SoNeUCON$_{ABC}$, an expressive usage control model for Web-Based Social Networks

Lorena González-Manzano*, Ana I. González-Tablas, José M. de Fuentes, Arturo Ribagorda

Avda. de la Universidad, 30. Computer Science and Engineering Department. University Carlos III of Madrid, 28911 Leganés, Spain

Abstract

In the era of hyper-connectivity Web-Based Social Networks (WBSNs) are demanding applications. They facilitate the interaction of huge amounts of users and the development of appropriate Access Control Models (ACMs) is an arising necessity. Particularly, the development of WBSNs ACMs with expressive power and capable of managing access control along the whole usage process is the challenge pursued. To contribute on this issue, first, 23 proposals have been analysed and second, $SoNeUCON_{ABC}$, an expressive usage control model for WBSNs, is proposed. It extends $UCON_{ABC}$ [1] including relationships management and it is formally defined, specifying entities and elements involved and an access control policy language. Moreover, policy construction is carefully detailed by using regular expressions and access control enforcement functions are described. Finally, the evaluation shows, theoretically, the significant expressive power of $SoNeUCON_{ABC}$ and, empirically, the feasibility of its implementation by the development of a proof of concept system.

Keywords: Web Based Social Networks, Access control, Access control model, Expressive power, Access control policy

1. Introduction

Web-Based Social Networks (WBSNs) facilitate the interaction of large number of users and the sharing of huge amounts of data worldwide, from which arises the necessity of managing access control. Users have to establish the appropriate mechanisms to limit who has access to their data according to their privacy preferences. Multiple studies remark that users do not want to share all their resources and personal information to the same extent [2, 3]. For instance,

*Corresponding author

Email addresses: lgmanzan@inf.uc3m.es (Lorena González-Manzano), aigonzal@inf.uc3m.es (Ana I. González-Tablas), jfuentes@inf.uc3m.es (José M. de Fuentes), arturo@inf.uc3m.es (Arturo Ribagorda)

in a Facebook research, data such as personal address, cellphone or videos are considered extremely sensitive [2]. On the contrary, data such as school name, interests or photos are largely disclosed. Indeed, mimicking daily life interactions is the pursued challenge [4].

A great amount of access control models (ACMs) have been proposed. They are key elements because every access to a system has to be controlled and only authorized ones can take place [5]. Traditional ACMs can be classified as Mandatory Access Control Model (MAC) where objects and subjects are classified according to security levels and access is granted in regard to them; Discretionary Access Control (DAC), in which access to information is carried out in respect to the user's identity and a set of authorizations or rules; and Role-Based Access Control (RBAC) which bases on the definition of different roles, the assignment of permissions to roles and the assignment of roles to subjects.

Although traditional ACMs can be used in all systems, WBSNs require managing access to resources by a set of users who are related between each other. Thus, this management complexity has to be addressed by WBSN ACMs. In particular, assorted ACMs have been proposed for the WBSN field. Some of them apply techniques that involve distance or trust between users and follow traditional mechanisms such as Access Control Lists [6, 7]. Other contributions focus on the use of cryptography, managing access control by encrypting data and delivering decryption keys to chosen contacts [8, 9]. Some recent proposals point out the viability of involving attributes in access control management, mainly user and data attributes [9, 8, 10, 11]. Nonetheless, there is an issue which has not been deeply addressed yet. It refers to the necessity of providing ACMs with expressive power, which means "*the ability of the models to support a variety of access control policies. In other words, the expressivity of an access control model is a measure of the range of policies it can support*" [12]. Expressive power has been studied by several researchers in the context of access control [12, 13, 14] but not for WBSNs where this paper contributes.

The contribution of this paper is twofold. Firstly, inspired by [12], the expressive power of 23 approaches has been compared. In particular, their *semantic equivalence*, that is the semantic similarity between access control policies, has been analysed. The comparison focuses on models to manage access control in WBSNs as well as mechanisms that, without proposing a specific model but being based on a particular one, contain a policy language. Results of the analysis show the lack of expressive power in WBSN ACMs, being [15], [16] and [10] the more expressive ones. In second place, trying to address the limitations of analysed proposals and given the need of managing access control along the whole usage process (the concept "usage control" comes into play), an expressive usage control model for WBSNs, called $SoNeUCON_{ABC}$, is developed. It extends $UCON_{ABC}$, an usage control model appropriate for dynamic environments [17], including relationship management. $SoNeUCON_{ABC}$ considers the set of relationships between the administrator of the requested object and the requester, that is, the set of relationships by which the requester can be reached, directly or indirectly, from the administrator. Specifically, multiple paths, cliques, distance and common-contacts are the features managed concerning relationships. Moreover, access control management is carried out in a privacy-preserving way, as apart from the attributes of the administrator and the requester of particular data, the attributes of the rest of nodes involved in the relationship between them remain hidden. $SoNeUCON_{ABC}$ is specified by presenting a formalization of the model, a policy language and a set of enforcement functions. The proposal is evaluated theoretically and empirically. The theoretical evaluation shows that $SoNeUCON_{ABC}$ satisfies all established requirements. Additionally, the empirical evaluation shows that the model can be successfully implemented and applied in the majority of proposed cases. Considering that the tolerable waiting time of WBSN users for information retrieval is approximately 2,000 ms [18], the enforcement of policies without cliques is satisfactory as long as less than about 200,000 nodes and 200 relationships per node are explored. By contrast, in case of policies with cliques, their enforcement does not have to exceed about 30,000 explored nodes and 200 relationships per node.

The article is structured as follows. Section 2 describes related work. Section 3 lays the basis to compare the expressive power of WBSNs ACMs and presents a comparison of 23 approaches. Subsequently, Section 4 defines $SoNeUCON_{ABC}$, an expressive usage control model. Next, the proposed model is evaluated, theoretically in Section 5 and empirically in Section 6. Finally, Section 7 presents conclusions and open research issues.

2. Related work

This Section presents the classification of 23 WBSN ACMs (Section 2.1), an analysis of the application of expressive power (Section 2.2) and the procedures to compare expressive power (Section 2.3).

2.1. ACMs for WBSNs

Since the creation of traditional ACMs until the emergence of WBSNs, ACMs developed for the social networking field can be classified under the following groups [19]:

- *Role-Based Access Control (RBAC)* [20]:roles are assigned to permissions and then, such roles are linked to WBSN users.
- *Trust-Based Access Control (TBAC) [21]*: it focuses on rule-based models that include trust as a condition to access to data. Depending on the level of trust of users and data, access is granted or denied.
- *Relationship-Based Access Control (RelBAC)* [16]: it bases on managing access control through relationships created between pairs of users. Thus, authorizations are established as relationships between users and data.
- *Attribute-Based Access Control (ABAC)* [22]: it is a rule-based approach that consists of managing access control by the application of objects, resources and environment attributes. Then, authorization decisions base on attribute values within rules established by data administrators.

- *Ontology-Based Access Control (OBAC)* [15]: it focuses on creating assorted and fine-grained access control policies by the use of ontologies specially developed for social networks.

According to the aforementioned groups of ACMs, a total of 23 proposals regarding access control management in WBSNs are classified in Table 1. It is noticed that 3 approaches fall in RBAC, 7 in RelBAC, 5 in ABAC, 5 in TBAC and 3 in OBAC. Thus, as the majority of approaches fall in the RelBAC model, relationships management can be pointed out as a remarkable research issue. For the sake of clarity, a brief description of each proposal can be found in Appendix B.

Table 1: ACMs for WBSNs

	General access control models				
	RBAC	RelBAC	ABAC	TBAC	OBAC
J. Park *et al.* [10]			√		
A. Masoumzadeh *et al.* [15]					√
B. Carminati *et al.* [21]				√	
J. Li *et al.* [20]	√				
T. Abdessalem *et al.* [23]				√	
P. Fong *et al.* [16]		√			
P. Fong *et al.* [24]		√			
A. Ahmad *et al.* [25]	√				
Y. Cheng*et al.* [26]		√			
CVD. Munckhof [22]			√		
A. Tapiador *et al.* [27]	√				
B. Ali *et al.* [28]				√	
B. Carminati *et al.* [7]					√
H. Wang *et al.* [29]				√	
H. Hu *et al.* [30]		√			
W. Villegas *et al.* [31]				√	
I.B. Dhia [32]		√			
I.B. Dhia [33]		√			
S. Jahid *et al.* [9]			√		
R. Baden *et al.* [8]			√		
S. Braghin *et al.* [34]			√		
M. Alizadeh *et al.* [35]					√
G. Bruns *et al.* [36]		√			

2.2. The application of expressive power

Being expressive is a desirable feature in many areas within the computation context. N. Knoch *et al.* analyse the expressive power of UML, the Universal Modelling Language, in the context of web applications [37]. Also related to programming languages, E. Dantsin *et al.* surveys multiple ways of logic programming [38] and B. Bérard *et al.* propose an expressive language for timed automatas [39].

Another area of interest is access control. Many authors highlight the relevance of developing expressive ACMs [40, 41] but, as pointed out by S. Capitani *et al.* in [41], there have been few authors whose research focus has gone in this direction. Indeed, to the best of authors knowledge, the work of J.B. Joshi *et al.* is the only identified proposal specially focused on the study of the expressive power of an ACM, called Generalized Temporal Role-Based Access Control [42].

2.3. Expressive power of ACMs: comparison procedures

The study of techniques to compare expressive power of ACMs has received some attention, though not particularly focused on ACMs for WBSNs. For instance, Ganta *et al.* present a formalization to compare expressive power focused on Typed Access Matrix Model (TAM), Augmented TAM and their variations [13]. Similarly, Bertino *et al.* present a framework for reasoning about ACMs [12]. It focuses on mapping rules (composed of objects, subjects, privileges and authorizations) of a couple of models to a common language based on mathematical logic, and comparing results to determine the model that is at least as expressive as the other one. Afterwards, Tripunitara *et al.* propose a theory for comparing models based not only on the state of the model, expressed by a particular access control policy, but on the state-transition [14]. Their proposal expresses an ACM as a set of states, policies and state-transitions rules to define how to pass through from one state to another. For the same purpose but from a different perspective, several works base on applying simulation to perform the comparison [43, 44].

On the other hand, some contributions analyse several WBSN access control features, though not being particularly related to expressive power. Carminati and Ferrari present a survey of access control in WBSNs to study features such as the kind of relationships existing in current WBSNs and the management procedures applied [45]. From a more specific point of view, A. Lazouski *et al.* survey literature related to $UCON_{ABC}$ *usage model*, studying among other aspects managed elements and subjects' behaviour [17]. Moreover, Cheng *et al.* point out the necessity of achieving expressive fine-grained policies and study characteristics of different ACMs for WBSNs like the management of multiple relationships types [26].

3. Comparing the expressive power of ACMs for WBSNs

The comparison of the expressive power of ACMs for WBSNs involves the description of the motivation and the applied methodology (Section 3.1), the identification and definition of a set of WBSN features and control policies (Section 3.2) and the summary and discussion of the analysis of 23 studied proposals (Section 6.4).

