directly matched by the physical structure of the plant. For this reason, the areas of responsibility for the different identified groups are also showed in the right side of Fig. 1. These groups are: *operators*, *line supervisors*, *technicians*, *engineering staff*, *maintenance staff*.

Operations flow in the cell areas are controlled by *operators*. The same tasks can be performed by *line supervisors*, which, however, are also responsible for some operations at the process level, spanning over multiple cells.

Technicians prepare and test the configurations of devices in the lines, that is in process and cell areas. Instead, the *engineering staff* group shares the same responsibilities of technicians for what concerns the process area but extend their activities over the MES areas for the scheduling and planning of the entire production process of a specific site.

Finally, *maintenance staff* is responsible for the maintenance of machines and facilities. For this reason, activities of the maintenance staff span over the entire physical site and involve specifically some testing operations on devices.

It is worth noting that here we presented a small set of groups to keep the description simple. However, many other group selection criteria can be used. For instance, the level of experience of employees can determine groups such as *junior*, *senior*, *expert*. Moreover, in general, each user can belong to multiple groups depending on his/her attributes.

C. Security requirements

The next step in the description of our example is the definition of the security requirements that will drive the design of the system access policy. Security requirements are identified for each group of users.

The responsibilities identified in the previous section describe what some groups of users are supposed to do, while security requirements further refine what they shall and shall *not* do. This will lead to the detailed identification of the permissions (that is, access to operations) each user (group) should be granted with.

Operators work inside the production cells, interacting with the machines, mostly through HMI interactions. They should have access only to the devices in close proximity, that is in the same cell they are working in. Line supervisors, instead, are responsible for the regular execution of the production process throughout their entire production line. As such, they can control the execution of procedures of the devices in the process and cells areas.

Technicians can setup devices and should access all the diagnostic and test features of every device in the cell areas. Moreover, sometimes technician must access the programming features of the devices in the cells, in order to fine tune some parameters or activate specific configurations. However, this kind of operations should be performed only from the workstations that are available in the process area related to the involved cells (that is, within the same line of production).

The engineering staff of any plant site, instead, mainly operates at the MES level, deploying production plans on all the controllers of the different lines (within the same site). As such, they have complete access to controllers, scada, database workstations and HMIs in MES and process areas. However, programming features should be constrained to be accessed

only from the MES within the same site of the controlled device.

Maintenance usually refers to maintenance of machines and facilities. In this example we consider only maintenance of machines which involves some regular checks (diagnostic and test features) on devices to identify possible risks of breakage. As for technicians, access to test features should be restricted to physical operations, excluding remote accesses through the network.

Overall, access to sites should be limited to personnel which belong to that site. For instance, access to Turin site is granted to Turin employees (technicians, operators, and so on).

Tab. II summarizes the different responsibilities and related permissions associated to each user group. This schema is used as a guide to define the policy for the system.

D. Policy design with RBAC

After having defined the main tasks and related permissions, it is possible now to design the access policy for the described industrial system. Initially, we follow the RBAC approach by first identifying the roles and then assigning permissions to them.

Following the schema of Tab. II we start by the identification of the following roles: *Operator*, *Line supervisor*, *Technician*, *Engineer*, *Maintenance*, *Employee*.

Related to the site distribution of personnel, we define three roles: *Turin employee*, *Milan employee*, *Palermo employee*.

One way to implement the requirements related to the physical access to different sites, it is possible to assign the permission to enter the Turin site to *Turin employee* role only. So that a user u_1 assigned to the role *Technician* can then be assigned also to the role *Turin employee* gaining access to Turin site only.

The role *operator*, instead, is assigned with a set of permissions to access read/write operations on HMIs in cells. However, it is not directly possible to specify that operators in "Cell11" have access only to HMIs in "Cell11" and so on. For this specific requirement we should define a special role for each production line so to have, for instance, "operator for Cell11" and "operator for Cell12" which are assigned, respectively, access to HMIs in "Cell11" and HMIs in "Cell12". The same happens when we consider further lines and sites.

The role *line supervisor* includes the permissions of *operator* or, in terms of hierarchical RBAC, inherits from *operator*. However, the definition of this role suffers from the same difficulties highlighted for the "operator" role. Indeed, restricting accesses to a specific area is not possible without duplicating and further specifying ad-hoc roles (in this case, this means to define a "line1 supervisor" different from "line2 supervisor" and so on).

