Journal of Ubiquitous Systems & Pervasive Networks
Volume 3, No. 1 (2021) pp. 00-00

Access Control Metamodels: Review, Critical Analysis, and Research Issues

Nadine Kashmar [a]*, Mehdi Adda [a], Hussein Ibrahim [b]

[a] *Université du Québec à Rimouski, Rimouski, Canada, QC G5L 3A1*
[b] *Institut Technologique de Maintenance Industrielle, Sept-Îles, Canada, QC G4R 5B7*

Abstract

The new generation of networking environments such as the internet of things (IoT), cloud computing, etc. is emerging and releases new prospects to traditional information systems by merging new technologies and services for seamless access to information sources at anytime and anywhere. Concurrently, this emergence opens new threats to information security and new challenges to controlling access to resources. To ensure security, several techniques have been employed, and access control (AC) is one of the essential security requirements especially for recent networking environments. Various authentication and AC methods are proposed to enforce AC policy and to prevent any unauthorized access to logical/physical assets. The continuous technology upgrades and the diversity of AC models force the need to find AC metamodels with a higher level of abstraction that serves as a unifying framework for specifying any AC policy. AC metamodels are proposed to encompass AC features and are used to derive various instances of AC models and methods. In this paper we review the proposed AC metamodels and their implementation scenarios, we analyze them, their objectives, their limitations, and present current research issues and open questions that still need to be addressed.

Keywords: *Access control, metamodels, IoT, Industry 4.0, security and privacy, security policy*

1. Introduction

The importance of security, data protection, and privacy requirements increases with the massive presence and integration of new paradigms and technologies, such as cloud computing and the Internet of Things (IoT), also with the deployment of digital and intelligent solutions based on the industry 4.0 concept [1, 2]. To contain and mitigate the impact of cyberattacks, several techniques have been employed, and access control (AC) is one of the essential solutions for privacy settings to measure and optimize IT security [3] in IoT [4], cloud computing [5], social networks [6] and other fields. Access control methods are implemented to control what users can access, when, and how by enforcing AC policy to prevent any unauthorized access for logical or physical assets. In any organization (or industry sector) there might be different types of policies such as: password policy, network access policy, remote access policy, etc., they are defined by managers and system administrators based on the rules and the guidelines of the organization.

To enforce organizational policies, various AC models are developed such as are Discretionary Access Control (DAC), Mandatory Access Control (MAC), Role-Based Access Control (RBAC), Organization Based Access Control (OrBAC), and Attribute-Based Access Control (ABAC) [7–9]. To enhance AC methods, various hybrid models are implemented by combining features of two or more AC models. Despite the advantages of the common AC models in controlling access in various computing environments, they also have various limitations. Moreover, with the emergence of highly dynamic environments, especially with the concept of industry 4.0 and IoT applications, it is realized that AC models (also hybrid models) have reached their limits. They no longer meet the increasing demand for privacy and security standards with the widespread of devices and resources [9]. This reality urges the need to find more advanced AC methods and develop AC metamodels with advanced features for specifying and enforcing different AC policies [10–12]. AC metamodels

*Corresponding author. Tel.: +14188338800
Fax: +141883311; E-mail: nadine.kachmar@gmail.com

DOI: 10.5383/JUSPN.03.01.000

are used to derive various instances for the common AC models, hybrid models, and other AC methods. Note that, in [8] we present a preliminary survey for the commonly used AC models with some proposed AC metamodels, then raise some questions in this domain. Fig.1 summarizes the aim of AC metamodels.

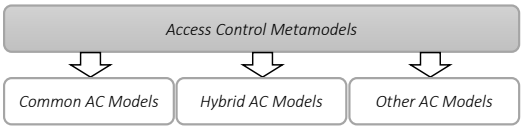

Fig. 1. The aim of AC metamodels

The objective of this paper is to present a literature review and investigate the state-of-the-art of AC metamodels, find out their limitations in the presence of new technologies, and determine the various research issues in this domain, then raise some essential research questions. This review can be considered as a step towards developing a new generic and dynamic AC metamodel with advanced features for IoT and non-IoT systems. In this paper, we provide a detailed literature review for the existing AC metamodels with discussion and critical analysis. The contribution of this paper can be summarized as follows:

- Reviewing recent studies of AC metamodels by providing a summary of the objectives of each study.
- Analyzing and criticizing the proposed AC metamodels.
- Explaining their limitations and why they are not effective in the presence of new technologies and for future upgrades.
- Determining different research issues in this domain and raise some essential research questions.

The remainder of this paper is organized as follows: Section 2 summarizes the existing AC models. Section 3 presents the state-of-the-art of the proposed metamodels, their objectives, and their limitations. Discussion and critical analysis and common limitations for the proposed metamodels are presented in section 4. Current research issues and open questions are proposed in section 5. Section 6 concludes this paper with future perspectives.

2. Access Control Models

In any computing environment subjects request permission (read, write…) to access some objects (file, class…). For this purpose, the defined AC policy that is formally represented by an AC model is enforced to control what objects a subject (user) can access when and how. A subject is allowed to perform some operation(s) on an object or denied accessing this object based on the defined access rights that are granted to him. An access right or a privilege definition might have the form (u, ar, o), which means a subject (u) has an access right (ar) to an object (o), another defined form is (ar, o), a capability of u or referred to as permission of u [7, 13]. AC policies might have the following form:
Allow/Deny doctors, nurses, etc. to… and…
if… and/or… Except…when…

2.1. The Common Access Control Models

Access control is the process of restricting access to a place or resource based on a defined set of security policies. Security policies are the definition of rules that must be regulated in an organization, and they are usually defined by managers and system administrators. An AC model is a framework for making authorization decisions based on the defined AC policies, and an AC mechanism is the process of enforcing AC policy and translating user's access request [7, 8]. Despite the presence of several papers reviewing the state-of-the-art of the common AC models [8, 14], in this paper, we summarize them since their features are used in building different AC metamodels.

2.1.1. Discretionary Access Control (DAC)

DAC model was first introduced in the 1960s. The system protection notion includes three major components: objects, subjects, and permission. DAC is defined as a user-centric model where a file owner controls permissions that are given to other users requiring access to that file. The AC rights of subject(s) over object(s) are specified by Access Control Matrix (ACM). Other ACM variations include Capability Lists (CLs) and Access Control Lists (ACLs). Lampson and Harrison Ruzzo Ullman (HRU) are two variants of DAC model. It is very flexible to assign access rights between subjects and objects, and it is provided with operating systems to authenticate system administrators and users using some procedures, for example, passwords [7, 8].