3.1. Motivation and methodology

Considering that policies are the main managed elements, Bertino *et al.* proposal inspires the comparison performed herein. They propose a framework to represent ACMs homogeneously to be later compared [12]. In particular, models are represented under a common logic language and their expressive power is compared by analysing their *structural* and *access equivalence*. The former refers to verifying if models are built from the same set of structural components and the latter to checking whether models instances enforce the same set of accesses. Nonetheless, Bertino *et al.* framework is not appropriate for WBSNs due to a pair of reasons. Firstly, their framework is specially focused on

DAC, MAC and RBAC, thereby limited for the large quantity of WBSN access control management elements. It manages access control policies (called authorizations) composed of objects, subjects and privileges, which are a reduced set of elements for the WBSN field. For instance, relationships are essential WBSN elements, as well as subjects, objects or relationships attributes, and they are hard to manage within Bertino *et al.* framework. Secondly, instead of studying *structural* and *access equivalence*, the presented approach analyses the *semantic equivalence* of WBSN ACMs. Given a set of models to compare, *semantic equivalence* refers to the semantic similarity of a set of policies that are defined by using each model policy language. For instance, the policy "*grant access to friends of a friend*" can be expressed in multiple policy languages but the semantic meaning must be analogous regardless of the applied language. Therefore, thanks to the proposed semantic equivalence technique, it is possible to assess the expressive power of ACMs without the need of translating them into a common, unified representation.

The comparison methodology proposed herein consists of several tasks. Initially, a total of six features that WBSN ACMs should include and access control policies should cover, are identified. Besides, seven policies are defined to cover all pointed out features. Applying inductive reasoning it is asserted that the model that expresses a policy linked to a particular feature can express any policy associated with this feature (see Appendix A). Subsequently, 23 approaches are analysed to establish whether it is possible to define the proposed access control policies using each provided policy language (see Appendix B). Finally, a summary and a discussion is presented according to the *semantic equivalence* analysis carried out in each studied proposal. It should be noticed that the mentioned set of approaches focus on WBSN ACMs as well as mechanisms that, without proposing a specific model but being based on a particular one, contain a policy language.

3.2. WBSN features and access control policies

WBSNs are novel systems described as graphs in which nodes correspond to users and relationships to edges [21]. Then, access control management consists of establishing which users are able to perform actions over certain data according to contextual features and relationships between them. Therefore, in WBSNs users, data, actions, relationships and context are the elements at stake.

A relationship can be defined as a connection between an administrator and a requester. The former is the user who owns particular data and establishes access control policies and the latter refers to the user who requests an action over particular data. Then, a directed arrow from the administrator to the requester expresses that the administrator has a certain type of relationship with the requester but not the other way round. For instance, in Figure 1, given User1 and User2, the arrow from User1 to User2 means that User1 has a particular relationship with User2, being User1 the administrator and User2 the requester.

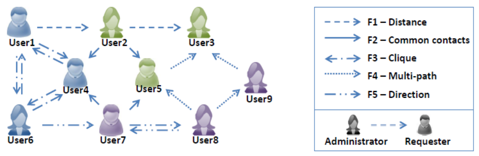

Figure 1: WBSNs features

Taking into account these aspects and according to literature, a set of six interesting features, which are pointed out as the bases of the expressive power of WBSNs, are identified and depicted in Figures 1 and 2:

F1 *Distance* [46, 21]: WBSNs are composed of a vast quantity of users who interact between each other. However, two users of a WBSN may not be directly connected but indirectly, that is, a direct relationship does not exist between them but a path connecting both users can be found considering other users and their relationships. For instance, depicted in Figure 1, User1 and User3 are indirectly connected through User2.

F2 *Common-contacts* [46, 47, 48]: WBSNs users have, in multiple circumstances, common contacts. In this proposal such contacts are defined as the establishment of a unidirectional relationship (F5) from the administrator to his contacts and a unidirectional relationship (F5) between the requester and his contacts. For instance, in Figure 1 User2 is unidirectionally connected with User4 and User5, and User7 has a unidirectional relationship with both users too. Then, User2 and User7 have User4 and User5 in common.

F3 *Clique* [46, 47]: a set of WBSNs users form a clique, that is they belong to a close-knit group in which all contacts are bidirectionally connected between each other. For instance, User1, User4 and User6 depicted in Figure 1 form a clique.

F4 *Multi-path* [49, 45]: WBSNs consist of users related that establish connections with other users of the WBSN. When two users are not directly connected but indirectly, it is said that a path exists between both. In particular, several paths may exist between two users, that is, involving each path a different set of ordered nodes. For instance, in Figure 1, User8 and User3 are connected by a pair of paths which are different because they are composed of a distinct sets of nodes.

F5 *Direction* [21, 47, 26]: a relationship can be established in a unidirectional or bidirectional way. The former corresponds to the case in which a relationship request is only established in one direction. For example, in Figure 1, User6 says to have a relationship with User7 and thus, they are connected by a directional relationship where User6 is the administrator and User7 the requester. On the contrary, the latter, a bidirectional relationship, implies that both users associated to a relationship are administrators and requesters simultaneously (i.e., User7 and User8 of Figure 1). Moreover, notice that a bidirectional relationship consists of a pair of unidirectional ones.

F6 *Flexible attributes* [26, 50]: assuming the diversity and quantity of WBSNs users, demanding necessities are unpredictable. Consequently, flexible and fine-grained ACMs become essential to allow the definition of concrete preferences in regard to relationships, objects and users. For instance, in Figure 2, User3 grants access to users (e.g. User7) who are over 20 and under 30 (20$<$ age $<$ 30) and who are distanced a pair of hops (path with length two), where the relationship of the first hop is highly (H) trusted and has the role colleague and the relationship of the second hop is highly trusted and started in 2011. Also regarding fine-grained management, in Figure 2, User2 establishes that given a particular requester (such as User6), all users involved in every path that connects the requester and User2, have to highly trust the requester.

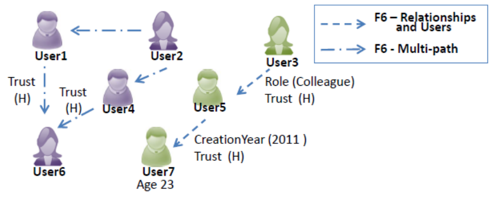

Figure 2: F6: Flexible attributes

In order to study the aforementioned set of WBSN features, it is assumed that, in a WBSN, an administrator (the owner of the data) establishes the following set of access control policies (P) to grant access to particular data, an object, to a requester. Additionally, for the sake of simplicity, a single right is managed, i.e. the right to access to an object. Other rights, such as write or delete, are left aside because an analogous management procedure is assumed and it does not affect the definition of policies.

Policies established by the administrator are the following ones (note that at the end of each policy, covered features are stated within brackets):

P1 Access is granted to users who are friends of neighbours of his/ her relatives if the relationship between his/ her relatives and his/ her relatives' neighbours was established before 2,000. (F1 and F6)

P2 Access is granted to users who have three friends in common with the administrator of the requested object. (F2)

P3 Access is granted to users who belong to a clique in which two users and the administrator of the requested object are involved, having all of them a friendship relationship. (F3)

P4 Access is granted to users who are connected to the administrator by two different paths composed of unidirectional relationships oriented from the requester to the administrator. Moreover, relationships involved in all paths have to be highly trusted. (F4 and F6)

P5 Access is granted to users who are friends of the administrator of the requested object, also having a bidirectional relationship with him/ her. (F5)

P6 Access is granted to users who are friends of the administrator of the requested object. (F5)

P7 Access is granted to users if they are females under 30 years old or if they are females under 40 who have studied computer science or if they are females who have studied computer science and physics. (F6)

3.3. Expressive power analysis: summary and discussion

Once concluded the study of the expressive power of ACMs for WBSNs by studying their *semantic equivalence*, detailed in Appendix B, this Section presents a deep analysis. The most appropriate model corresponds to the one that allows the specification (based on the proposed model-dependant definitions) of as many policies as possible, facilitating the expression of a wide set of user preferences. Table 2 presents a summary of the analysis. Each policy is marked as *completely* ($\surd$) or *partially* ($\mathcal{P}$) defined according to the achieved expressive power. Moreover, note that policies that involve more than a single feature, P1 and P4 in particular, require $\surd$ or $\mathcal{P}$ per each involved feature. No marks are used for undefined features.

The most surprising issue is that, even being models specially focused on WBSNs, none of them allows the expression of all identified features (Section 3.2), thus limiting the possibilities of users to manage their personal preferences. Furthermore, concerning the general classification (RBAC, RelBAC, ABAC, TBAC and OBAC, recall Section 2.1), models based on roles seem to be the least expressive while those that manage relationships and attributes are the most successful.

According to indirect relationships (F1 and P1), the great majority of models deal with them. More specifically, 7 out of 23 achieve the successful definition of indirect relationships with an unlimited distance between users [15, 21, 23, 16, 26, 7, 36]. By contrast, [24] and [30] try to define P1 but they only achieve the definition of two hops relationships.

Concerning common contacts (F2 and P2), a total of 5 models out of 23 manage this feature [51, 15, 16, 24, 36]. Nonetheless, even attaining a successful definition of P2, it would be desirable to work in describing access control enforcement procedures similarly to [36].

A challenging issue is the specification of a clique (F3 and P3). Surprisingly, this feature is managed by the same models that deal with F2 and they refer to those proposed by Fong *et al.* [16, 24]. Specifically, F3 management involves a great deal of complexity. As highlighted in [46], users involved in a clique have to be discovered from unreachable users and the difficulty of their management is not even mentioned in Fong *et al.*'s proposals. Besides, no guidelines regarding access control enforcement are provided.

On the other hand, multi-path (F4 and P4) is another appealing feature that 4 models out of 23 manage [15, 16, 24, 36]. Nevertheless, three of them ([15, 24, 36]) do not fully deal with F4. They cannot define policies which include multiple and different paths. Indeed, the satisfaction of F4 is a challenging matter in the WBSN context [49, 45], although it can be simplified to groups management. For instance, the existence of the relationship "relative" and the relationship "friend" can be compared with the creation and management of a group of friends and a group of relatives.

Relationships direction is another studied feature (F5, P5 and P6). Concerning bidirectional relationships (F5 and P5), they are interestingly managed by 12 out of 23 proposals. Moreover, assorted techniques are applied to deal with this type of relationships: specific attributes are created [51, 52], pairs of directional relationships are established [15, 23, 36] or, as in the majority of current WBSNs, relationships are considered inherently bidirectional [20, 24, 30]. However, it is remarkable that [23] unsuccessfully defines P5 since the created policy is satisfied by unidirectional and bidirectional relationships. On the other hand, in respect to unidirectional relationships (F5 and P6), 20 out of 23 approaches manage them. Those that base on cryptography require exchanging decryption keys and thus, unidirectional relationships are implicitly established [32, 33, 34]. By contrast, other work like the one proposed by M. Alizadeh *et al.* highlight the management of directed labelled relationships [35].

In regard to flexible attributes (F6, P1, P4 and P7), the model proposed by J. Park *et al.* [10] is the most expressive one, attaining the proper specification of P1 and P7. According to P1, fine-grained management is considered in [15, 21, 16, 26]. In particular, these proposals particularly focus on relationships roles. On the other hand, concerning P4 and meeting expectations, TBAC models express the proposed trust relationship. Finally, only J. Park *et al.* model successfully achieves the specification of P7. In general, except for J. Park *et al.* model, ACMs do not include disjunctives management and then, the creation of as many access control policies as sentences connected by disjunctives is required. Furthermore, the management of multi-valued attributes is also a challenging matter. Again, [10] is the only one which deals with multi-valued attributes. A. Masoumzadeh *et al.* [15] and T. Abdessalem *et al.* [23] models could be quite easily modified to manage this kind of attributes but currently, they do not deal with them. As a final remark, despite the unreachable generalization of F6 (pointed out in Appendix A), much more improvements must be performed to express assorted preferences using, simultaneously, disjunctives and conjunctives together with different attributes.