The definition of the *technician* role with respect to the *engineer* one raises another kind of difficulty. An *engineer*, in fact, *partially* includes the technician permissions as he/she does not need (and is not provided with) permissions to access devices in production cells. The *engineer* role cannot be defined to inherit from the *technician* one. Even though some permissions are shared, these are to be explicitly and separately assigned to both roles. Or, willing to leverage the hierarchical RBAC features, it is possible to define a *engineer-technician* role to which are assigned the common permissions and to define both *engineer* and *technician* role to inherit from this ad-hoc role. However, this kind of approach is not flexible nor easily maintained, introducing unwanted complexity in the policy model.

Finally, the *maintenance* role is allowed to execute test operations that are accessed in a specific way (physical interaction in this case). It is not possible within the RBAC model to specify this class of operation and, for this reason, the definition of the *maintenance* role permissions requires an explicit enumeration of all such operations and related objects throughout all the plant sites. Of course, in large systems this kind of approach is not very efficient.

Moreover, the problem of the explicit enumeration of all operation and object pairs to assign as permissions is often recurring when following the RBAC approach. The concept of role, in fact, allows to define groups of users but no similar concept exists to define groups of objects or operations.

Finally, when a user performs an access request, that is he/she asks the system to perform an operation on a specific object, some contextual information of this request should be considered. For instance, the location from which the user triggers the request is relevant and this location can be either physical (a specific room or site) or a logical one (from an host in a specific subnetwork). Standard RBAC model does not include such information and, as showed here, is not flexible enough.

III. Policy Specification in ABAC

In the previous section we highlighted some of the aspects that reduce the suitability of the RBAC model in the definition of complex access control policies. Here we propose a different model able to express complex security requirements in a simple and compact way.

Access control policy are specified at a high level of abstraction, several model are available today to design and describe the access control policy.

As specified in the RFC 3192 [5], a policy is a set of rules to administer, manage, and control access to network resources.

As already introduced before, the most common and used model for the access control policy is Role Base Access Control (RBAC) [1], [2]. In RBAC a rule r is specified by the triple:

$$r = (ro, op, ob)$$

where:

- ro is a role in the INS;
- op is an operation in the INS;
- ob is an object of the INS;

It is worth noting that, in RBAC, each rules identifies an *allowed* action. In fact, there are no negative rules in RBAC.

As we showed in the previous section, in many industrial networked systems, the RBAC model is not flexible enough, on the contrary the Attribute Base Access Control (ABAC) paradigm [6], [4] is more powerful.

The new policy specification model that we present is based on ABAC paradigm, that simplifies the administration work and improves the policy expressiveness with respect to our previous policy model based on the RBAC [7], [8].

Basically, we embrace the ABAC idea of specifying access requests by means of several attributes defined for users and objects. In particular, we use some attributes to identify *sets* of users and objects that share some characteristics.

In this new model a policy rule r is a tuple:

$$r = (US, OP, OB, ac)$$

where

- US is a set of users that work in the INS;
- OP is a set of operations that can be executed on some object in the INS;
- OB is a set of objects in the INS;
- ac specifies the action (i.e. *allow* or *deny*).

The clear advantage of our policy model, is that each policy rule can be referred to set of users, operations and objects, while in the RBAC model each policy rule is associated to a set of users (i.e. the user with role specified in the policy rule), but is specified for only one specific pair of operation and object.

Referring on ABAC paradigm where each user, operation and object are characterized by several attributes, in our model we can simply identify, in a query like form, a specific set of users, operations and objects by the specification of same attributes. This approach simplifies the policy specification work, permitting to reduce the number of policy rules and allowing the administrator to define more restrictive policies.

We indicate the set of all users within the system with $\mathcal{U}$, the set of all operations with $\mathcal{OP}$ and the set of all objects with $\mathcal{OB}$.

Specifically, a user u is characterized by two different attributes: *Id* and *Group*. The values of these attributes for the specific user u are indicated, respectively, with $u.id$ and $u.gp$. In this way, it is possible to define a set of users US_1, by specifying:

- a set of user Ids ID_1;
- a set of user Groups GP_1 (e.g. Turin employees, technicians, operators);

$$US_1 =< ID_1, GP_1 >$$

In other word, the set of user US_1 includes all the user with an Id included in ID_1 and that belong to one of the groups specified in GP_1.

$$US_1 = \{\mathcal{U}_{|ID_1} \cap \mathcal{U}_{|GP_1}\}$$

where:

$$\mathcal{U}_{|ID_1} = \{u|u.id \in ID_1\}$$

$$\mathcal{U}_{|GP_1} = \{u|u.dv \in GP_1\}$$

We use the symbol '*' to represent all the elements in a specific set. For example, we identify users with the group *Turin employees*, with the user set $US_1 =< *, \{\text{Turin employees}\} >$. Obviously the set $\mathcal{U}$ of all users is equal to the set $US_1 =< *, * >$, that we shortly represent with $US_1 = *$.