2.1.2. Mandatory Access Control (MAC)

MAC model was presented in the 1970s. In MAC, users cannot define AC rights by themselves, AC policy is managed in a centralized manner. It is based on the concept of security levels associated with each subject and object where permissions and actions are derived. These levels have hierarchical and nonhierarchical components. Hierarchical components include unclassified, confidential, secret, and top-secret types. Nonhierarchical components represent a set of categories where labels are used to indicate security levels for objects classification and subjects clearance. Its key components are a set of objects, a set of subjects, permissions, and security levels. Bell and LaPadula (BLP) and BIBA (Kenneth J. Biba) are two MAC variants [7–9].

2.1.3. Role-Based Access Control (RBAC)

RBAC was proposed in 1992 as an alternative approach to MAC and DAC. It is based on several entities: users, roles, permissions, actions, operations, and objects. A role is a group of permissions to use object(s) and perform some action(s), it can be associated with several users. Also, users can be assigned to several roles (e.g., doctor). The aim of RBAC is to facilitate the administration of AC policy, it controls user's access to information through roles for which a user is authorized to perform [7–9]. RBAC example can be represented in the hospital system where there exists a variety of relations between doctors, nurses, etc. Only the system administrator has the right to control system security and assign roles to users [15]. Flat RBAC (RBAC0), Hierarchical RBAC (RBAC1), Constrained RBAC (RBAC2), and Symmetric RBAC (RBAC3) are RBAC variants [9].

2.1.4. Organization-Based Access Control (OrBAC)

OrBAC model was first presented in 2003 to solve some problems in DAC, MAC, and RBAC, by finding a more abstract control

policy. Each organization is comprised of a structured group of subjects having certain roles or entities. OrBAC exceeds the concept of only granting permissions to subjects, it also addresses the concept of prohibitions, obligations, and recommendations. A role may have a permission, prohibition, or obligation to do some activity on some view given an associated context. OrBAC is composed of seven entities that are distributed in two levels: the role, activity, and view are found in the abstract level, and the subject, action, and object entities in the concrete level; the context lies between the two levels to express dynamic rules [8, 13].

2.1.5. Attribute-Based Access Control (ABAC)

This is the latest AC model development, its concepts have paralleled that of RBAC. It has the ability to support dynamic attributes and its benefits in managing authorizations. It has three types of attributes: object, subject, and environmental (e.g., the current time, day of the week, etc.) attributes. It allows or denies user requests based on some attributes for users, objects, and environment, and a set of policies that are specified in terms of those attributes and conditions. It is dynamic since it uses attributes to determine access decisions, and subjects are enabled to access a wider range of objects without specifying individual relationships between each subject and each object. AC permissions are evaluated at the time of the actual user's request which offers a larger set of possible combinations of variables to reflect a larger set of possible rules to express policies. Two standards that widely address the ABAC framework are: The Extensible AC Markup Language (XACML) and Next Generation AC (NGAC) with AC facility for applications and other important features [7, 8].

Fig. 2 summarizes the historical evolution of common AC models. Also, various models extensions are proposed in the literature to enhance their features along with the technology progressions, for example, Integrity-OrBAC (I-OrBAC) [16] and Multi-Organization Environments called Trust-OrBAC [17] are two OrBAC extensions, a Higher-order Attribute-Based Access Control Model (HoBAC) [18] is an ABAC extension, and others.

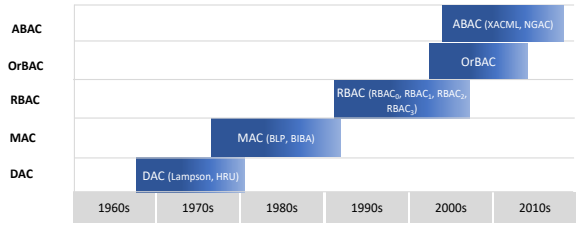

Fig. 2. Historical Evolution of common AC Models

2.2. Enhancing Features of Access Control Methods

The need to use enhanced AC methods imposes the necessity to find models with combined features from two or more models called hybrid AC models. Various hybrid AC models are presented in the literature, for example, hybrid RBAC and ABAC.

In RBAC it is difficult to set up an initial role structure in rapidly changing environments also it does not support dynamic attributes, Kuhn et al. in [19] address the idea of adding attributes to RBAC. The aim is to find a model that supports dynamic attributes, especially in organizations to handle relationships between roles and attributes to provide better AC features in dynamic environments. As well, Rajpoot et al. in [20] propose Attribute Enhanced RBAC (AERBAC) model to enhance features from both RBAC and ABAC because both have complementary features to each other. Moreover, in [21] authors state that the integration of RBAC and ABAC still have some shortcomings in terms of AC flexibility and decision efficiency. For this purpose, they propose a more fine-grained, flexible, and efficient RABAC (RBAC/ABAC) model. To increase the flexibility of RBAC, an Emergency RBAC (E-RBAC) approach is proposed in [22]. In [23], an ABAC scheme integrated with controlled access delegation capabilities for collaborative e-Health environments is proposed.

2.3. Some Limitations of the Common AC Models

Table 1 summarizes the limitations common AC models [7, 8].

Table 1. Limitations of the common AC models

Model	Limitation(s)
DAC	• in large systems granting permissions between subjects and objects are time consuming and difficult to manage. • granted user allow others to read a file without asking the owner.
MAC	• security levels assignment places limits on user actions which prevents dynamic modification of original policies. • is difficult to implement due to dependence on trusted components.
RBAC	• poor support for dynamic attributes (e.g., time of day) • in large systems role inheritance and the need for customized privileges make administration potentially heavy.
OrBAC	• poor support for dynamic attributes (e.g. time of day). • inflexible in rapidly changing IT environments. • it has some vulnerabilities to some kinds of attacks. e.g. covert channels.
ABAC	• its implementations require significant time to run. • often not possible to compute the set of users that may have access to a given resource. • difficult to efficiently calculate the resulting set of permissions for a given user.