In the light of this analysis, [15], [16] and [10] models are the most expressive for social networking applications. Their level of *semantic equivalence* is significant and they allow to define a lot of features. The first pair of proposals focus on relationships while the latter bases on attributes management. Consequently, it is asserted that the development of expressive ACMs for WBSNs has to go towards the management of relationships, as well as the management

Table 2: Expressive power comparison of ACMs

	Policies						
	P1	P2	P3	P4	P5	P6	P7
WBSN Models	*(Distance)* — *(Flexible attributes)*	*(Common-contacts)*	*(Clique)*	*(Multi-path)* — *(Flexible attributes)*	*(Direction:Bidi.)*	*(Direction:Unidi.)*	*(Flexible attributes)*
RBAC							
J. Li *et al.* [20]					√	√	
A. Tapiador *et al.* [27]					√	√	
A. Ahmad *et al.* [25]						√	
TBAC							
B. Ali *et al.* [28]				— $\mathcal{P}$	√		
B. Carminati *et al.* [21]	√ —			— $\mathcal{P}$		√	
T. Abdessalem *et al.* [23]	√ —			— $\mathcal{P}$	$\mathcal{P}$	√	$\mathcal{P}$
W. Villegas *et al.* [31]	$\mathcal{P}$ —			— √		√	
H. Wang *et al.* [29]	$\mathcal{P}$ —			— √		√	
RelBAC							
P. Fong *et al.* [16]	√ — $\mathcal{P}$	√	√	√ —	√	√	
P. Fong *et al.* [24]	$\mathcal{P}$	√	√	$\mathcal{P}$ —	√		
Y. Cheng *et al.* [26]	√ — $\mathcal{P}$				√	√	
H. Hu *et al.* [30]	$\mathcal{P}$				√		
I.B. Dhia [32]	√ — $\mathcal{P}$			— √	√	√	$\mathcal{P}$
I.B. Dhia [33]	√ — $\mathcal{P}$			— √	√	√	$\mathcal{P}$
G. Bruns *et al.* [36]	√ — $\mathcal{P}$	√		$\mathcal{P}$ —	√	√	
ABAC							
J. Park *et al.* [10]	— √	√	√		√	√	√
CVD. Munckhof [22]						√	√
S. Jahid *et al.* [9]	— $\mathcal{P}$					√	
R. Baden *et al.* [8]	— $\mathcal{P}$					√	
S. Braghin *et al.* [34]	√ — $\mathcal{P}$					√	
OBAC							
A. Masoumzadeh *et al.* [15]	√ — $\mathcal{P}$	√	√	$\mathcal{P}$ —	√	√	$\mathcal{P}$
B. Carminati *et al.* [7]	√ —					√	
M. Alizadeh *et al.* [35]						√	

√ : A feature is completely expressed
$\mathcal{P}$: A feature is not completely expressed

of attributes, either being user, object or relationship attributes.

4. SoNeUCON$_{ABC}$: an expressive usage control model

Regarding the aforementioned expressive power analysis, the development of an expressive ACM, that addresses all features described in Section 3.2, is the following goal. It should be noticed that [10], [15] and [16], selected as the most expressive models, are proposals to consider.

Expressive power is mainly related to the management of relationships, users, objects and their respective attributes. Thus, an ABAC model seems to be an appealing model to work with. On the other hand, in contrast to current ACMs that focus on protecting resources until the access is granted, new developments require to manage access along the whole usage process [17]. For instance, undesirable copies or unnoticed dissemination of data should be avoided. In this regard, given than [15] and [16] mainly base on relationships and put aside usage control and [10], based on $UCON_{ABC}$ [1], manages users, objects and usage control, it is the latter one the chosen proposal to start up. Extending [15] or [16] would require, among other issues, the complex task of converting them into usage control models. Consequently, $UCON_{ABC}$ is extended by including relationship management and thus, $SoNeUCON_{ABC}$, an expressive model for WBSNs, is proposed.

Although the $SoNeUCON_{ABC}$ model has been briefly introduced in [53], the work presented herein comprises, after a brief introduction to $UCON_{ABC}$ (Section 4.1), the formalization of the model (Section 4.2), the specification of a policy language (Section 4.3) and the description of the enforcement functions (Section 4.4).

4.1. $UCON_{ABC}$ model

The $UCON_{ABC}$ model considers eight components: *subjects (S)*, that are entities that exercise rights on objects; *objects (O)*, that are entities which subjects hold rights on; *subject attributes (ATT(S))* and *object attributes (ATT(O))* that refer to features associated with subjects and objects, respectively; *rights (R)*, which are recognized as privileges exercised on objects such as read or write; *Authorizations (A)*, that correspond to predicates on subject and object attributes that are evaluated in order to decide whether the requested right on a specific object made by a certain subject should be allowed or denied; *oBligations (B)*, that represent predicates that must be satisfied before, during or after the right is granted; and *Conditions (C)*, that correspond to environmental or system factors which are taken into account during the access decision process.

Additionally, $UCON_{ABC}$ introduces a pair of decision properties that are applied in respect to A, B and C. First, *continuity* refers to the enforcement of the usage decision along the whole usage period (pre/ ongoing). Second, *mutability* refers to changes produced in attributes along the usage process (pre/ ongoing/ post). Consequently, the usage has to be revoked when policies become unsatisfied.

In the original $UCON_{ABC}$ model, it is assumed that an access control policy is defined by the system's administrator and this policy is applied to all users in the system. A recent work by Salim *et al.* propose an administrative model, orthogonal to the $UCON_{ABC}$ model, where the attributes and rights of subjects and objects are established through assertions made by authorized subjects [52].

4.2. Formalization of $SoNeUCON_{ABC}$

Recalling that a WBSN can be defined as a graph, G, in which nodes (V) corresponds to users, edges (E) to relationships and data refer to elements that are somehow associated with users. Assuming this structure, the proposed model called $SoNeUCON_{ABC}$, mainly focuses on managing users, objects and relationships attributes while hiding users identity.

Contrary to $UCON_{ABC}$ which only manages direct relationships [54], the $SoNeUCON_{ABC}$ access control model extends the $UCON_{ABC}$ model by including a new independent entity, *relationships* (RT), and its attributes, *relationships attributes* ($ATT(RT)$). The new entity, RT, is composed of the sets of relationships (direct and indirect as described below) between pairs of users. Original entities, attributes and functions considered in the $UCON_{ABC}$ model are also considered in the $SoNeUCON_{ABC}$ model. Access control is now managed through the establishment of policies defined over $ATT(S)$, $ATT(O)$, $ATT(RT)$, rights (R), obligations (B) and conditions (C). Note that graph terminology is applied according to [55].

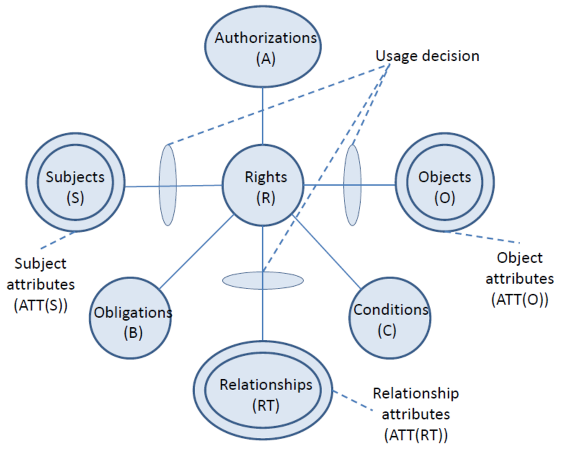

Figure 3: $SoNeUCON_{ABC}$

- *Subjects* (S) are the WBSN users (V) who play the role of requesters. User attributes can be derived with the mapping $vAT : S \longrightarrow ATT(S)$. Notice that vAT includes multiple individual mappings such that $vAT_i : S \longrightarrow ATT(S)_i$. For example, $vAT_1 : S \longrightarrow AGE$ where AGE contains all possible values for the AGE attribute.

- *Objects* (O) are WBSN data (D) that refer to as photos, videos, wall messages and personal messages and they are also called resources. Additionally, they have attached a set of object attributes which can be derived with the mapping $dAT : O \longrightarrow ATT(O)$. For instance, a possible individual mapping is $dAT_1 : O \longrightarrow TITLE$.

- *Relationships* (RT) represent the set of existing relations between the users of the WBSN. Under the approach taken in this work, given a request (s, o, r) (where s is the requester, o the requested object and r the requested right over o), a specific set of relationships rt is considered to manage access control. In particular, rt is defined as the set of relations that exist between the administrator a of the requested object o and the requester s.

  Consider that a *path* p in G is a sequence of alternating vertices $v_i \in V$ and edges $e_i \in E$ such that $p_i = (v_{i_0}, e_{i_1}, v_{i_1}, \ldots, v_{i_{j-1}}, e_{i_j}, v_{i_j}, \ldots e_{i_k}, v_{i_k})$ where i is the path identifier, j the node order and k the length of the path. Let $\mathcal{P}$ be the set of all simple strong paths in G. A path p_i is said to be *simple* if no node occurs more than once, i.e., $\forall i_q, i_r$ where $q, r \in \{0, \ldots, k\}$, if $i_q \neq i_r$, then $v_{i_q} \neq v_{i_r}$. A path p_i is said to be *strong* if for all nodes v_{i_j} in the path, except for the initial node v_{i_0}, there exists an edge e_{i_j} in the path such that $e_{i_j} = (v_{i_{j-1}}, v_{i_j})$. In Figure 4, a simple strong path may be found between v_8 and v_1 using the edges referred to as e_{13}, e_{21} and e_7, $(v_8, e_{13}, v_5, e_{21}, v_3, e_7, v_1)$. Note that in the case of the figure the number in the subscripts is just an identifier of the specific node or edge.

Let V_{p_i} and E_{p_i} be respectively the node set and edge set of path p_i. Then, p_i^*, the enriched path of p_i, is built by adding to E_{p_i} the edges of G that link, either forwards or backwards, two consecutive nodes of path p_i. Thus, $\mathcal{P}^*$ is the set of enriched paths p_i^* built from paths in $\mathcal{P}$. Enriched paths are represented as $p_i^* = (v_{i_0}, (e_{i_1}^1, \ldots; e_{i_1}^2, \ldots), v_{i_1}, \ldots,$ $v_{i_{j-1}}, (e_{i_1}^{l_j}, \ldots; e_{i_2}^{l_j}, \ldots),$ $v_{i_j}, \ldots,$ $v_{i_{k-1}}, (e_{i_k}^1, \ldots; e_{i_k}^2, \ldots), v_{i_k})$, being l_j the multiplicity of edges between nodes $v_{i_{j-1}}$ and v_{i_j} and k the length of the path. Note that edges that link two consecutive nodes of the path have been classified in two groups (separated by a semicolon), that contain respectively the forward and backward edges. Symbol $\emptyset$ indicates that no edge exists in that direction. For instance, in Figure 4, given the consecutive nodes set $\{v_8; v_5; v_3; v_1\}$, p_1^* is an enriched path such that $(v_8, (e_{13}; \emptyset), v_5, (e_{21}; e_{22}), v_3, (e_7; e_6, e_5), v_1)$. Note that edge e_{12} is not included as although it links two nodes of path p_1^*, they are not consecutive. This example highlights that enriched paths facilitates expressive power management because they allow navigating along all the relationships between a pair of users, no matter if they are direct or indirect.

Let v_a and v_s be respectively the nodes representing the administrator a of the requested object o and the requester s. Then rt is built as the set of all enriched paths, from $\mathcal{P}^*$, that allow reaching node v_s from node v_a. It is considered that each enriched path in rt is a relationship between v_a and v_s. In the example shown in Figure 4, rt for $v_a = v_8$ and $v_s = v_1$ is formed by the vertex set $V_{rt} = \{v_8, v_4, v_2, v_7, v_6, v_5, v_3, v_1\}$ and the edge set E_{rt} is composed by all continuous arrows.