An operation op, instead, is characterized by the specification of three different attributes: *Label*, *Access mode* and *Location*. Their values for the specific operation op are indicated, respectively, with $op.lb$, $op.ac$, $op.lo$. The label represents the name of the operation itself. The *Access mode*, instead, describes the type of access to the operation. In our model we include two types: physical, remote. An operation is accessed physically if there is some physical interaction between the user and the object. For instance, the interaction with a touch panel of an HMI. A remote operation, instead, is accessed through some network connection between the user and the object providing the specific operation (e.g.: a remote access to a diagnostic web page of a PLC). Finally, the location indicates *from where* the access request is performed. It is worth noting that, although not modelled here, all existing locations defined in our model are included in a hierarchical structure that includes the notion of physical inclusion of environments (e.g. an object in "Cell11" is also included in the "Line1" location).

In this way, a set of operations OP_1, can be specified by indicating:

- a set of operation labels LB_1 (e.g. enter, login, test);
- a set of access modes AC_1 (e.g. physical, remote);
- a set of locations LO_1, from where the access to the operation is performed (e.g. home, Turin, Line1);

$$OP_1 =< LB_1, AC_1, LO_1 >$$

That means:

$$OP_1 = \mathcal{OP}_{|LB_1} \cap \mathcal{OP}_{|AC_1} \cap \mathcal{OP}_{|LO_1}$$

where:

$$\mathcal{OP}_{|LB_1} = \{op|op.lb \in LB_1\}$$

$$\mathcal{OP}_{|AC_1} = \{op|op.ac \in AC_1\}\}$$

$$\mathcal{OP}_{|LO_1} = \{op|op.lo \in LO_1\}$$

As for the user group model, we use the symbol '*' to represent all the elements in a specific set. For example, we identify all the operations performed at "home", with the operations set $OP_1 =< *, *, \{\text{home}\} >$. Obviously, the set $\mathcal{OP}$ all the operations is equal to the set $OP_1 =< *, *, * >$, shortly represented with $OP_1 = *$.

Finally, an object ob is characterized by the specification of three different attributes: *Id*, *Type*, and *Location*. Indicated, respectively, with $ob.id$, $ob.ty$ $ob.lo$. In this way, a set of objects OB_1, can be selected by indicating:

- a set of object ids ID_1;
- a set of object types TY_1 (e.g. Room, Server, PLC);