3. Access Control Metamodels

Access control models must consider the continuous developments and changes to answer the needed security requirements. The new technology trends (cloud computing, IoT, social networks…), the variety of platforms and applications, users' types, etc. reflect the difficulty of controlling secure and private access to the needed resources in different areas. All this makes AC models and even combining some of them (hybrid models) are insufficient to handle the needed target. This fact forces the need to find models with a higher level of abstraction, called AC metamodels, that serve as unifying frameworks for specifying and enforcing any AC policy [8, 24]. However, metamodels are presented in the literature to concurrently handle multiple AC models. Different AC models can be derived as special instances from the same metamodel.

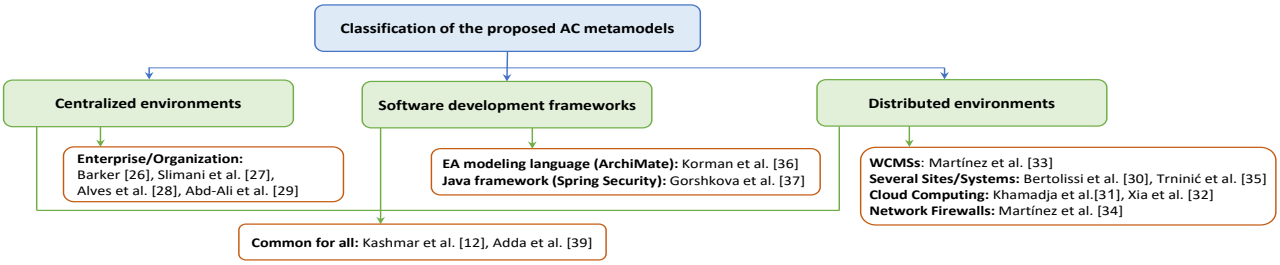

Fig. 3. Classification for the proposed AC Metamodels

3.1. General Definition of Metamodel Concept

The metamodel is defined as textual, graphical/visual, or formal representation of concepts in a certain domain and how they are linked together, these concepts might be rules, guidelines, etc. for an institution or organization. Moreover, metamodeling is defined as the modeling of a model to describe the permitted structure to which models must adhere. Also, models and metamodels need adaptable supporting tools due to changing requirements and policies. There are different metamodeling tools and languages such as Unified Modeling Language (UML), Eclipse Modeling Framework (EMF), ArchiMate, MetaEdit, etc. [8].

3.2. State-of-the-Art

Several AC metamodels are proposed for centralized computing environments, distributed computing environments, and for software development frameworks (Fig. 3). To the best of our knowledge, there is a limited number of recent works proposed in this domain. In this paper, we review them within a decade, analyze them to find if they are effective to follow technology upgrades. Ferraiolo et al. in [25], revise some concerns and raise some questions related to AC policy enforcement and focuses on the important role a metamodel might play when achieved.

3.2.1. Centralized Environments

To address the questions raised in [25], a paper published by Barker [26] demonstrates that multiple models can be derived as special cases from a defined AC metamodel called Category Based Access Control (CBAC) metamodel. A category is interpreted as a synonym for a role, a class, a group, security levels, etc. where entities (e.g., subjects) may be assigned. CBAC metamodel includes features of MAC, DAC, and RBAC where a wider range of constraints may be expressed based on it. Barker demonstrates that the presented AC models in the literature are based on a limited and small number of primitive notions. These notions are related to the concept of categories, relationships between categories and between categories and principals, and modalities. However, AC primitives are given a more general interpretation to allow developing many AC models by combining the primitives of AC models, hence a wider range of constraints may be expressed.

In [27], Slimani et al. extend Barker's metamodel to support resource and action hierarchies. They propose a Unified Access Control Modeling Language (UACML) to provide support for hybrid AC policies by allowing categories to be associated with other categories and finding hierarchical relationships between them. A CBAC metamodel extension is proposed by Alves et al. in [28] to expand a general notion of obligation for the existing AC models and study the interaction between obligations and permission. The aim of their approach is to allow security administrators to check the consistency of a policy combining authorizations and obligations.

Furthermore, Abd-Ali et al. in [29] propose an integration metamodel for hybrid policies to concurrently handle multiple models. Their idea is based on the concept of abstracting each AC model (e.g., RBAC metamodel), then including a special element named DecisionHandler to determine AC decision. The AC decision depends on more than one AC metamodel (CW metamodel, BLP metamodel …). Their approach depends on the idea of clustering the DecisionHandler instances of a hybrid policy, then apply them to combining algorithms (ComAl) to find one AC decision as output in response to multiple AC decisions as input. The integration of several AC models is based on a tree structure of AC decision systems named Ascending Decision Tree (ADT). ADT nodes are DecisionHandler instances or ComAl nodes. ADT has a unique root node and the decision it returns is the decision of the whole tree carrying out the hybrid AC policy.

3.2.2. Distributed Environments

Another approach based on CBAC metamodel is proposed by Bertolissi et al. in [30] for distributed environments that consist of several sites. A system of several sites might be composed of several policies at each site, and in the distributed metamodel the request can be passed to other sites and evaluated in a distributed manner. They demonstrate the expressive power of their metamodel by showing how a distributed, dynamic, event-based access control model (DEBAC) can be defined as an instance of the metamodel. In the context of cloud computing, saving data on cloud servers by cloud users raise security challenges to protect sensitive data. In [31] authors states that the classical AC models (DAC, MAC …) are not adequately expressive for highly flexible and dynamic environments. For this purpose, they present a metamodel approach for cloud computing services called Category Based Access Control (CatBAC) framework, it has two stages at the different organization sites by considering the local constraints of each site. The first is achieved by the cloud provider (abstract stage), and the second is by network administrators (concrete stage). In the abstract level categories are connected to express authorizations and are named abstract authorizations. The concrete level represents AC decisions in relation to concrete level entities, which are subject, resources, action, and context, and are called concrete authorizations. Hence, this AC metamodel allows security administrators in the various company sites to find a concrete model with the constraints and specificities of each site. Xia et al. in [32] propose another metamodel approach to handle

security and privacy in cloud service development and operations, called the Cloud Security and Privacy Metamodel (CSPM). CSPM is proposed to address security and privacy in cloud services via integrating and extending the existing metamodels of cloud security together with newly added concepts.