Attributes of the relationships between v_a and v_s are derived from the attributes of the direct relationships that compose them. The set of attribute values associated to direct relationships is denoted as $ATT(E)$ and can be derived with the mapping $eAT : E \longrightarrow ATT(E)$. As in the case of the other attribute mappings, eAT is composed by multiple individual mappings; for example, $eAT_i : E \longrightarrow ROLE$,where $ROLE$ contains all possible values for the $ROLE$ attribute. As enriched paths contain one or more edges, the concatenation of the attributes of these edges constitute the set of attributes of the enriched path and they can be derived with the mapping $pAT : P^* \longrightarrow ATT(P)$ with $ATT(P) = ATT(E)^{n_e}$ and n_e the number of edges in the considered enriched path. The structure of the enriched path (number of edges between two consecutive nodes and their direction) needs to be kept, therefore, the set of its attributes may be represented following the notation used to represent an enriched path but excluding the nodes. Consider, for example, the enriched path p_1 such that $(v_8, (e_{10}, e_{11}; \emptyset), v_4, (e_2; e_1), v_1)$. Let's assume

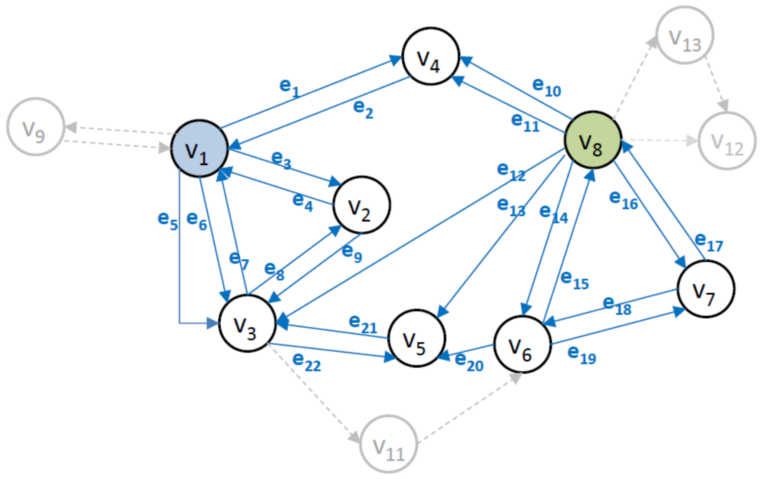

Figure 4: Relationships example, $G_{RT[v_8,v_1]}$

that only edge attributes of $ROLE$ and $TRUST$ are considered, and that their values for the edges in p_1 are $eAT(e_{10}) = (friend, 3)$, $eAT(e_{11}) = (colleage, 2)$, $eAT(e_2) = (friend, 4)$ and $eAT(e_1) = (friend, 3)$. Then, the attributes of the enriched path p_1 will be represented as $pAT(p_1) = (((friend, 3), (colleage, 2); \emptyset), ((friend, 4); (friend, 3)))$. Therefore, a set of relationships, rt, has also associated a set of attributes $ATT(RT)$, which can be derived with the mapping $rtAT : rt \longrightarrow ATT(RT)$ and built similarly from the attributes of its enriched paths.

Note that although rt is defined as the set of enriched paths between v_a and v_s, the relationships information could be condensed in the subgraph induced from the nodes involved in those relationships.

- *Rights* (R) refer to the actions (AC) that can be performed over WBSN data such as read, update or delete.
- *Authorizations* (A) are the rules defined as functional predicates over $ATT(S)$, $ATT(O)$, $ATT(RT)$ and R, that have to be satisfied, before or while the usage process, in order to grant a right to a subject on an object.
- *Obligations* (B) refers to requirements that users have to satisfy before or while the usage process.
- *Conditions* (C) correspond to requirements regarding the system or the environment status that have to be satisfied before or while the usage process.

A challenging point is the definition of how access control enforcement is carried out. It has to be noticed that usage control is essential in WBSNs because they are dynamic systems where access control conditions may change at any time [17]. For instance, the age or the hobbies of a user may change at a particular time. Analogously, a relationship may expire inbetween two access requests. However, the continuity of the enforcement process has to be ensured. In general, access control is based on the establishment and satisfaction of access control policies which consist of A – ($ATT(S)$, $ATT(O)$, $ATT(RT)$, R) – B and C. The enforcement is carried out by a *reference monitor*, which is the core component of access control systems. The location of the reference monitor is analysed next when a specific ACM is selected regarding the $UCON_{ABC}$ properties of *mutability* and *continuity*. According to pre-A/pre-B/pre-C and post-B/post-C models, the reference monitor is located at the server-side, that is, where access control policies are stored and evaluated. On the other hand, enforcement concerning ongoing-A/ongoing-B/ongoing-C models requires the introduction of a Usage Decision Facility (UDF) and a Usage Enforcement Facility (UEF) which are always active [56]. UDF identifies changes in attributes and accordingly, UEF enforces access control. Besides, though it must be studied in respect to particular implementations, UDF may be located at the server-side to detect attributes changes and UEF may be located at the client-side (i.e. in the browser) to enforce access control along the usage period, though client-side operations, particularly those related to enforcement, must be properly designed and implemented. As a matter to notice, developers must be aware of the most critical web application security risks ([57]) in order to face them.

Next, it is discussed who determines attributes and rights and how revocation is managed (which are questions raised in [52]). In this ACM, values in regard to a total of six elements ($ATT(S)$, $ATT(O)$, $ATT(RT)$, R, B and C) have to be specified. First, identity providers (IdPs) are in charge of providing $ATT(S)$ to users, though it is highly probable that they are provided by the system (WBSN) and users set them. Second, $ATT(O)$ values are defined by users and/or the system. Attributes such as the object's title are specified by users and attributes like object's size are established by the system. Third, analogous to $ATT(O)$, $ATT(RT)$ values are specified by users and/or the system. Fourth, the system defines available R. Fifth, analogous to R, the system defines available B. Last of all, C is also defined by the system. On the other hand, in respect to revocation, if administrators or requesters change attributes, modifications become effective according to the applied model, after or while an access request.

Lastly, it is remarkable that in this model the unique data managed are the attributes of the requester, of the administrator of the requested object and of the relationships between the administrator and the requester. Therefore, attributes of the rest of nodes involved in indirect relationships remain hidden when access control is enforced.

4.3. Access control policies

A request is expressed as $\{s, r, o\}$. In $SoNeUCON_{ABC}$, access control policies (ρ) are defined in terms of $ATT(S)$, $ATT(O)$, $ATT(RT)$, R, B, and C. As illustrated next, the requested $r \in R$ over $o \in O$ is granted if the values of $ATT(S)$ of the requester, $ATT(O)$ of the requested object and $ATT(RT)$ the set of relationships between the requester and the administrator satisfy the appropriate ρ. Note that although the general definition of access control policies is based on $ATT(RT)$, given that $ATT(RT)$ is built from $ATT(E)$, it is this last set of attributes the one directly managed during policies specification and enforcement. Therefore, the key matter is the definition of the set of attributes types regarding $ATT(S)$, $ATT(O)$ and $ATT(E)$, detailing possible operators, as well as, the way of constructing policies.

4.3.1. Policy attributes

Attribute types can be classified as follows:

- Boolean ($\mathcal{B}$). Only one value out of a pair can be taken by these attributes. For instance, the subject attribute *married* can take values "Yes" or "No".
- Free-valued ($\mathcal{FV}$). These attributes can take a value from a set of multiple possibilities. This attribute type is divided in two groups:
  - Numeric ($\mathcal{M}$): it refers to attributes which have numeric values. For example, the subject attribute *age* may take values from 15 to 99.
  - String ($\mathcal{S}$): it corresponds to attributes whose values are strings of characters. A key example is the direct relationship attribute *role* which takes values "friend", "colleague", "relative" and so on.
- Data structures ($\mathcal{D}$): it refers to structures in which numeric, string and boolean attributes can be combined. For instance: the subject attribute *hobbies*, which may be considered multivalued, can be created as a list which takes values following the pattern "$att_iValue_1, att_iValue_2, ..., att_iValue_n$", where n is the maximum number of possible values.

4.3.2. Policy operators

Regarding attributes management, the following set of operators can be applied:

1. *Logical operators* ($\mathcal{L}$) $::= \wedge | \vee | \neg$. Operator $\wedge$ represents a logic AND, operator $\vee$ refers to a logic OR and operator $\neg$ represents negation. Operators $\wedge$ and $\vee$ can be applied to two elements that can be attributes of boolean type or boolean expressions result of applying a relational or complex operator (see below). Operator $\neg$ can be applied to one of these elements.
2. *Relational operators* ($\mathcal{T}$) $::= < | > | \leqslant | \geqslant | =$. These operators are applied to two attributes of numeric or string type. For example, given the subject attribute *age*, the age of a pair of users, v_1 and v_2, can be compared by building the expression $age(v_1) < age(v_2)$ to identify if v_1 is younger than v_2. The result of applying these operators is a boolean value. Note that a concrete specification of string types management should defined according to each particular context.

3. *Complex operators* ($\mathcal{X}$). These operators are an open set based on the combination of logic and relational operators and they are applied to data structures. For instance, assuming the existence of a list of attribute values such that $\{att_iValue_1, att_iValue_2, ..., att_iValue_n\}$, $x_i \in \mathcal{X}$ can be defined as an operator which goes recursively through the list until identifying a particular attribute value. The result should be also a boolean value.

4.3.3. Policy construction

Concerning the above operators, constraints and policy attributes, each policy (ρ) in $SoNeUCON_{ABC}$ model is generally depicted as $\rho(\rho_s; \rho_o; \rho_{rt}; r; \partial_b; \partial_c)$.

More specifically, ρ_s, ρ_o and ρ_{rt} correspond to predicates ($\alpha_{att(w)_i}$) built using operators applied to the attributes of a subject, $ATT(S)$, an object, $ATT(O)$, and the set of relationships between the subject and the administrator of the requested object, $ATT(E)$, where w in $\alpha_{att(w)_i}$ can take three possible values – s, o or e –, to indicate the set of attributes that the attribute used in the expression belongs to. Moreover, ∂_b and ∂_c refer to sets of obligations and conditions. Policies are formally described through BNF notation [58]. For the sake of simplicity, when a specific mapping on an entity should be used to obtain the value of an attribute (e.g., $age(v)$, $title(o)$, or $trust(e_i)$), only the name of the mapping will be used (i.e., *age*, *title*, or *trust*), as the policy construction allows to clearly identify the entity to which the mapping should be applied in each case.

- Regarding ρ_s and ρ_o, ρ_s (analogously ρ_o) is defined as follows:
  - ρ_s ::= (‘$\emptyset$’|([‘$\neg$’] $\alpha_{att(s)_1}$) { ‘$\wedge$’|‘$\vee$’ ([‘$\neg$’] $\alpha_{att(s)_j}$)}*)
    For instance: $\rho_s = (age > 18 \wedge student = uc3m)$
- By contrast, ρ_{rt} is composed of predicates built over $ATT(RT)$. It consists of a list of three elements: the first one (σ) corresponds to the set of conditions of paths that must be satisfied; the second one (ϖ) refers to the number of times that the enriched path represented by σ should exist in rt when σ refers to a single enriched path; and the last one (δ) corresponds to the number of nodes involved in a clique. Indeed, δ is directly related to the number of different bidirectional enriched paths that exist in a clique with the formula ($\sum_{K=1}^{N} P(K,N) + 1$) where $N = \delta - 2$ (N is the number of members of the clique excluding v_a and v_s) and $P(K,N)$ refers to the number of K-permutations in a set of N elements.
  - ρ_{rt} ::=‘$\emptyset$’ | (σ, ϖ, δ)
  - σ ::= (ψ {‘$\wedge$’|‘$\vee$’ ψ}*)
    - ψ ::= ($fert|bert$|‘$\emptyset$’ {‘$\wedge$’|‘$\vee$’ $fert|bert$ {; $fert|bert$|‘$\emptyset$’ {‘$\wedge$’|‘$\vee$’ $fert|bert$}*}*}*), where $fert$ and $bert$ refers, respectively, to predicates built over attributes of forward and backward direct relationships between pairs of users.
      - $fert$ ::= ([‘$\neg$’] $\alpha_{att(e)_i}$ {‘$\wedge$’|‘$\vee$’ [‘$\neg$’] $\alpha_{att(e)_i}$ }*)

        · $bert$ ::= $-fert$

    For example: $\rho_{rt} = ((((role = friend) \wedge -(role = friend)) \wedge ((role = relative))), \emptyset, \emptyset)$ refers to the existence of a pair of enriched paths. One of them corresponds to a forward and a backward friendship relationship. The other one corresponds to a forward relative relationship.