$r_1 : (< *, \text{Turin employees} >, < *, *, * >, < *, *, \{\text{Milan, Palermo}\} >, deny)$
$r_2 : (< *, \text{Milan employees} >, < *, *, * >, < *, *, \{\text{Turin, Palermo}\} >, deny)$
$r_3 : (< *, \text{Palermo employees} >, < *, *, * >, < *, *, \{\text{Milan, Turin}\} >, deny)$
$r_4 : (< *, \{\text{Operators,Line supervisors }\} >, < \{\text{read,write}\}, \text{physical}, \text{Cell11} >, < *, \text{HMI}, \text{Cell11} >, allow)$
$r_5 : (< *, \{\text{Operators,Line supervisors }\} >, < \{\text{read,write}\}, \text{physical}, \text{Cell12} >, < *, \text{HMI}, \text{Cell12} >, allow)$
$r_6 : (< *, \{\text{Operators,Line supervisors }\} >, < \{\text{read,write}\}, \text{physical}, \text{Cell13} >, < *, \text{HMI}, \text{Cell13} >, allow)$
$r_7 : (< *, \{\text{Operators,Line supervisors }\} >, < \{\text{read,write}\}, \text{physical}, \text{Cell21} >, < *, \text{HMI}, \text{Cell21} >, allow)$
$r_8 : (< *, \{\text{Operators,Line supervisors }\} >, < \{\text{read,write}\}, \text{physical}, \text{Cell22} >, < *, \text{HMI}, \text{Cell22} >, allow)$
$r_9 : (< *, \text{Line supervisors} >, < \{\text{login,execute}\}, *, \text{Line1} >, < *, \text{workstation}, \text{Line1} >, allow)$
$r_{10} : (< *, \text{Line supervisors} >, < \{\text{login,execute}\}, *, \text{Line2} >, < *, \text{workstation}, \text{Line2} >, allow)$
$r_{11} : (< *, \text{Line supervisors} >, < \text{execute}, *, \text{Line1} >, < *, \{\text{PLCs,DCSs}\}, \text{Line1} >, allow)$
$r_{12} : (< *, \text{Line supervisors} >, < \text{execute}, *, \text{Line2} >, < *, \{\text{PLCs,DCSs}\}, \text{Line2} >, allow)$
$r_{13} : (< *, \text{Technicians} >, < \{\text{test,program,diagnostic}\}, *, \text{Line1} >, < *, *, \text{Line1} >, allow)$
$r_{14} : (< *, \text{Technicians} >, < \{\text{test program,diagnostic}\}, *, \text{Line2} >, < *, *, \text{Line2} >, allow)$
$r_{15} : (< *, \text{Engineering staff} >, < \text{program}, \text{remote}, \text{MES} >, < *, *, \{\text{Proc1}, \text{Proc2}, \text{MES}\} >, allow)$
$r_{16} : (< *, \text{Engineering staff} >, < \text{program}, \text{remote}, * >, < *, *, \{\text{Proc1}, \text{Proc2}, \text{MES}\} >, deny)$
$r_{17} : (< *, \text{Engineering staff} >, < *, *, * >, < *, *, \{\text{Proc1}, \text{Proc2}, \text{MES}\} >, allow)$
$r_{18} : (< *, \text{Maintenance staff} >, < \text{test}, \text{physical}, * >, < *, *, * >, allow)$
$r_{19} : (< *, \text{Maintenance staff} >, < \text{diagnostic}, *, * >, < *, *, * >, allow)$
$r_{def} : (< *, * >, < *, *, * >, < *, *, * >, deny)$

Fig. 2. Resulting policy model for the provided example.

- a set of object locations LO_1 (e.g. home, Turin, Line1);

$$OB_1 =< ID_1, TY_1, LO_1 >$$

That means:

$$OB_1 = \mathcal{OB}_{|ID_1} \cap \mathcal{OB}_{|TY_1} \cap \mathcal{OB}_{|LO_1}$$

where:

$$\mathcal{OB}_{|ID_1} = \{ob|ob.id \in ID_1\}$$

$$\mathcal{OB}_{|TY_1} = \{ob|ob.ty \in TY_1\}\}$$

$$\mathcal{OB}_{|LO_1} = \{ob|ob.lo \in LO_1\}$$

The set $OB_1 =< *, PC, \{\text{Turin}, \text{Palermo}\} >$, for example, specifies all the resources of types PC in the Turin and Palermo sites.

A. Example

Having defined our policy specification model based on ABAC paradigm, we can described how the security requirement described in the example of Section II can be specified using this model, and the advantages with respect to the RBAC paradigm.

The complete list of resulting rules is included in Fig. 2, for simplicity these rules are referring only to one of the sites (i.e. Turin, Milan, Palermo), so that "Cell12" refers to the second cell in the Line1 structure of that site.

In addition, in this policy system the order of rules correspond to their priority, that is the first rule as the top priority.

The first three rules r_1, r_2, r_3, implement the security requirements specified for all the employees of the three different sites. Specifically, these rules permit to limit the access of employees to their respective sites, i.e. Turin employ can not *work* on Milan and Palermo offices, Milan employ can not *work* on Turin and Palermo and Palermo employ can not work on Milan and Turin.

These type of requirements are implemented in a RBAC paradigm by the definition of many rules, one for each object in the three sites.

The rule from r_4 to r_8 and the default rule implement all the security requirements specified for the Operators and partially implement the requirements specified for the Line supervisors (as Line supervisors include operator's permissions). These rules permit to the Operators and Line supervisors (`<*,{Operators,Line supervisors}>`) to access, only physically, read and write operations (`<{read,write},physical,Cell11>`) on HMIs of a specific Cell (`<*,HMI,Cell11>`). In addition, these rules permit to perform operations on HMIs if the Operators and Line supervisors perform the action inside the Cell (i.e. remote operation are denied). The RBAC paradigm does not support the possibility to specify this last requirement, because in RBAC is not possible to specify where the operations are performed with respect to where the objects are.

The rule form r_9 to r_{12} complete the implementation of the other requirements specific for the Line supervisors, while the rules r_{13} to r_{14}, from r_{15} to r_{17} and from r_{18} to r_{19} respectively enforce the requirement specified for Technicians, Engineering staff and Maintenance staff.