Moreover, an approach is presented for web services by Martínez et al. in [33] to the representation of Web Content Management System (WCMS) AC policies to ease the analysis and manipulation of security requirements by abstracting them from vendor-specific details. Although AC methods are integrated with most WCMS systems (e.g., Wordpress, Drupal …), some limitations still exist. For this purpose, the authors' aim to raise the level of abstraction of the AC implementation to be represented according to a vendor-independent metamodel. They propose a WCMS metamodel inspired from the RBAC concept, its abstract representation is developed using Model-Driven Engineering (MDE). The aim of their approach is to automatically extract the AC information in the domain of WCMSs. Also, Martínez et al. in [34] propose a model-driven approach to extract network AC policies enforced by firewalls within a network system. Their concept tackles the problem of filtering the traffic of a network with the presence of several filtering rules due to several firewalls. They suggest raising the level of abstraction of the information contained in the firewall configuration files, hence the AC policy would be easier to understand, analyze and manipulate. A model-driven approach is proposed to extract a model of the AC policy enforced by the firewalls within a network system, it consists of host and connection entities. The former represents a network host, e.g., IP address, and the latter represents connections between hosts, where the port and the protocol are specified to establish connections and specify if the connection is allowed or denied.

Trninić et al. in [35] present a generic AC management infrastructure for a broad set of systems, to provide a general method for specifying AC rules for different AC models. Their approach is based on models at three different abstraction levels defined by Meta-Object Facility (MOF) classification. The AC policy metamodel is defined at level M2 and used to derive different AC models at level M1 (e.g., RBAC). At level M0, PolicyDSL is used to specify concrete AC policies in a system. Their proposed metamodel is a Domain-Specific Language (DSL) with the syntax that is dynamically adapted to system features that are being modeled. Hence, a security expert would be able to express AC policies for a given AC model using the generated DSL.

3.2.3. Software Development Frameworks

Due to the lack of security features in software development frameworks, some metamodel extensions are proposed. In [36], a unified metamodel as a prospective extension for ArchiMate is proposed, the common Enterprise Architecture (EA) modeling language. The aim is to support the development of enterprises by extending their abilities to model authorization and AC in their architectures. They propose an extension to an established EA modeling language. The metamodel is developed based on the conceptual model of ABAC because of its ability to include most of the other AC models, then mapped to ArchiMate to enrich its existing models. Also, Gorshkova et al. in [37] introduce a fine-grained AC model and provide a metamodel extension for the Spring Security framework to meet modern security requirements. Spring Security is one of the major market players of open source security frameworks for Java. Gorshkova et al. focus on the implementation of authorization frameworks with Java applications, their proposed framework defines a fine-grained extension of RBAC.

3.2.4. Any Computing Environment

The proposed metamodels reflect the importance of constructing more robust AC models in all computing environments, especially with the presence of heterogeneous technologies and platforms [38]. For this purpose, we propose a new generic AC metamodel approach in [12], it includes all AC models features by unifying a common set of AC concepts which can be used to instantiate the needed components and derive various instances of different AC models; also it can be used as a base to construct other essential metamodel features (section 5). Our approach is proposed for all computing environments and its components can be integrated with frameworks to support AC features. In the same way, Adda et Aliane proposed in [39] a generic ABAC AC model that is suitable for all computing environments.

Table 2 summarizes the proposed AC metamodels and their features.

4. Discussion and Critical Analysis

As shown in Table 2, AC metamodels are constructed based on some features of AC models where various models instances can be derived from them. They are defined as textual or visual, and some of the used tools are UML, Eclipse, and Java. Some of the used modeling languages are xtext, spring expression, etc. However, based on the historical evolution of AC methods, Fig.4 illustrates the era of developing AC metamodels. Some meta-

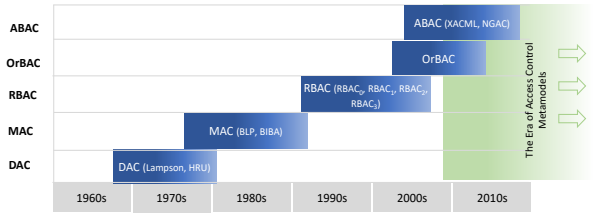

Fig. 4. The Era of Access Control Metamodels

models are proposed as generic, unifying, hybrid, and metamodel extensions for different distributed, centralized environments, and software development frameworks. Hence:

1- Some AC metamodels are constructed based on features of some AC models, and the only AC model(s) (also hybrid) instance(s) that can be derived are the one(s) that are employed in the core structure, for example, [27] and [29]. These metamodels are proposed as *Hybrid Metamodels*.
2- Some frameworks (for example, Drupal, ArchiMate, Spring Security, Network Firewalls ...) are extended to support AC features of one or more AC models, and the extracted AC policies belong to the model(s) that are used to extend the main framework, for example, [33, 34, 36, 37]. These metamodels are proposed as *Metamodel Extensions*.
3- Some AC metamodels are constructed based on a general notion that encompasses some (or all) AC features for some (or all)

Table 2. Summary of the Proposed Access Control Metamodels

ref.	Author	Year	Proposed for	Metamodel	Visual rep. Y/N	Visual rep. Tool	Type	Based on	instance(s)	Modeling lang.
Proposed AC metamodels for Centralized Environments										
[26]	Barker	2009	Enterprise	Barker's Metamodel	No	n/a	Unifying Metamodel	CBAC	RBAC,MAC	Rule/Logic Language
[27]	Slimani et al.	2011	Enterprise	UACML Metamodel	Yes	UML	Hybrid Metamodel	CBAC and Hybrid models	Group based, MAC, RBAC, hybrid model	Object constraint language (OCL)
[28]	Alves et al.	2014	Enterprise	Obligations in CBAC Metamodel	No	n/a	Metamodel Extension	CBAC	CBAC	rewrite-based operational semantics
[29]	Abd-Ali et al.	2015	Enterprise	Integration metamodel	Yes	UML	Hybrid Metamodel	CW,BLP,BIBA, RBAC	Hybrid models	First-order logic
Proposed AC metamodels for Distributed Environments										
[30]	Bertolissi et al.	2014	Distributed system of several sites	Distributed Metamodel	No	n/a	Generic Metamodel	CBAC	CBAC	rewrite-based operational semantics
[31]	Khamadja et al.	2013	Cloud Computing	CatBAC metamodel	Yes	UML	Generic Metamodel	CBAC	Hybrid models	First-order logic
[32]	Xia et al.	2018	Cloud services	cloud security & privacy (CSPM)	Yes	UML	Metamodel Extension	n/a	n/a	UML
[33]	Martinez et al.	2013	WCMSs	WCMS Metamodel	Yes	MDE	Metamodel Extension	RBAC	RBAC	UML
[34]	Martinez et al.	2012	Network Firewalls	Network Connection	Yes	Eclipse	Metamodel Extension	Network Firewalls	RBAC, OrBAC	Xtext
[35]	Trninić et al.	2013	Set of systems	PolisyDSL	Yes	UML	Generic Metamodel	n/a	RBAC	Textual DSL
Proposed AC metamodels for Software Development Frameworks										
[36]	Korman et al.	2016	Enterprise Architecture framework	Unified Metamodel	Yes	ArchiMate	Metamodel Extension	DAC,BLP,Biba, CW,RBAC, ABAC	DAC,BLP,CW, RBAC,ABAC	ArchiMate
[37]	Gorshkova et al.	2017	Enterprise application framework	Spring security framework	Yes	Java-ORM	Metamodel Extension	RBAC	RBAC	Spring expression lang.(SpEL)
Proposed AC metamodels for any Computing Environment										
[39]	Adda et al.	2020	any computing environment	Generalization of ABAC	Yes	UML	ABAC Metamodel	ABAC	ABAC models	UML
[12]	Kashmar et al.	2021	any computing environment	Generic with unified set of AC concepts	Yes	UML	Generic Metamodel	common AC models	common models & hybrid models	UML