  – ϖ ::='$\emptyset$'$|n$, where $n \in \mathbb{Z}$ and $n > 2$. Note that in case ϖ takes value n, σ must be composed of a single ψ to identify n occurrences of a particular enriched path. Otherwise ϖ is $\emptyset$. For instance: $\rho_{rt} = \{\{\{\{role = friend\}; \{role = relative \wedge creationYear > 2010\}\}\}, 2, \emptyset\}$ expresses the necessity of existing, at least, a pair of enriched paths of length two where the first hop involves a forward friendship relationship and the second hop involves a forward relative relationship which must have been created after the year 2010.

  – δ ::='$\emptyset$'$|n$, where $n \in \mathbb{Z}$ and $n > 0$. Note that if δ takes a value distinct from $\emptyset$, then σ must correspond to a single ψ composed of a forward relationship. Besides, it is assumed the existence of a backward relationship of the same type as the forward one. Otherwise δ is $\emptyset$. For instance: $\rho_{rt} = ((((role = friend \wedge trust = high))), \emptyset, 3)$ corresponds to the existence of a clique of three users where the relationships between each pair of users are highly trusted and have the *role friend*.

- $r ::= write|read|....$
- ∂_b ::= ('$\emptyset$' | $\{abligation_n\}^*$).
- ∂_c ::= ('$\emptyset$' | $\{condition_n\}^*$).

Then, a right (r) to an object (o), which satisfies ρ_o, is allowed if the requester (s) satisfies ρ_s, the set of relationships between the administrator of o and the requester, directly or indirectly, satisfies ρ_{rt} and ∂_b and ∂_c are also satisfied: $allowed(s, o, r) \Rightarrow \rho(\rho_s; \rho_o; \rho_{rt}; r; \partial_b; \partial_c)$. Notice that components of ρ, except for r, can be empty $(\emptyset)$. This issue can be used to specify public policies or establish undetermined object, subject or relationship conditions. For instance: $\rho = (\emptyset; (title = party); (((\emptyset; \emptyset)), \emptyset, \emptyset); read; \emptyset; \emptyset)$ expresses that objects entitled "party" can be read by users with whom the administrator has some kind of relationship, as long as users are located at a maximum length of 2 hops.

On the other hand, concerning *mutability* and *continuity* management, policies can be analogously defined for pre and on-going usage processes, while post usage processes are bound to the specification of conditions and obligations. Specifically, they are defined as $\rho_{pre}(\rho_s; \rho_o; \rho_{rt}; r; \partial_b; \partial_c)$, $\rho_{on-going}(\rho_s; \rho_o; \rho_{rt}; r; \partial_b; \partial_c)$ and $\rho_{post}(\partial_b; \partial_c)$. Then, as long as attributes change and regarding the stage of the usage process, the appropriate access control policies are evaluated. Nonetheless, the unpredictability of conditions and obligations definition has to be discussed. For instance, in the WBSN Badoo, an obligation exists such that it is necessary to upload three photos to the personal profile before accessing to other users' albums. On the other hand, a devised obligation may require to have five contacts before accessing to other users' profiles. By contrast, a possible condition is referred to the system load capacity (e.g. free or busy), that is, accesses are rejected until the system is free. These examples illustrate the large set of possible conditions and obligations that can exist. Consequently, such variability together with the aim to get as much flexibility as possible, support the inappropriateness of providing a concrete specification of both types of elements.

4.4. Access control enforcement

Access control enforcement is carried out through the execution of a set of functions which, in this proposal, are linked to the proposed policy language.

Once a request $\{s, o, r\}$ is received, stored policies are retrieved. A pair of alternatives to enforce access control are identified. The first one bases on searching for enriched paths between a and s throughout the whole WBSN graph (G), and verifying the policies during the process. On the other hand, the second approach first builds the set rt. Afterwards, policies are verified considering this set. Given the difficulty in accurately measuring the complexity of the first alternative, the second alternative is adopted herein. This alternative allows the reuse of rt while verifying policies, as well as the calculation of the upper bound of the computational complexity concerning access control enforcement. Note that functions described herein are experimentally evaluated in Section 6.

Functions are described in Appendix C and their specifications base on the definitions presented in [59]. Along the functions descriptions, the names of functions that are called within a given one are pointed out in square brackets.

CheckAccess This is the main function. It evaluates each policy of the system against an object. If any of the policies is satisfied, then the result is "true" and the right on the object is granted. On the contrary the request is rejected. More specifically, as aforementioned, the request is composed of a subject (s), who is the requester, an object (o) and the requested right (r). Then, after the construction of rt [$CreateRT$] concerning a (the administrator of o) and s, for each existing policy five elements are verified. First, it is verified if attributes of s match the set of subject attributes of the policy (ρ_s) [$Match$]. Second, it is checked if attributes of o match the set of object attributes of the policy (ρ_o) [$Match$]. Third, it is verified if attributes of rt match the set of the relationship attributes of the policy (ρ_{rt}) [$MatchRT$]. Fourth, the match between the right within the policy and r is verified. Lastly, a pair of elements, conditions and obligations, are verified [$MatchC/MatchO$]. Note that, at first ρ_{pre} are the ones evaluated. However, to guarantee the continuity of the usage process, the last task of the function is to remain waiting along the usage process. For this purpose, another function is called [$ContinuityCheckAccess$].

ContinuityCheckAccess This function is rather similar to $CheckAccess$, being distinguished a pair of issues. First, rt is already computed and then, just policy elements have to be verified. Second, this function is called once attributes have changed or the usage process has concluded to evaluate $\rho_{on-going}$

and ρ_{post}. It should be noticed that the management of $\rho_{on-going}$ and ρ_{post} is quite related to particular implementations. Therefore, this function has to be integrated within an entity (such as, the UDF and/or the UEF recall Section 4.2) to adequately manage all requests and changes produced.

Match The goal of this function is to verify the match between each value of a set of ω attributes ($ATT(\omega)$) and ω attributes involved in a particular policy (ρ_ω) [$VerifyDAttTypes/\ VerifyFVAttTypes/\ VerifyBAttTypes$]. It returns "true" if all attributes predicates are satisfied and "false" otherwise. Notice that ω corresponds to the evaluation of a subject or an object.

CreateRT This is a recursive function that focuses on creating rt from the WBSN graph, G, given the administrator (a) and the requester (s) of a particular request. Departing from the administrator node a, the process starts by visiting each of the contacts of a [$GetNumContacts/\ GetConnectedUser$]. When the algorithm visits node v, the contacts of v are also visited [$CreateRT$] recursively until the length of the path between a and the node being currently visited reaches 6 or node s is found. Note that the maximum path length that is considered in the system is 6 due to theoretical studies [60]. Then, if the path length reaches 6, the algorithm continues visiting the remaining contacts of the previous node in the path if any. If node s is reached, then the corresponding enriched path is stored (with all its forwards and backwards relationships and its attributes).

MatchRT The goal of this function is to verify the satisfaction of ρ_{rt} given that rt is already built. The process consists of verifying the use of parameters ϖ and δ (involved in ρ_{rt}, that is $\rho_{rt}.\varpi$ and $\rho_{rt}.\delta$) and performing verifications accordingly. First, if $\rho_{rt}.\delta$ is not $\emptyset$, the existence of a clique is noticed [$VerifyClique$]. Second, if $\rho_{rt}.\varpi$ is not $\emptyset$, it is analysed the satisfaction of a set of enriched paths with the same attributes in all of them, that is, a single ψ in $\rho_{rt}.\sigma$ [$MatchPathPolicy$]. Finally, if $\rho_{rt}.\delta$ and $\rho_{rt}.\varpi$ are $\emptyset$, the satisfaction of a set of enriched paths with different attributes involved in each of them is verified, that is, several ψ in $\rho_{rt}.\sigma$ [$GetPathsPolicies/\ MatchPathPolicy/\ VerifyPolicy$]. Once the verification is successfully performed, the result is "true", and "false" otherwise.

MatchPathPolicy The goal of this function is to verify if an enriched path in rt matches a particular set of conditions of a path ($pathCond$) involved in a policy. The general process consists of four steps. First, it is calculated the length of $pathCond$ ($pLength$) [$GetLengthPath$]. Second, enriched paths of rt with the same length as $pLength$ are collected ($rtPathsL$) [$GetEnrichedPathsWithLength$]. Thirdly, paths in $rtPathsL$ are processed, getting the value of the attributes of the direct forward and backward relationships at every hop. Lastly, it is verified if attributes match with those of $pathCond$ [$MatchDirectPaths/\ GetDirectRelAtt$]. Once the verification is completely and successfully performed the result is "true", and "false" otherwise.

GetEnrichedPathsWithLength The goal of this function is to get a list of enriched paths of a particular length ($length$) from rt. A list is returned and "null" otherwise.

GetPathsPolicies The goal of this function is, given a set of conditions (σ)

involved in paths of a policy, to separate them creating a list of independent path policy (ψ_i), as well as creating a list of operators that join every ψ_i. Then, both lists are returned and "null" otherwise. Note that a key point is the order, that is, the list of operators has to be directly linked with the list of enriched paths and then, both lists have to be ordered regarding σ.

GetDirectRelAtt The goal of this function is, given conditions of a path (*pathCon*), to create a list composed of conditions of direct forward and backward relationships found at a particular position (*pLength*) of *pathCon*. The process bases on identifying the set of forward and backward relationships between a semicolon (";") found at ($pLength{-}1$) and a semicolon found at *pLength*.

GetErtDivision The goal of this function is, given a set of conditions of direct forward and backward relationships found at a particular position in a path involved in a policy (*rels*), to process *rels* separating and storing conditions of each direct relationship in a list, as well as storing operators that join these relationships conditions in another list. Then, both lists are returned and "null" otherwise. Analogous to *GetPathsPolicies*, the order is a key matter and thus, returned lists have to be built keeping the order of *rels*.

MatchDirectPaths The goal of this function is to verify that a set of direct forward and backward relationships found at a particular position of an enriched path of *rt* (*listPathsOfRT*) match conditions, at the same position, of a path involved in a policy (*pathsPolicy*). Once the verification is completely and successfully performed the result is "true", or "false" otherwise.

VerifyPolicy The goal of this function is, having evaluated all paths within a policy (list of boolean values, *listBoolean*), to verify if the set of operators applied in the policy to join paths (*listOpe*) appropriately match (*listOpe*). Once the verification is completely and successfully performed the result is "true", or "false" otherwise.

VerifyClique The goal of this function is to verify the existence of a clique. The process involves a set of four steps. First, it is calculated the number of paths of each particular length (*listlengthEP*) that are involved in a clique of a certain set of users (δ) [*CalculateCliqueUsers*]. Second, *rt* is analysed, storing all paths whose length matches those stored in *listlengthEP* (*pathsClique*) [*pathsDivided*/ *GetEnrichedPathsWithLength*]. Thirdly, *pathsClique* is processed to verify enriched paths whose direct forward and backward relationships match ρ_{rt} (σ). If they match, they are stored (*acceptedPaths*) [*MatchDirectPaths*/ *GetDirectRelAtt*/ *GetFirstNode*/ *GetLastNode*]. Lastly, the result is satisfactory if nodes involved in *acceptedPaths* do not exceed δ [*GetNode*]. Once the verification is completely and successfully performed the result is "true", or "false" otherwise.