It is worth noting how rules r_{13}, r_{14} restrict the accesses for the technicians. In fact, these rules specify that a well defined set of three operations are allowed on any device in "Line1" only if they are performed from "Line1" itself. The same for devices in "Line2". This kind of rules leverage both the specification on the source of the access request ($op.lc$) and the location structure of our model. In fact, by specifying "Line1" the rule includes "Proc1" and "Cell1{1,3}" but, obviously, not

"Proc2" and other cells.

Rules r_{15}, r_{16}, r_{17} show another way to limit accesses within specific environments. The Engineering staff, in fact, has complete access on any device in the process and MES areas in their site (rule r_{17}). However, the programming features of the devices can be executed *remotely* only from the MES area. This is specified by rule r_{15}. Any other remote access (for instance from "Proc1" to "Proc2") is denied (by rule r_{16}). The specification of objects and operations by means of their types and locations is not possible in RBAC but effectively simplifies the management of the policy rules set. In fact, each one of the rules specified in our example would result in a large number of rules in RBAC model.

IV. Conflict Analysis

The use of access control policy simplifies the work of a security administrator, however the administrator must give special attention to its specification. The policies have to be well specified in order to exactly implement the security requirements and in large industrial systems this is a very complex and error-prone task. Moreover, when the industrial networked system contains a large number of policy rules, the possibility of writing anomalous policy is high.

In this section, we provide an analysis on different anomalies that may exist among the ABAC policy rules that we define in our model. This preliminary analysis is quite different to the existing work on anomalies analysis in access control policy [9], [10], [11] and it is based on the main works proposed in literature on firewall policy analysis [12], [13], [14]. Specifically we identify two main classes of anomalies: *Intra-Rule* and *Inter-Rule*. The first one identifies anomalies within a single policy rule, while the former identifies anomalies between two different rules.

A. Intra-Rule

An Intra-Rule anomaly occurs when there is an erroneous specification of a policy rule so that it will never be applied. We identify two different types of intra-rule anomalies: *Irrelevancy* and *Inconsistency*.

1) Irrelevancy Anomaly: A rule is irrelevant when one of the sets in the tuple is an empty. That is:

$$US = \emptyset \ \vee \ OB = \emptyset \ \vee \ OP = \emptyset$$

This type of anomaly typically occurs when the system resources change but the system policy is not correctly updated. Removal of an irrelevant rule does not change the policy behaviour and increases the system performance (an hight number of irrelevant rules can reduce the performance of system).

Referring on Section II, an example of irrelevancy anomaly is represented by policy r_i:

$$r_i : (< *, \text{Technicians} >, < *, *, * >, < *, *, \text{Madrid} >, allow)$$

The rule r_i is an old rule, defined by the administrator when industrial system had a site in Madrid city. Clearly this rule is irrelevant because the object set specified in the policy is empty.

2) Inconsistency Anomaly: A rule is inconsistent when either due to temporary state or permanent mismatch between the rule conditions (i.e. user, object and operation set) and the system resources there is no access request that could be matched, that is, again, the rule will never be applied.

A very simple example of inconsistency anomaly, is represented by a policy r_i:

$$r_i : (< *, \text{Turin Employees} >, < \text{login}, *, * >,$$
$$< *, \text{Room}, \{\text{Milan, Palermo}\} >, deny)$$

The rule r_i is inconsistent because the administrator erroneously specified *login* instead of *enter*.

B. Inter-Rule

When some policy rules are correlated (i.e. they share some user, operation or object), an access request may match more then one policy rule. In this case, the system selects the policy rule to enforce by applying a resolution strategy. The most used resolution strategy is the First Matching Rule (FMR) one, that selects the action of the first applicable policy rule in an ordered list. For this reason an inter-rule policy anomaly occurs when two correlated policy rules are not correctly ordered.

In the following, we formally describe four types of inter-rule policy anomalies: *Shadowing*, *Correlation*, *Duplication* and *Redundancy*, using as an example two policy rules r_i and r_j:

$$r_i = (US_i, OP_i, OB_i, ac_i)$$
$$r_j = (US_j, OP_j, OB_j, ac_j)$$

1) Shadowing Anomaly: A policy rule r_i is shadowed by rule r_j (i.e. $sha(r_i, r_j)$) when:

1) r_i and r_j specify different actions;
2) the priority π of r_j is higher than r_i (r_j precedes r_i);
3) all access requests that match r_i also match r_j.

$$sha(r_i, r_j) \leftrightarrow \pi(r_j) > \pi(r_i) \wedge US_i \subseteq US_j \wedge$$
$$OP_i \subseteq OP_j \wedge OB_i \subseteq OB_j \wedge ac_i \neq ac_j$$

In this case r_i will never be applied.