models. Based on this metamodel, AC model instance(s) can be derived, for example [12, 26, 30, 31, 35]. These metamodels are proposed as *Generic Metamodels*.

4- Some of the existing AC metamodels are augmented with additional features to reflect a larger and more definitive set of possible rules to express AC policies, for example, [28, 32, 39]. This type of metamodels is proposed as *Metamodel Extensions*.

Hence, the proposed works of AC metamodels in the literature can be classified into two concepts:

- In (1) and (3) the aim is to find a generic metamodel that encompasses most AC features where various AC models (and hybrid models) can be derived, Fig. 5 illustrates the idea of generic metamodels. With regard to this definition of generality, the existing AC metamodels are not generic[1], they have a hybrid structure with some AC features rather than a generic metamodel. This hybrid structure is employed to derive some AC models where their features are employed in the core structure. As shown in Fig. 5, if the metamodel includes features of DAC, MAC, and RBAC models, then the instances that can be derived are DAC, MAC, RBAC, and their combinations (hybrid models based on the existing features, e.g., hybrid MAC/ RBAC).
- In (2) and (4) the aim is to enhance features of the existing frameworks/metamodels by extending them to support AC features and express more AC policies, Fig. 6 illustrates the idea of metamodel extension where AC features are added to the core metamodel/framework to allow defining (more) AC policies. But the structure of the proposed AC metamodels is not extended, for example, no new components or attributes are defined, but AC features are added to the core metamodel structure. As shown in Fig. 6, AC features are implemented and added to an existing AC metamodel or framework to enhance its features and allow defining more AC policies. Then, the extended AC metamodel (or framework) can be used to derive various instances of AC models based on the features which are added to the core metamodel (or framework) structure.

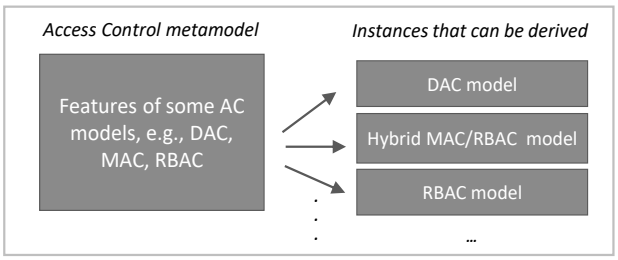

Fig. 5. Illustration for the concept generic metamodel

[1] Except the proposed metamodel in [12] since it includes most of the features of common models

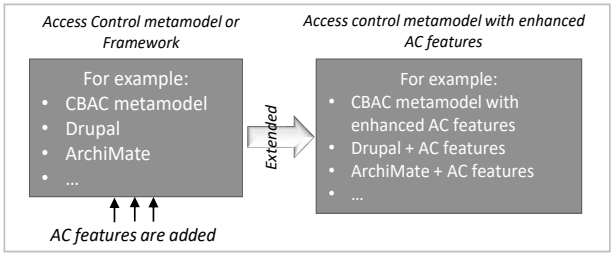

Fig. 6. Illustration for the concept of metamodel extension

However, the presented AC metamodels come with some advantages, and several combined features from AC models are implemented to enhance AC methods. But they also have several limitations especially in the light of new technologies, sections 4.1 and 4.2.

4.1. Limitations of the proposed AC metamodels

In this section, we highlight the limitations for each of the proposed AC metamodels to check out their effectiveness in the presence of new technologies. However, the proposed metamodel extension for Drupal framework in [33] is RBAC-inspired, it is for web contents and it is well known that such environment is rich of variants (time, system updates …), in this metamodel extension the notion of variable attributes is not considered. Although authors in [30] provide a comprehensive theoretical description for their approach which is considered generic with no real case studies are explained or implemented. Hence, their proposed metamodel is still within the theoretical frame. Also, Khamadja et al. in [31] propose CatBAC metamodel to support various AC models in Cloud with no case study or testing result. In [31] and [32] authors have not explained or mentioned how access can be controlled in the context of several heterogeneous clouds (multi-clouds). In [31], authors mention that their proposed solution does not completely consider dynamic constraints, and this important issue should be considered to provide a general method for specifying AC rules for different AC models. Korman et al. in [36] present some of their metamodel limitations, such as the proposed approach misses the concept of logging, and the difficulty for potential implementation of automated analytical capabilities of the unified metamodel. In [27] and [29] the proposed metamodels are based on the concept of combining some models then instantiate one or more AC model(s) based on a hybrid structure, hence they are general templates to derive some AC models that are employed in the core structure rather than a metamodel. Barker's approach [26] lacks the support of resource hierarchies and action hierarchies which are useful to specify high-level access rules [27]. The extension of CBAC metamodel in [28] is proposed to accommodate a general notion of obligation, authors adjust the notion of events and describe a set of core axioms for defining obligations with some examples to specify dynamic policies. Nevertheless, they have not explained how their approach could be dynamic in distributed contexts that are rich in events and variable attributes. The proposed metamodel extensions in [33, 34, 36, 37] tackle specific projects or frameworks to support some AC features without explaining how these extensions can be upgraded due to unexpected updates or changes, especially in distributed environments. The model proposed in [39] is limited to generate ABAC AC models. Moreover, although our proposed AC metamodel in [12] is promising, many other phases are still missing and need to be handled and implemented, for example, developing DSL, a detailed case study, etc. Table 3 summarizes the objective(s) and limitation(s) for each of the proposed metamodels. Despite the proposed AC metamodels have gained the attention of researchers for a decade, they have common limitations. These limitations cannot be ignored, especially, with recent computing environments which are open to all kinds of attacks and threats.