CalculateCliqueUsers The goal of this function is to identify the number of paths of different lengths that the construction of a clique of a certain number of users (δ) requires. Positions of the returned list correspond to path lengths and values of the returned list refer to the number of paths of each length. This is calculated through counting sequences without repetition, that is, K-permutations in a set of N elements where N refers to the number of distinct nodes (including the administrator and the requester) involved in those paths.

A list is returned and "null" otherwise.

GetNode The goal of this function is to identify the node in a path (*path*) located at a certain position (*pos*). A node id is returned and "null" otherwise.

GetFirstNode/GetLastNode The goal of these functions refers, respectively, to return the id of the first and last node of a particular path. A node id is returned and "null" otherwise.

GetLengthPath The goal of this function is to identify the length of a given path (*path*). Considering that policies use operator semicolon to distinguish path lengths, the length *path* is obtained by counting all semicolons plus one.

MatchB/MatchC The goal of these functions focuses on verifying the satisfaction of conditions and obligations but they are very assorted and their implementation is left to systems' developers.

GetConnectedUser This function returns the id of a user directly connected to a given one. It is considered that contacts are stored ordered and then, they are returned according to a given position (*pos*). An id is returned or "null" otherwise.

GetNumContacts This function returns the number of contacts of a given user (s).

VerifyBAttTypes This function verifies the satisfaction of a particular attribute value ($\gamma^j_{att_i}$), being the attribute type $\mathcal{B}$, regarding particular policy predicates (*policyAttPred*). If the verification is successful it returns "true", or "false" otherwise.

VerifyFVAttTypes This function verifies the satisfaction of a particular attribute value ($\gamma^j_{att_i}$), being the attribute type $\mathcal{FV}$, i.e., $\mathcal{M}$ or $\mathcal{S}$, regarding particular policy predicates (*policyAttPred*). If the verification is successful it returns "true", or "false" otherwise.

VerifyDAttTypes This function verifies the satisfaction of a particular attribute value ($\gamma^j_{att_i}$), being the attribute type $\mathcal{D}$, regarding particular policy predicates (*policyAttPred*). If the verification is successful it returns "true", or "false" otherwise.

GetSubAtt/GetObjAtt These functions refer to "get" functions that return user attributes and object attributes respectively. They return attributes or "null" otherwise. Notice that input parameters depend on each function.

GetAdmin This function, taking as input an object (o), returns the subject who is its administrator. If the administrator of o is not found, "null" is returned.

5. Theoretical evaluation

This Section presents the evaluation concerning three issues. Firstly, it is analysed if the proposed model manages access control according to the set of identified WBSN features (*distance*, *multi-path*, *direction*, *common-contact*, *clique*, *flexible attributes*) described in Section 3.2 (Section 5.1). Secondly, the expressive power of the proposed policy language is studied (Section 5.2).

5.1. Capability of the model to consider WBSN features

Access control policies are specified in respect to a set of attributes of the requesting subject, the requested object and the set of relationships between the administrator and the requester (respectively, ATT(S), ATT(O) and ATT(RT)). Recalling the definition of $SoNeUCON_{ABC}$ and considering that the relationship management is an essential issue in this model, for a particular request of an action r made by a requester over an object o of an administrator a, the set of managed relationships, rt, corresponds to all enriched paths that have as initial node a and terminal node s. Next, it is analysed if the six WBSN features described in Section 3.2 can be addressed by the $SoNeUCON_{ABC}$ model.

Distance. It is immediate to show that the model can manage policies that consider indirect relationships between s and a, as rt contains, in theory, all the relationships (direct and indirect) between both entities.

Multi-path. Similarly, as rt contains all the relationships between a and s (in the form of enriched paths), the model allows the definition of access control policies that consider multiple paths.

Direction. In the same way, as the enriched paths comprising rt contain all the edges connecting two consecutive nodes in the path, in the forward and backward directions, it is possible in the model to define access control policies that consider unidirectional and bidirectional relationships.

Common-contacts. In a WBSN, assuming the existence of two users, a and s and a set V_l of common contacts, the existence of common-contacts between a and s can be considered in two different ways, where "$\cdots$" refers to the existence or not of additional edges:

1 a has a unidirectional and direct relationship with each $v_{l_i} \in V_l$ and each v_{l_i} has a bidirectional relationship with s. This will be represented by the following enriched path:
$\{a, (e^k_{a,l_i}, \cdots ; \cdots), v_{l_i}, (e_{l_i,s}, \cdots ; e_{s,l_i}, \cdots), s\}$

2 a and each v_{l_i} have bidirectional relationships with s. Thus, there are enriched paths between a and s such that:
$\{a, (e_{a,l_i}, \cdots ; e_{l_i,a}, \cdots), v_{l_i}, (e_{l_i,s}, \cdots ;\ e_{s,l_i}, \cdots), s\}$

As rt contains all the relationships (enriched paths) between a and s, a policy considering the existence of common contacts can be evaluated within the model.

Clique. A clique in a digraph D (i.e. directed graph) is referred to a complete digraph between a set of C nodes (including a and s). Then, a clique corresponds to the existence of different bidirectional relationships between all nodes involved in it. In particular, assuming that the number of nodes in the clique distinct than a and s is $N = C-2$, a clique exists if there are $\sum_{K=1}^{N} P(K,N)+1$ different enriched paths, such that only N distinct nodes plus a and s are involved in those paths and there exists a bidirectional direct relationship of the same type between all these nodes. Note that $P(K,N)$ refers to the number of K-permutations in a set of N elements. Then, in case $C = 2$ the number of paths is 1, in case $C = 3$, the required number of such paths is 2, for $C = 4$, it is 5, and for $C = 5$, 16 paths are required to exist.

Therefore, given rt, the model allows verifying the existence of a clique of N nodes, and then, it may support access control policies that consider cliques defined in such a way.

Flexible attributes. In this model, $ATT(S)$, $ATT(O)$, $ATT(RT)$, as well as B and C, are used to define access control policies. Then, flexible attributes are used to achieve the specification of fine-grained preferences.

Note that although the model allows managing access control in regard to this set of features, it is necessary to complement the model with an expressive policy language that also supports them.

5.2. Expressive power of the policy language

$SoNeUCON_{ABC}$ expressive power, according to the expressive power analysis presented in Section 6.4, is studied by determining the possibilities of the model to define policies presented in Section 3.2. It is assumed that the granted right over objects entitled "party" is "read" and not a single condition and/ or obligation has to be satisfied:

P1 Access is granted to users who are friends of neighbours of his/ her relatives if the relationship between his/ her relatives and his/ her relatives' neighbours was established before 2,000. (F1 and F6)
This policy corresponds to an indirect relationship composed of three direct forward relationships from a to s which involve the use of the attribute $role$ in each hop and the attribute $creationYear$ in the second hop.
$\rho = (\emptyset;\ (title = party);\ ((((role = relative);\ (role = neighbour \wedge creationYear < 2000); (role = friend))), \emptyset, \emptyset); read; \emptyset;\ \emptyset)$

P2 Access is granted to users who have three friends in common with the administrator of the requested object. (F2)
One possible option of being common-contact (F4) refers to the existence of an enriched path between a and s where the first hop refers to a direct forward relationship from a to one of his contacts (F3-unidirectional) and the second hop refers to a forward and a backward relationship between such contact and s (F3-bidirectional). Besides, as 3 common-contacts are required, a total of 3 analogous enriched paths have to be identified (F2) and thus, ϖ takes value 3.
$\rho = (\emptyset;\ (title = party);\ ((((role = friend);\ (role = friend) \wedge -(role = friend))),\ 3,\ \emptyset);\ read;\ \emptyset;\ \emptyset)$

P3 Access is granted to users who belong to the clique in which two users and the administrator of the requested object are involved, having all of them a friendship relationship. (F3)
This policy corresponds to the existence of two enriched paths, one between a and s and one that includes a, s and other user (F5). Then, δ is 3 because 3 users are involved in the clique. Additionally, enriched paths are composed of a direct forward friendship relationship (F6) and implicitly, a backward one.
$\rho = (\emptyset;\ (title = party);\ ((((role = friend))),\ \emptyset,\ 3);\ read; \emptyset;\ \emptyset)$

P4 Access is granted to users who are connected to the administrator by two different paths composed of unidirectional relationships oriented from the requester to the administrator. Moreover, relationships involved in all paths have to be highly trusted. (F4 and F6)

This policy refers to the existence of, at least, a pair of paths with a certain kind of constraints regarding the level of trust of the relationship.

$\rho = (\emptyset;\ (title = party);\ ((((trust = high))),\ 2,\ \emptyset);\ read; \emptyset;\ \emptyset)$

P5 Access is granted to users who are friends of the administrator of the requested object, also having a bidirectional relationship with him/ her. (F5)

This policy refers to the specification of a direct forward and a direct backward friendship relationship.

$\rho = (\emptyset;\ (title = party);\ ((((role = friend) \wedge -(role = friend))), \emptyset,\ \emptyset);\ read;\ \emptyset;\ \emptyset)$

P6 Access is granted to users who are friends of the administrator of the requested object. (F5)

Apart from conditions and obligations, subject and relationship attributes take value $\emptyset$.

$\rho = (\emptyset;\ (title = party);\ \emptyset;\ read;\ \emptyset;\ \emptyset)$

P7 Access is granted to users if they are females under 30 years old or if they are females under 40 who have studied computer science or if they are females who have studied computer science and physics. (F6)

$\rho = (((gender = female) \wedge ((age < 30) \vee ((age < 40) \wedge (studies = c.science)) \vee ((studies = c.science) \wedge (studies = physics))));\ (title = party);\ \emptyset;\ read;\ \emptyset;\ \emptyset)$

All policies are satisfactorily expressed by $SoNeUCON_{ABC}$. Thus, the suitability of the model for the WBSN field is recognized.

6. Empirical evaluation

The feasibility of implementing $SoNeUCON_{ABC}$ is analysed by the development of a proof of concept system which also helps to study policy enforcement Temporal Workload (TW). Firstly, four WBSNs are randomly constructed. Table 3 depicts the number of nodes ($\#v_i$), the number of relationships ($\#e_i$) and the mean number of relationships per node ($\overline{e_i/v_i}$) that each WBSN involves. Then, based on developed WBSNs, policy enforcement is studied. It is assumed that the number of hops between a pair of WBSN users is limited to 6 due to theoretical studies [60].

Table 3: WBSNs structure

WBSNs id	$\#e_i$	$\#v_i$	$\overline{e_i/v_i}$
1	2,980,388	50,000	60
2	5,965,777	50,000	120
3	8,949,375	50,000	185
4	10,929,713	50,000	219

The experimental study of policy enforcement is divided in two steps:

1. *Analysis of rt construction*: For each WBSN, a total of 7 *rt* structures are constructed choosing random requesters and administrators. Table 4 details the number of relationships ($\#e_i$ *explo.*) and nodes explored ($\#v_i$ *explo.*) for constructing *rt*, the number of relationships (*rt* $\#e_i$) and nodes (*rt* $\#v_i$) that each final *rt* comprises and the TW of constructing *rt* (*rt* $TW(ms)$). Note that even all *rt* are constructed choosing random users, the amount of nodes and relationships involved in them are considered sufficient to guarantee the appropriateness of the evaluation process.
2. *Policy evaluation*: In each constructed *rt*, policies proposed in Section 5.2 are independently evaluated. The TW of performing policy evaluation is summarized in Table 5.