For example, considering the scenario described in Section II, let's assume that the security administrator is requested to implement a new policy requirement: "Operators can not perform any write operation on HMI devices throughout Cell13". In order to implement this requirement the administrator can specify a new policy rule r_i:

$$r_i : (< *, \text{Operators} >, < \text{write}, *, * >,$$
$$< *, \text{HMI}, \text{Cell13} >, deny)$$

In this case, if the administrator puts the policy r_i after the rule r_6, he/she introduces a shadowing anomaly in the system.

2) Correlation Anomaly: Two policy rule r_i and r_j are correlated (i.e. $cor(r_i, r_j)$), when:

1) r_i and r_j specify different actions;
2) some access requests that match r_i also match r_j;
3) some access requests that match r_i do not match r_j and vice versa.

$$cor(r_i, r_j) \leftrightarrow US_i \cap US_j \neq \emptyset \wedge$$
$$OP_i \cap OP_j \neq \emptyset \wedge OB_i \cap OB_j \neq \emptyset \wedge$$
$$ac_i \neq ac_2 \wedge \overline{sha(r_i, r_j)} \wedge \overline{sha(r_j, r_i)}$$

For example, in the case the security administrator has to implement a new policy: "The Turin employees can not perform any tests on devices in the MES areas", in order to implement this requirement, he can specify a new policy rule r_i:

$$r_i : (<*, \text{Turin Employees}>, <\text{test}, *, *>,$$
$$<*, *, \text{MES}>, deny)$$

This new rule is correlated with the rule r_{18}. In this case, in order to ensure a correct policy behaviour with this new policy requirement, the administrator must put the policy r_i before the rule r_{18}, otherwise all the Maintenance staff of Turin site would be allowed to perform physical tests on any devices in the MES area.

3) Duplication Anomaly: A policy rule r_i duplicates rule r_j (i.e. $dup(r_i, r_j)$) when:

1) r_i and r_j specify the same action;
2) the priority π of r_j is lower than r_i (r_j comes after r_i);
3) all access requests that match r_i also match r_j and vice versa.

$$dup(r_i, r_j) \leftrightarrow \pi(r_i) > \pi(r_j) \wedge US_i = US_j \wedge$$
$$OP_i = OP_j \wedge OB_i = OB_j \wedge ac_i = ac_j$$

In this case r_j will never be activated and the removal of r_j will not affect the overall policy behaviour in any way.

4) Redundancy Anomaly: A policy rule r_i is redundant with respect to rule r_j (i.e. $red(r_i, r_j)$) when:

1) r_i and r_j specify the same action;
2) the priority π of r_i is higher than r_i (r_i precedes r_j);
3) all access requests that match r_i also match r_j;
4) it does not exist a rule r_z correlated with r_i with a priority between r_i and r_j;

$$red(r_i, r_j) \leftrightarrow \pi(r_i) > \pi(r_j) \wedge US_i \subseteq US_j \wedge$$
$$OP_i \subseteq OP_j \wedge OB_i \subseteq OB_j \wedge ac_i = ac_j \wedge$$
$$\nexists r_z \;\; cor(r_i, r_z) \wedge \pi(r_i) > \pi(r_z) > \pi(r_j)$$

In this case removal of r_i will not change the policy behaviour.

For example, if the security administrator adds at the end of rules list, a new policy rule r_i:

$$r_i : (<*, \text{Maintenance staff}>, <*, *, *>, <*, *, *>, deny)$$

This new rule is redundant with respect to the default rule.

On the contrary rule r_{15} is *not* redundant with respect to rule r_{17} because rule r_{16} is correlated to rule r_{15} and makes false the fourth condition.

V. Related Works

Access control policy specification models have evolved from Discretionary Access control (DAC) [15] and Mandatory Access control (MAC) [16] into the widely used Role-Based Access control (RBAC) [1], [2] and, more recently, into the Attribute-Based Access Control (ABAC) [6], [4].