4.2. The Common Limitations

Even with the advancements of implementing AC metamodels in various scenarios, each particularly has its limitation(s) in addition to some common limitations. They all lack some essential characteristics and can be enumerated as follows:

- Each metamodel is itself a case and does not encompass a general base concept to derive various instances for all AC models. In other words, they are planned for dedicated scenarios or case studies based on some features of AC models;
- They do not support the ability to define various types of attributes. So, they are not dynamic enough to follow the continuous technology upgrades.
- Neither the generic nor extended proposed metamodels is enough to address the needed target of enforcing AC policy, especially with the current technologies and continuous upgrades;
- No provided explanations about how the derived models could collaborate within the same computing architecture e.g., IoT;
- An essential aspect is not considered in all of the presented AC metamodels which is the migration of AC policy from one AC model to another. Having a metamodel should make it possible to translate an existing AC policy between the different AC models covered by the metamodel.

Fig. 7 summarizes the common limitations that should be addressed in the proposed AC metamodels. Accordingly, we are

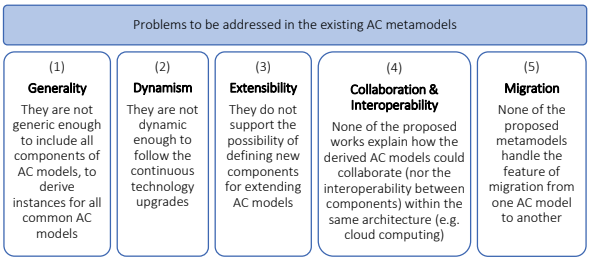

Fig. 7. The common limitations in the existing AC metamodels

constructing a unified and generic AC metamodel [12] that considers the continuous technology progressions, the variety of information systems, and the heterogeneity of AC models.

5. Research issues and open questions

The definition of security policies with the current computing environments, especially IoT, involves complexities and difficulties due to the following facts:

Table 3. Objective(s) and Limitation(s) of The Proposed Access Control Metamodels

Author(s)	Objective(s)	Limitation(s)
Barker [26]	Multiple models can be derived as special cases from CBAC metamodel.	- lacks the support of resource and action hierarchies.
Slimani et al. [27]	To provide support for hybrid AC policies by allowing categories to be associated with other categories and finding hierarchical relationships between them.	- hybrid structure to derive some AC models rather than a metamodel.
Alves et al. [28]	To allow security administrators to check the consistency of a policy combining authorizations and obligations.	- no explanation of how the approach could be dynamic in distributed contexts which are rich of events and variable attributes.
Bertolissi et al. [30]	To provide semantics for distributed AC mechanisms within distributed environments consisting of several sites.	- no real case studies are explained or implemented.
Khamadja et al.[31]	To develop a new cloud computing service named "Access Control as a Service".	- no case study or testing result, also they do not explain how access can be controlled in the context of multi-cloud.
Xia et al. [32]	To handle security and privacy in cloud service development and operations.	- have not explained how access can be controlled in the context of multi-cloud.
Martinez et al. [33]	To ease the analysis and manipulation of security requirements in WCMSs.	- the notion of variable attributes is not considered, also no explanations of how Drupal framework can be upgraded.
Martinez et al. [34]	To extract network AC policies enforced by firewalls within a network system, then AC policy would be easier to understand, analyze and manipulate.	- no explanations of how the extended networks firewall systems can be upgraded.
Abd-Ali et al. [29]	To concurrently handle multiple AC models (CW, BLP, BIBA, and RBAC)	- hybrid structure to derive some AC models rather than a metamodel.
Trninić et al. [35]	to allow a security expert to express AC policies for a given AC model.	- does not consider dynamic constraints.
Korman et al. [36]	To provide support for architectures of enterprises by extending their abilities to model authorization and AC in their frameworks.	- difficulty for potential implementation of automated analytical capabilities, also no explanations of how the extended ArchiMate framework can be upgraded.
Gorshkova et al. [37]	To provide a metamodel extension for Spring Security framework to meet modern security requirements.	- they extend Spring Security framework to support some AC features without explaining how these extensions can be upgraded.
Adda et al. [39]	To provide a generic ABAC metamodel to generate a wide variety of AC models related to ABAC.	- limited to ABAC models.
Kashmar et al. [12]	To provide a generic AC metamodel with a unified set of AC concepts	- no case study or testing result, also no explaination of how access can be controlled in distributed environment.

- the heterogeneity of security strategies for information systems such as centralized, decentralized, or both
- the diversity of AC rights which might be raised from different information systems such as allow, deny, mixed, or undetermined, for different units such as subjects, roles, categories, groups, etc.
- the heterogeneity of security policies for different AC models and their extensions.
- the heterogeneity of security elements of various AC models such as objects, subjects, types, relations, etc.
- the heterogeneity of networks, platforms, applications, devices, etc. with multimillions of users

These facts and the complex structure of the recent technologies (cloud computing, IoT, ...) reflect the importance of developing an enhanced AC metamodel approach to adapt the continuous technology progressions and the existing heterogeneities in different domains. Through this review, we can find that there is a limited number of recent research proposals for AC metamodels. Yet, various research is still conducting for the AC metamodeling approach to find a more general metamodel that can be used to dynamically define AC policies.

However, finding a new generic metamodel that includes all AC models features, dynamic, and upgradable is a challenging topic. What makes it a critical need are the following:

- the heterogeneity and complexity in the structures of recent technologies and their environments;
- the continuous upgrades of the new technologies, especially IoT;
- the dynamic requirement for enforcing security issues;
- the need to find the collaboration between various AC models within the same architecture;
- the importance of migrating AC policies from one model to another.

Despite the proposed AC metamodels have some enhanced features, they lack some important characteristics that are essential to the current fact of technologies. Through this review we can find that some issues need to be addressed which are:

5.1. Generality

Generality is the first essential feature that must be considered in developing an AC metamodel. A generic AC metamodel should have the following characteristics:

- includes most of the features of the common AC models;
- can be oriented to derive various AC models and methods, and for specifying any AC policy in centralized and distributed computing environments;
- works as a base to construct other essential characteristics (e.g., a collaboration between AC models).