Concerning technical details, the proof of concept system is developed in Java 1.7, using a MySQL 5.2 database to store nodes and relationships. Moreover, experiments have been executed over a Pentium D 2.3 GHz with a Lion 10.8 operating system using 500 MB of RAM.

6.1. Analysis of rt construction

The TW of building *rt* increases exponentially according to the number of explored nodes, that is, *rt* is built by visiting, recursively, all contacts of each user (starting from the administrator) until the requester is reached (or the maximum path length is reached). Consequently, the TW of constructing *rt* increases according to the sum of all visited users at each path length, that is, $\sum_{i=1}^{K}(\eta^i)$ where η refers to the average number of users' contacts and K corresponds to the path length. Table 4 depicts the TW of constructing multiple *rt* in each proposed WBSN. In the worst analysed situation, $rt\ id = 22$, the TW exploring 10,929,713 relationships is 105,478 ms. By contrast, in a better situation, for example, in $rt\ id = 12$, the TW exploring 119 relationships is 60 ms.

Nonetheless, it should be noticed that, under certain circumstances, some *rt* involve more nodes and relationships than those that generally appear in a real scenario. Taking Facebook as a representative WBSN, assuming that the average number of Facebook contacts is 190 [61] and the maximum number of hops are two (friend-of-a-friend), the average maximum number of relationships and nodes among a pair of users is $190 + 190^2 =$ 36,290. Consequently, *rt* whose creation involves the exploration of more than 36,290 relationships exceed the average case.

6.2. Policy evaluation

Proposed access control policies are evaluated and Table 5 depicts the TW of their evaluation. All access control policies, except for P3 that refers to cliques construction, are quickly evaluated reaching a TW lower than 90 ms. The most significant *rt* to analyse, with the highest number of relationships and nodes, are $rt\ id = 1,\ 8,\ 15$ and 22. They involve 83, 164, 502 and 751 relationships and 49, 80, 170 and 251 nodes respectively. Policy evaluation concerning these *rt* does not exceed 100 ms but for P3. Evaluating policy P3 takes 3,316 ms in

Table 4: Analysis of rt construction

WBSN id = 1					
rt id	$\#e_i$ explo.	$\#v_i$ explo.	$\#e_i$ *rt*	$\#v_i$ *rt*	*rt* TW (ms)
1	252,691	505,382	36	24	4,142
2	3,662	7,324	4	4	435
3	81	162	1	2	28
4	79	158	2	2	54
5	69	120	1	2	38
6	65	130	1	2	51
7	1	2	1	2	13
WBSN id = 2					
rt id	$\#e_i$ explo.	$\#v_i$ explo.	$\#e_i$ rt	$\#v_i$ rt	rt TW (ms)
8	1,958,163	3,916,326	164	80	21,287
9	13,557	27,114	6	5	712
10	139	278	1	2	57
11	126	252	2	2	88
12	119	238	1	2	60
13	115	230	1	2	62
14	1	2	1	2	30
WBSN id = 3					
rt id	$\#e_i$ explo.	$\#v_i$ explo.	$\#e_i$ rt	$\#v_i$ rt	rt TW (ms)
15	6,163,496	12,326,996	502	170	56,811
16	29,771	49,542	6	5	573
17	201	402	1	2	80
18	187	374	2	2	110
19	174	348	1	2	88
20	174	348	1	2	86
21	37	74	1	2	36
WBSN id = 4					
rt id	$\#e_i$ explo.	$\#v_i$ explo.	$\#e_i$ rt	$\#v_i$ rt	rt TW (ms)
22	10,929,713	22,231,690	751	251	105,478
23	445,839	91,678	8	6	1,721
24	244	488	1	2	44
25	230	460	2	2	134
26	216	432	1	2	96
27	216	432	1	2	83
28	33	66	1	2	46

rt id = 1, more than 100,000 ms in *rt id* = 11 and more than 200,000 ms in *rt id* = 15 and *rt id* = 22.

In sum, policy evaluation TW is satisfactory and just cliques management has to be discussed. The attained results may be justified by the fact that the implemented algorithm for searching cliques is not efficient enough and by the fact that experiments are executed in a computer with limited resources.

Table 5: Policies evaluation temporal workload

WBSN id = 1							
rt id	P1-TW (ms)	P2-TW (ms)	P3-TW (ms)	P4-TW (ms)	P5-TW (ms)	P6-TW (ms)	P7-TW (ms)
1	1	1	3,316	1	1	<1	<1
2	<1	<1	717	<1	<1	<1	<1
3	<1	<1	247	<1	<1	<1	<1
4	<1	1	753	<1	<1	<1	<1
5	<1	<1	559	<1	<1	<1	<1
6	<1	<1	571	<1	<1	<1	<1
7	<1	<1	463	<1	<1	<1	<1
WBSN id = 2							
rt id	P1-TW (ms)	P2-TW (ms)	P3-TW (ms)	P4-TW (ms)	P5-TW (ms)	P6-TW (ms)	P7-TW (ms)
8	4	17	>100,000	4	2	2	<1
9	<1	<1	1,786	<1	1	<1	<1
10	1	<1	355	<1	<1	1	<1
11	<1	<1	1,138	1	1	<1	<1
12	1	<1	822	<1	<1	1	<1
13	<1	<1	843	1	<1	<1	<1
14	1	<1	729	<1	<1	<1	<1
WBSN id = 3							
rt id	P1-TW (ms)	P2-TW (ms)	P3-TW (ms)	P4-TW (ms)	P5-TW (ms)	P6-TW (ms)	P7-TW (ms)
15	14	5	>200,000	4	2	9	<1
16	1	<1	3,026	1	<1	<1	<1
17	<1	<1	634	<1	1	<1	<1
18	<1	1	1,634	<1	<1	<1	<1
19	1	<1	1,067	1	<1	<1	<1
20	<1	<1	1,014	<1	<1	1	<1
21	<1	1	959	<1	1	<1	<1
WBSN id = 4							
rt id	P1-TW (ms)	P2-TW (ms)	P3-TW (ms)	P4-TW (ms)	P5-TW (ms)	P6-TW (ms)	P7-TW (ms)
22	71	67	>200,000	76	80	18	85
23	<1	1	9,076	<1	<1	<1	1
24	<1	1	1,952	<1	<1	<1	1
25	1	<1	3,683	<1	<1	<1	1
26	<1	1	2,198	<1	<1	<1	1
27	<1	<1	2,149	<1	<1	<1	1
28	<1	<1	1,396	<1	<1	<1	1

6.3. Summary: policy enforcement

Table 6 presents the TW of the policy enforcement process, that is, the sum between steps *Analysis of rt construction* and *Policy evaluation*. The tolerable waiting time of WBSN users for information retrieval is approximately 2,000 ms [18]. Thus, results of the implemented proof of concept system are successful in most cases. Particularly, they are satisfactory enforcing policies without cliques, if explored nodes do not exceed about 200,000 and 200 relationships per node.

Besides, the enforcement of policies with cliques remains successful if less than about 30,000 nodes and 200 relationships per node are explored.

Concerning *rt id* = 1, 8, 15, 16, 22, 23 and 25, that exceed 2,000 ms, some points are discussed to justify such results. Firstly, some *rt* may involve the exploration of more quantity of nodes and relationships than those that, on average, take place in WBSNs like Facebook. Secondly, despite the hard task of cliques evaluation due to the amount of paths to analyse (recall Section 5.1), the implemented algorithm could be enhanced to increase performance and reduce the TW. Lastly, contrary to the developed proof of concept system, WBSNs like Facebook apply huge and powerful servers which facilitate the celerity of the policy enforcement process.

Table 6: Policy enforcement temporal workload

WBSN id = 1							
rt id	P1-TW (ms)	P2-TW (ms)	P3-TW (ms)	P4-TW (ms)	P5-TW (ms)	P6-TW (ms)	P7-TW (ms)
1	4,143	4,144	7,458	4,143	4,143	4,142	4,142
2	435	435	1,152	435	435	435	435
3	28	28	275	28	28	28	28
4	54	55	807	54	54	54	54
5	38	38	597	38	38	38	38
6	51	51	622	51	51	51	51
7	13	13	476	13	13	13	13
WBSN id = 2							
rt id	P1-TW (ms)	P2-TW (ms)	P3-TW (ms)	P4-TW (ms)	P5-TW (ms)	P6-TW (ms)	P7-TW (ms)
8	21,291	21,304	>121,287	21,291	21,289	21,289	21,287
9	712	712	2,498	712	713	712	712
10	58	57	412	57	57	58	57
11	88	88	1,226	89	89	88	88
12	61	60	882	60	60	61	60
13	62	62	905	63	62	62	62
14	31	30	759	30	30	30	30
WBSN id = 3							
rt id	P1-TW (ms)	P2-TW (ms)	P3-TW (ms)	P4-TW (ms)	P5-TW (ms)	P6-TW (ms)	P7-TW (ms)
15	56,825	56,816	>256,811	56,815	56,813	56,820	56,811
16	274	273	3,299	274	273	273	273
17	80	80	714	80	81	80	80
18	110	111	1,744	110	110	110	110
19	89	88	1,155	89	88	88	88
20	86	86	1,100	86	86	87	86
21	36	37	995	36	37	36	36
WBSN id = 4							
rt id	P1-TW (ms)	P2-TW (ms)	P3-TW (ms)	P4-TW (ms)	P5-TW (ms)	P6-TW (ms)	P7-TW (ms)
22	105,549	105,545	>305,478	105,554	105,558	105,496	105,563
23	1,721	1,722	10,797	1,721	1,721	1,721	1,722
24	44	45	1,996	44	44	44	45
25	135	134	3,817	134	134	134	135
26	96	97	2,294	96	96	96	97
27	83	83	2,232	83	83	83	84
28	46	46	1,442	46	46	46	47

6.4. Discussion on scalability of the proposal

Scalability is specially affected by cliques management and the construction of *rt*.

Regarding identified features, cliques is the one with bigger impact on the enforcement process complexity. Specifically, the verification of P3 (related to cliques management) takes more than 100,000 ms over a WBSN with 120 relationships per node and more than 200,000 ms over WBSNs with 185 and 219 relationships per user. Nonetheless, as aforementioned, results could be enhanced improving the cliques algorithm verification and executing the enforcement in powerful hardware, as well as applying caching mechanisms, artificial intelligence, etc.

On the other hand, the creation of rt is a hard task. It seems to be satisfactory as long as explored nodes and relationships per user do not exceed 200,000 and 200 respectively, being these values higher than current average ones (36,290 and 190 respectively). However, the enforcement process could scale better if powerful software and hardware techniques are applied.

In sum, it must be noted that in a real-world setting, service providers own lot of resources which, regardless of aforementioned scalability issues, should allow the deployment of the proposed model. By contrast, the study of a large-scale scalability analysis is out of the scope of this proposal and a matter of future work.

7. Conclusions and open research issues

Current WBSN trends highlight the relevance of this proposal. Having identified that expressive power in the WBSN field is associated with the management of a set of six features, a total of 23 proposals have been analysed. This analysis verifies the lack of an expressive model capable of addressing all identified features. This issue together with the appropriateness of managing access control not only until granting access, but also along the whole usage process, leads to the development of $SoNeUCON_{ABC}$. This model extends $UCON_{ABC}$ [1] including relationships management. In particular, $SoNeUCON_{ABC}$ is formally defined, specifying entities and elements involved, as well as an access control policy language. Moreover, policy construction is carefully detailed by using regular expressions and access control enforcement functions are appropriately described. Finally, the evaluation shows, theoretically, that every identified feature is addressed in $SoNeUCON_{ABC}$ and, empirically, that the model's implementation is feasible and applicable to the majority of cases. A proof of concept system has been developed in this regard. Considering 2,000 ms the tolerable waiting time of WBSN users for information retrieval, results show that the enforcement of policies without cliques is satisfactory if explored nodes do not exceed about 200,000 and 200 relationships per node. Besides, the enforcement of policies with cliques remains successful if less than about 30,000 nodes and 200 relationships per node are explored. As a result, the appropriateness of the model for the WBSN context is highlighted.