In literature, there is not a standard and widely accepted ABAC model. There are, instead, several proposed models that focus on different aspects and different kind of attributes [17]. For instance, [18] proposes a logic-based framework for the specification and evaluation of policy rules but lacks the inclusion of object attributes. [19] introduces a flexible model with attribute and object hiearchies but does not deal with rule anomalies or conflicts. Other models are focused on specific domain aspects, such as in [20] where trust and privacy issues are introduced in the proposed model. Finally, in [21] a "role-centric" ABAC model is introduced, in which "roles" still drive the permissions assignment and attributes are used to determine roles assignment to user. This kind of "hybrid" model aims at introducing attributes in already existing policy systems but, as a consequence, limits the flexibility of a "pure" ABAC model.

In general, the ABAC flexibility comes with a cost, as it introduces new challenges deriving from possible errors in the policy specification. For this reason many works on policy analysis have been proposed in literature to address these problems.

Policy analysis, in general, is a process that analyses and checks a set of policies for possible violations of some security properties [22]. An example of policy analysis is the so called reachability analysis that evaluates allowed communications within a computer network (i.e. determines if a certain host can reach other ones) [23]. An other research area on policy analysis is the anomaly analysis that detects inconsistencies in the specifications, that can arise when two or more policies, defined by the administrators, lead to contradictory outputs [24].

In literature many works have been proposed on anomaly analysis of access control policy, and the most interesting and relevant ones are [9], [10], [11].

As previously described in Section IV, with respect to our work no one of these works consider, in their analysis, the order of policy rules, and consequentially they do not consider and classify different type of anomalies (i.e. they only identify macro types of anomaly).

In [10] Lin *et al.* propose a complete work on analysis of policy similarity. Specifically Lin *et al.* propose *EXAM* an environment tool that using a SAT-solver and MTBDD.

In [11] Shaikh *et al.* specify two types of anomalies: *Inconsistencies* and *Incompleteness*. Inconsistencies occur when two policy rule sets lead to direct contradictory conclusion. *Incompleteness* anomaly, instead, exists when no rule is defined for some situations. These first kind of anomalies is analysed in our model in the inter-rule anomalies.

Finally, in [9] Fisler *et al.*, introduced for the fist time the change-impact analysis. This analysis consumes two policies that span a set of changes and summarizes the differences between the two policies. Fisler *et al.* implement this type of anomaly analysis using the Multi-Terminal Decision Diagrams (MTBDD).

VI. Conclusions and Future works

In this paper we highlighted some of the limitations of the RBAC model when applied to the definition of access control policies in complex systems. To overcome such problems we proposed a new access control model, following the main ideas behind the ABAC model. In particular we showed how the proper selection of attributes for objects and users and their access requests, simplifies the definition of complex access control rules. Of course, our approach is general enough to be applicable also with different sets of attributes that can better suit different systems. We presented a typical industrial system that, even if kept small for clarity, is already complex enough to demonstrate how a more flexible model, such as the one presented here, is useful and can overcome some of the limitations of the RBAC model. However, we do recognize that a more powerful language for model specification can introduce subtle errors or inefficiencies in the specification itself if not carefully applied. For this reason, we identified and defined the possible types of anomalies and conflicts that can arise among different rules. The next steps in the complete definition of our approach are in two directions: increasing the flexibility and capabilities of the specification language (including, for instance, time related attributes) and, accordingly, extending the definition of the possible anomalies and conflicts while also providing a specific resolution strategy to avoid or solve these issues.

Moreover, the formal definition of our model and related analysis will be supported by the development of software tool able to perform semi-automatic validation and analysis of provided policies.

References

[1] *Role Based Access Control*, ANSI INCITS 359-2012, 2012.

[2] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman, "Role-Based Access Control Models," *Computer*, vol. 29, no. 2, pp. 38–47, 1996.

[3] D. R. Kuhn, E. J. Coyne, and T. R. Weil, "Adding attributes to role-based access control," *Computer*, vol. 43, no. 6, pp. 79–81, June 2010.

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

[5] A. Westerinen, J. Schnizlein, J. Strassner, M. Scherling, B. Quinn, S. Herzog, A. Huynh, M. Carlson, J. Perry, and S. Waldbusser, "Terminology for Policy-Based Management," RFC 3198 (Informational), Internet Engineering Task Force, November 2001, http://www.rfc-editor.org/rfc/rfc3198.txt. [Online]. Available: http://www.ietf.org/rfc/rfc3198.txt

[6] V. C. Hu, D. Ferraiolo, R. Kuhn, A. R. Friedman, A. J. Lang, M. M. Cogdell, A. Schnitzer, K. Sandlin, R. Miller, K. Scarfone *et al.*, "Guide to attribute based access control (abac) definition and considerations (draft)," *NIST special publication*, vol. 800, no. 162, 2013.