Note that, we address this issue in [10, 11, 38] then propose our metamodel approach in [12].

5.2. Dynamism

The term dynamism refers to the change within a system, model, etc., and being upgradable due to changing conditions or rules. A generic and dynamic AC metamodel should:

- describe how metamodel properties can be changed or modified over time along with technology progressions, for example, due to the changing environmental conditions.

- allow defining new types of attributes/entities, to describe a larger set of rules to express policies. Hence, various models can be formulated for static and dynamic policy enforcement.
- allow building relationships between its elements and describes the structural changes to reflect its dynamic characteristics.

5.3. Extensibility

Extensibility is the feature of being designed to allow adding new components, for an already defined model, with the relationships between them. Some of the proposed AC methods are based on (or extended from) the common AC models, while others are formulated based on the needed context. This reflects the diversity of the implemented AC models in different fields and the importance of upgrading them to follow technology progressions. The key components for the different AC methods are subjects, objects, actions, security levels, attributes, etc. The existing AC metamodels do not include the possibility of defining new components rather than the defined ones in the core structure. Hence, developing generic and dynamic metamodel is important to extend the existing AC methods, and to formulate and implement new ones.

5.4. Collaboration and interoperability

Collaboration is underlined as a goal for distributed computing environments, in collaborative computing environments, various collections of information systems and technologies are performed to support cooperation between organizations, individuals, etc. In these environments, organizations collaborate from remote locations, and users are allowed/denied to share information, upload content, communicate via applications such as video conferencing. To establish interoperability, various concepts must be studied such as autonomy, dynamism, and heterogeneity of systems, models, etc.; hence computational entities can collaborate to fulfill their mutual goals [40]. Collaborative environments need to control access to their assets to increase working cooperation efficiently and effectively. Finding a general basis for AC metamodel would allow handling multiple models to find advanced security features and operations, which would in turn, permit the collaboration between the obtained models and the interoperability between components of AC models.

5.5. Migration

Another interesting feature, that is missing in current AC metamodels, is the ease of migration from one model to another. In fact, having a metamodel should make it possible to translate an existing AC policy between different AC models covered by the metamodel. However, a metamodel with a generic, dynamic, and extendable structure can be implemented to allow migrating the AC policies from one model to another.
However, in this context we can raise the following questions:

- how a new generic and dynamic AC metamodel that considers the continuous technology progressions can be designed?
- what are the main features, componenets, etc. this AC metamodel can include?
- how its structure can be developed to handle collaboration/interoperability/extension/migration of AC models?
- how to construct a common set of AC concepts for the heterogeneous AC models?
- how heterogeneous AC models can interact to ensure privacy?

6. Conclusion and Future Perspectives

In this paper, we review and analyze the proposed AC metamodels, explain their objectives, their limitations especially with current technology progressions and upgrades. In this review, we provide a critical analysis, in addition to the potential research issues in this domain. The common limitations, which can also be considered as research issues in this domain, that have not been addressed yet are important to be implemented with the current heterogeneous computing environments.

Acknowledgment

We acknowledge the support of the Natural Sciences and Engineering Research Council of Canada (NSERC) [funding reference number 06351], Fonds Québécois de la Recherche sur la Nature et les Technologies (FRQNT), and Centre d'Entrepreneuriat et de Valorisation des Innovations (CEVI).

References

[1] EG Petrakis and Xenofon Koundourakis. ixen: Secure service oriented architecture and context information management in the cloud. *Journalof Ubiquitous Systems and Pervasive Networks (JUSPN)*, 14(2):01–10, 2021.

[2] Kamalendu Pal and Ansar-Ul-Haque Yasar. Convergence of internet of things and blockchain technology in managing supply chain. *Journalof Ubiquitous Systems and Pervasive Networks (JUSPN)*, 14(2):11–19, 2021.

[3] Jun Ho Huh, Rakesh B Bobba, Tom Markham, David M Nicol, Julie Hull, Alex Chernoguzov, Himanshu Khurana, Kevin Staggs, and Jingwei Huang. Next-generation access control for distributed control systems. *IEEE Internet Computing*, 20(5):28–37, 2016.

[4] Sowmya Ravidas, Alexios Lekidis, Federica Paci, and Nicola Zannone. Access control in internet-of-things: A survey. *Journal of Network and Computer Applications*, 144:79–101, 2019.

[5] Mehdi Sookhak, F Richard Yu, Muhammad Khurram Khan, Yang Xiang, and Rajkumar Buyya. Attribute-based data access control in mobile cloud computing: Taxonomy and open issues. *Future Generation Computer Systems*, 72:273–287, 2017.

[6] Nadine Kashmar, Mehdi Adda, Mirna Atieh, and Hussein Ibrahim. *Access Control in Cybersecurity and Social Media*, chapter 4. Université Laval, 2021.

[7] V.C. Hu, D.F. Ferraiolo, R. Chandramouli, and D.R. Kuhn. *Attribute-Based Access Control*. Artech House Publishers, 2017. ISBN 9781630814960.

[8] Nadine Kashmar, Mehdi Adda, and Mirna Atieh. From access control models to access control metamodels: A survey. In *Future of Information and Communication Conference*, pages 892–911. Springer, 2019.

[9] Nadine Kashmar, Mehdi Adda, Mirna Atieh, and Hussein Ibrahim. A review of access control metamodels. *Procedia Computer Science*, 184:445–452, 2021.

[10] Nadine Kashmar, Mehdi Adda, Mirna Atieh, and Hussein Ibrahim. A new dynamic smart-ac model methodology to enforce access control policy in iot layers. In *2019 IEEE/ACM 1st International Workshop on Software Engineering Research & Practices for the*

Internet of Things (SERP4IoT), pages 21–24. IEEE, 2019.

[11] Nadine Kashmar, Mehdi Adda, Mirna Atieh, and Hussein Ibrahim. Smart-ac: A new framework concept for modeling access control policy. *Procedia Computer Science*, 155:417–424, 2019.

[12] Nadine Kashmar, Mehdi Adda, Mirna Atieh, and Hussein Ibrahim. Access control metamodel for policy specification and enforcement: From conception to formalization. *Procedia Computer Science*, 184:887–892, 2021.

[13] Mohammed Ennahbaoui and Said Elhajji. Study of access control models. In *Proceedings of the World Congress on Engineering*, volume 2, pages 3–5, 2013.

[14] RS Sandhu, EJ Coyne, HL Feinstein, and CE Youman Role-Based. Access control models. *IEEE computer*, 29(2):38–47, 2013.

[15] Edwin Okoampa Boadu and Gabriel Kofi Armah. Role-based access control (rbac) based in hospital management. *Int. J. Softw. Eng. Knowl. Eng*, 3:53–67, 2014.

[16] Abdeljebar Ameziane El Hassani, Anas Abou El Kalam, Adel Bouhoula, Ryma Abassi, and Abdellah Ait Ouahman. Integrity-orbac: a new model to preserve critical infrastructures integrity. *International Journal of Information Security*, 14(4):367–385, 2015.

[17] Khalifa Toumi, César Andrés, and Ana Cavalli. Trust-orbac: A trust access control model in multi-organization environments. In *International Conference on Information Systems Security*, pages 89–103. Springer, 2012.

[18] Linda Aliane and Mehdi Adda. Hobac: toward a higher-order attribute-based access control model. *Procedia Computer Science*, 155:303–310, 2019.

[19] D Richard Kuhn, Edward J Coyne, and Timothy R Weil. Adding attributes to role-based access control. *Computer*, 43(6):79–81, 2010.

[20] Qasim Mahmood Rajpoot, Christian Damsgaard Jensen, and Ram Krishnan. Integrating attributes into role-based access control. In *IFIP Annual Conference on Data and Applications Security and Privacy*, pages 242–249. Springer, 2015.

[21] Hui Qi, Xiaoqiang Di, and Jinqing Li. Formal definition and analysis of access control model based on role and attribute. *Journal of information security and applications*, 43:53–60, 2018.

[22] Fatemeh Nazerian, Homayun Motameni, and Hossein Nematzadeh. Emergency role-based access control (e-rbac) and analysis of model specifications with alloy. *Journal of information security and applications*, 45:131–142, 2019.

[23] Harsha S Gardiyawasam Pussewalage and Vladimir A Oleshchuk. Attribute based access control scheme with controlled access delegation for collaborative e-health environments. *Journal of information security and applications*, 37:50–64, 2017.

[24] Saïd Assar. Meta-modeling: concepts, tools and applications. In *IEEE RCIS'15: 9th International Conference on Research Challenges in Information Science*, 2015.

[25] David Ferraiolo and Vijay Atluri. A meta model for access control: why is it needed and is it even possible to achieve? In *Proceedings of the 13th ACM symposium on Access control models and technologies*, pages 153–154, 2008.

[26] Steve Barker. The next 700 access control models or a unifying meta-model? In *Proceedings of the 14th ACM symposium on Access control models and technologies*, pages 187–196, 2009.

[27] Nadera Slimani, Hemanth Khambhammettu, Kamel Adi, and Luigi Logrippo. Uacml: Unified access control modeling language. In *2011 4th IFIP International Conference on New Technologies, Mobility and Security*, pages 1–8. IEEE, 2011.

[28] Sandra Alves, Anatoli Degtyarev, and Maribel Fernández. Access control and obligations in the category-based metamodel: a rewrite-based semantics. In *International Symposium on Logic-Based Program Synthesis and Transformation*, pages 148–163. Springer, 2014.

[29] Jamal Abd-Ali, Karim El Guemhioui, and Luigi Logrippo. A metamodel for hybrid access control policies. *JSW*, 10(7):784–797, 2015.

[30] Clara Bertolissi and Maribel Fernández. A metamodel of access control for distributed environments: Applications and properties. *Information and Computation*, 238:187–207, 2014.

[31] Salim Khamadja, Kamel Adi, and Luigi Logrippo. Designing flexible access control models for the cloud. In *Proceedings of the 6th International Conference on Security of Information and Networks*, pages 225–232, 2013.

[32] Tian Xia, Hironori Washizaki, Takehisa Kato, Haruhiko Kaiya, Shinpei Ogata, Eduardo B Fernandez, Hideyuki Kanuka, Masayuki Yoshino, Dan Yamamoto, Takao Okubo, et al. Cloud security and privacy metamodel. In *Proceedings of the 6th International Conference on Model-Driven Engineering and Software Development*, pages 379–386. SCITEPRESS-Science and Technology Publications, Lda, 2018.

[33] Salvador Martínez, Joaquin Garcia-Alfaro, Frédéric Cuppens, Nora Cuppens-Boulahia, and Jordi Cabot. Towards an access-control metamodel for web content management systems. In *International Conference on Web Engineering*, pages 148–155. Springer, 2013.

[34] Salvador Martínez, Jordi Cabot, Joaquin Garcia-Alfaro, Frédéric Cuppens, and Nora Cuppens-Boulahia. A model-driven approach for the extraction of network access-control policies. In *Proceedings of the Workshop on Model-Driven Security*, pages 1–6, 2012.

[35] Branislav Trninić, Goran Sladić, Gordana Milosavljević, Branko Milosavljević, and Zora Konjović. Policydsl: Towards generic access control management based on a policy metamodel. In *2013 IEEE 12th International Conference on Intelligent Software Methodologies, Tools and Techniques (SoMeT)*. IEEE, 2013.

[36] Matus Korman, Robert Lagerström, and Mathias Ekstedt. Modeling enterprise authorization: a unified metamodel and initial validation. *Complex Systems Informatics and Modeling Quarterly*, (7):1–24, 2016.

[37] Ekaterina Gorshkova, Boris Novikov, and Manoj Kumar Shukla. A fine-grained access control model and implementation. In *Proceedings of the 18th International Conference on Computer Systems and Technologies*, pages 187–194, 2017.

[38] Nadine Kashmar, Mehdi Adda, Mirna Atieh, and Hussein Ibrahim. Deriving access control models based on generic and dynamic metamodel architecture: Industrial use case. *Procedia Computer Science*, 177:162–169, 2020.

[39] Mehdi Adda and Linda Aliane. Hobac: fundamentals, principles, and policies. *Journal of Ambient Intelligence and Humanized Computing*, 11(12):5927–5941, 2020. . URL https://doi.org/10.1007/s12652-020-02102-y.

[40] Toni Ruokolainen. Modelling framework for interoperability management in collaborative computing environments. *Licentiate thesis, University of Helsinki, Department of Computer Science*, 2009.