[7] M. Cheminod, L. Durante, L. Seno, and A. Valenzano, "Semiautomated Verification of Access Control Implementation in Industrial Networked Systems," *IEEE Trans. Ind. Informat.*, vol. 11, no. 6, pp. 1388–1399, 2015.

[8] M. Cheminod, L. Durante, L. Seno, F. Valenza, and A. Valenzano, "Automated fixing of access policy implementation in industrial networked systems," in *2017 IEEE 13th International Workshop on Factory Communication Systems (WFCS)*, May 2017, pp. 1–9.

[9] K. Fisler, S. Krishnamurthi, L. A. Meyerovich, and M. C. Tschantz, "Verification and change-impact analysis of access-control policies," in *Proceedings of the 27th International Conference on Software Engineering*, ser. ICSE '05, 2005, pp. 196–205.

[10] D. Lin, P. Rao, E. Bertino, N. Li, and J. Lobo, "Exam: a comprehensive environment for the analysis of access control policies," *International Journal of Information Security*, vol. 9, no. 4, pp. 253–273, Aug 2010.

[11] R. A. Shaikh, K. Adi, and L. Logrippo, "A data classification method for inconsistency and incompleteness detection in access control policy sets," *International Journal of Information Security*, vol. 16, no. 1, pp. 91–113, Feb 2017.

[12] E. Al-Shaer, H. Hamed, R. Boutaba, and M. Hasan, "Conflict classification and analysis of distributed firewall policies," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 10, pp. 2069–2084, Oct 2005.

[13] C. Basile, A. Cappadonia, and A. Lioy, "Network-level access control policy analysis and transformation," *IEEE/ACM Trans. Netw.*, vol. 20, no. 4, pp. 985–998, Aug. 2012.

[14] F. Valenza, S. Spinoso, C. Basile, R. Sisto, and A. Lioy, "A formal model of network policy analysis," in *Proceedings of the 1st IEEE International Forum on Research and Technologies for Society and Industry (RTSI 2015)*, 2015, pp. 516–522.

[15] J. Wang, H. Zhai, P. Li, Y. Fang, and D. Wu, "Directional medium access control for ad hoc networks," *Wireless Networks*, vol. 15, no. 8, p. 1059, Feb 2008.

[16] D. E. Bell and L. J. LaPadula, "Secure computer systems: Mathematical foundations," MITRE CORP BEDFORD MA, Tech. Rep., 1973.

[17] D. Servos and S. L. Osborn, "Current research and open problems in attribute-based access control," *ACM Comput. Surv.*, vol. 49, no. 4, pp. 1–45, Jan. 2017.

[18] L. Wang, D. Wijesekera, and S. Jajodia, "A logic-based framework for attribute based access control," in *Proceedings of the 2004 ACM Workshop on Formal Methods in Security Engineering*, ser. FMSE '04. New York, NY, USA: ACM, 2004, pp. 45–55.

[19] X. Jin, R. Krishnan, and R. Sandhu, "A unified attribute-based access control model covering dac, mac and rbac," in *Data and Applications Security and Privacy XXVI*, N. Cuppens-Boulahia, F. Cuppens, and J. Garcia-Alfaro, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 41–55.

[20] J. Zhu and W. W. Smari, "Attribute based access control and security for collaboration environments," in *2008 IEEE National Aerospace and Electronics Conference*, July 2008, pp. 31–35.

[21] X. Jin, R. Sandhu, and R. Krishnan, "Rabac: Role-centric attribute-based access control," in *Computer Network Security*, I. Kotenko and V. Skormin, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 84–96.

[22] F. Valenza, T. Su, S. Spinoso, A. Lioy, R. Sisto, and M. Vallini, "A formal approach for network security policy validation," *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications*, vol. 8, no. 1, pp. 79–100, 2017.

[23] C. Basile, D. Canavese, C. Pitscheider, A. Lioy, and F. Valenza, "Assessing network authorization policies via reachability analysis," *Computers and Electrical Engineering*, vol. 64, pp. 110 – 131, 2017.

[24] F. Valenza, C. Basile, D. Canavese, and A. Lioy, "Classification and Analysis of Communication Protection Policy Anomalies," *IEEE/ACM Transactions on Networking*, pp. 1–14, 2017. [Online]. Available: http://ieeexplore.ieee.org/document/7967691/