Concerning open research issues, it is advisable to enhance the developed proof of concept system. Implementing a more efficient cliques evaluation algorithm would be desirable, as well as a large-scale scalability analysis. Furthermore, the identification of a holistic extensible and unified catalogue of

$ATT(S)$, $ATT(O)$ and $ATT(E)$ used in current WBSNs would be an appealing matter. Other open research issue focuses on the development of an administrative model for $SoNeUCON_{ABC}$ that also includes the management of co-ownership (i.e. a photo in which many people appear), a relevant matter in WBSNs. Besides, the analysis of the complexity and the amount of user actions involved in access control policies construction is other matter which is worth studying. Therefore, the search of usability in the policy construction is the following step. Related to this issue, proposed features have been identified from literature and, though researchers seem to be far from reality, studies support the appeal of fine-grained access control systems [62, 63]. However, it still remains as an open research issue the usability of the policy construction based on these features. Research studies may promote the incorporation of more flexible, powerful and fine-grained access control systems in commercial WBSNs, therefore, bringing academic proposals closer to users. Last but not least, though $SoNeUCON_{ABC}$ is a privacy-preserving model, much more work regarding relation privacy is required, that is, the protection of users relationships [64]. Although users identity is unknown, the network structure in terms of attributes can be currently inferred.

Acknowledgement

Authors would like to thank the anonymous reviewers for their helpful comments that have contributed to improve this work.

References

[1] J. Park, Usage Control: A Unified Framework for Next Generation Access Control, Ph.D. thesis, George Mason University (2003).

[2] R. Gross, A. Acquisti, Information revelation and privacy in online social networks, in: Proc. of the 2005 ACM wks. on Privacy in the electronic society, pp. 71–80.

[3] M. Mital, D. Israel, S. Agarwal, Information exchange and information disclosure in social networking web sites: Mediating role of trust, The Learning Organization 17 (6) (2010) 479–490.

[4] P. Consortium, Requirements and concepts for privacy- enhancing access control in social networks and collaborative workspaces, Tech. rep. (2009).

[5] P. Samarati, S. Capitani di Vimercati, Access control: Policies, models, and mechanisms, in: Foundations of Security Analysis and Design, Springer, 2001, pp. 137–196.

[6] S. R. Kruk, S. Grzonkowski, A. Gzella, T. Woroniecki, H. C. Choi, D-FOAF: Distributed identity management with access rights delegation, in: Proc. 1st Asian Semantic Web Conf., Springer, 2006, pp. 140–154.

[7] B. Carminati, E. Ferrari, R. Heatherly, M. Kantarcioglu, B. Thuraisingham, A semantic web based framework for social network access control, in: Proceedings of the 14th ACM symposium on Access control models and technologies, SACMAT '09, 2009, pp. 177–186.

[8] R. Baden, A. Bender, N. Spring, B. Bhattacharjee, D. Starin, Persona: an online social network with user-defined privacy, in: ACM SIGCOMM Computer Communication Review, Vol. 39, ACM, 2009, pp. 135–146.

[9] S. Jahid, P. Mittal, N. Borisov, EASiER: encryption−based access control in social networks with efficient revocation, in: Proc. of the 6th ACM Symposium on Information, Computer and Communications Security, ASIACCS '11, ACM, 2011, pp. 411–415.

[10] J. Park, R. Sandhu, A Position Paper: A Usage Control (UCON) Model for Social Networks Privacy, 2000.

[11] J. Park, R. Sandhu, Y. Cheng, ACON: Activity-Centric Access Control for Social Computing, in: Proc. of the 2011 Sixth Intl. Conf. on Availability, Reliability and Security, ARES '11, IEEE Computer Society, 2011, pp. 242–247.

[12] E. Bertino, B. Catania, E. Ferrari, P. Perlasca, A logical framework for reasoning about access control models, in: Proceedings of the sixth ACM symposium on Access control models and technologies, SACMAT '01, ACM, 2001, pp. 41–52.

[13] S. Ganta, Expressive power of access control models based on propagation of rights, Ph.D. thesis (1996).

[14] M. V. Tripunitara, N. Li, A theory for comparing the expressive power of access control models, J. Comput. Secur. 15 (2) (2007) 231–272.

[15] A. Masoumzadeh, J. Joshi, OSNAC: An Ontology-based Access Control Model for Social Networking Systems, in: Proceedings of the 2010 IEEE Second International Conference on Social Computing, SOCIALCOM '10, IEEE Computer Society, 2010, pp. 751–759.

[16] P. W. L. Fong, Relationship−based access control: protection model and policy language, in: Proceedings of the first ACM conference on Data and application security and privacy, CODASPY '11, ACM, 2011, pp. 191–202.

[17] A. Lazouski, F. Martinelli, P. Mori, Usage control in computer security: A survey, Computer Science Review 4 (2) (2010) 81–99.

[18] F. F.-H. Nah, A study on tolerable waiting time: how long are web users willing to wait?, Behaviour & Information Technology 23 (3) (2004) 153–163.

[19] S. Gürses, D2.1-State of the Art. Security and Privacy for Online Social Networks, `https://www.cosic.esat.kuleuven.be/publications/article-2077.pdf` (2011).

[20] J. Li, Y. Tang, C. Mao, H. Lai, J. Zhu, Role Based Access Control for social network sites, in: 2009 Joint Conferences on Pervasive Computing (JCPC), IEEE, 2009, pp. 389–394.

[21] B. Carminati, E. Ferrari, A. Perego, Rule−Based Access Control for Social Networks, in: Proc. OTM 2006 Workshops (On the Move to Meaningful Internet Systems), Vol. 4278 of LNCS, Springer, 2006, pp. 1734–1744.

[22] C. Munckhof, Content Based Access Control in Social Network Sites, Master's thesis, Eindhoven University of Technology (2011).

[23] T. Abdessalem, I. B. Dhia, A reachability-based access control model for online social networks, in: Databases and Social Networks, DBSocial '11, ACM, 2011, pp. 31–36.

[24] P. W. L. Fong, M. Anwar, Z. Zhao, A privacy preservation model for facebook-style social network systems, in: Proceedings of the 14th European conference on Research in computer security, ESORICS'09, Springer-Verlag, 2009, pp. 303–320.

[25] A. Ahmad, B. Whitworth, Distributed access control for social networks, in: Information Assurance and Security (IAS), 2011 7th International Conference on, 2011, pp. 68 –73.

[26] Y. Cheng, J. Park, R. Sandhu, A User-to-User Relationship-Based Access Control Model for Online Social Networks, in: Data and Applications Security and Privacy XXVI, Vol. 7371 of Lecture Notes in Computer Science, 2012, pp. 8–24.

[27] A. Tapiador, D. Carrera, J. Salvachúa, Tie-RBAC: An application of RBAC to Social Networks, Web 2.0 Security and Privacy.

[28] B. Ali, W. Villegas, M. Maheswaran, A trust based approach for protecting user data in social networks, in: Proceedings of the 2007 conference of the center for advanced studies on Collaborative research, CASCON '07, IBM Corp., 2007, pp. 288–293.

[29] H. Wang, L. Sun, Trust-involved access control in collaborative open social networks, in: Proceedings of the 2010 Fourth International Conference on Network and System Security, NSS '10, 2010, pp. 239–246.

[30] H. Hu, G.-J. Ahn, J. Jorgensen, Multiparty Access Control for Online Social Networks: Model and Mechanisms, IEEE Transactions on Knowledge and Data Engineering 99.

[31] W. Villegas, B. Ali, M. Maheswaran, An Access Control Scheme for Protecting Personal Data, in: Proceedings of the 2008 Sixth Annual Conference on Privacy, Security and Trust, PST '08, 2008, pp. 24–35.

[32] I. B. Dhia, Access control in social networks: a reachability-based approach, in: Proceedings of the 2012 Joint EDBT/ICDT Workshops, ACM, 2012, pp. 227–232.

[33] B. I. Dhia, T. Abdessalem, M. Sozio, Primates: a privacy management system for social networks, in: Proceedings of the 21st ACM international conference on Information and knowledge management, ACM, 2012, pp. 2746–2748.

[34] S. Braghin, V. Iovino, G. Persiano, A. Trombetta, Secure and policy-private resource sharing in an online social network, in: Privacy, security, risk and trust (passat), 2011 IEEE third international conference on and 2011 IEEE third international conference on social computing (SocialCom), IEEE, 2011, pp. 872–875.

[35] M. Alizadeh, S. A. Javadi, M. Amini, R. Jalili, Policy specification and enforcement in online social networks using MKNF+, in: Information Security and Cryptology (ISCISC), 2012 9th International ISC Conference on, IEEE, 2012, pp. 48–53.

[36] G. Bruns, P. W. Fong, I. Siahaan, M. Huth, Relationship-based access control: its expression and enforcement through hybrid logic, in: Proceedings of the second ACM conference on Data and Application Security and Privacy, ACM, 2012, pp. 117–124.

[37] N. Koch, A. Kraus, The expressive power of uml-based web engineering, in: Second International Workshop on Web-oriented Software Technology (IWWOST02), Vol. 16, CYTED, 2002.

[38] E. Dantsin, T. Eiter, G. Gottlob, A. Voronkov, Complexity and expressive power of logic programming, ACM Computing Surveys (CSUR) 33 (3) (2001) 374–425.

[39] B. Bérard, A. Petit, V. Diekert, P. Gastin, Characterization of the expressive power of silent transitions in timed automata, Fundamenta Informaticae 36 (2) (1998) 145–182.

[40] A. Carreras, E. Rodriguez, J. Delgado, X. Maroñas, Access control issues in social networks, in: 11th International Workshop of the Multimedia Metadata Community, 2010, pp. 47–52.

[41] S. Capitani di Vimercati, P. Samarati, S. Jajodia, Policies, models, and languages for access control, in: DNIS, 2005, pp. 225–237.

[42] J. D. Joshi, E. Bertino, A. Ghafoor, An analysis of expressiveness and design issues for the generalized temporal role-based access control model, IEEE Trans. Dependable Secur. Comput. 2 (2) (2005) 157–175.

[43] A. Chander, J. C. Mitchell, D. Dean, A state-transition model of trust management and access control, in: Proceedings of the 14th IEEE workshop on Computer Security Foundations, CSFW '01, IEEE Computer Society, Washington, DC, USA, 2001, pp. 27–43.

[44] L. Habib, M. Jaume, C. Morisset, Formal definition and comparison of access control models, Journal of Information Assurance and Security 4 (2009) 372–378.

[45] B. Carminati, E. Ferrari, Access control and privacy in web-based social networks, in: International Journal of Web Information Systems, Vol. 4, 2008, pp. 395–415.

[46] M. Anwar, P. W. L. Fong, X. Yang, H. Hamilton, Visualizing privacy implications of access control policies in social network systems, in: Proceedings of the 4th international workshop, and Second international conference on Data Privacy Management and Autonomous Spontaneous Security, DPM'09/SETOP'09, Springer-Verlag, 2010, pp. 106–120.

[47] P. W. Fong, I. Siahaan, Relationship-based access control policies and their policy languages, in: Proceedings of the 16th ACM symposium on Access control models and technologies, SACMAT '11, ACM, 2011, pp. 51–60.

[48] W. Villegas, A trust-based access control scheme for social networks, Ph.D. thesis, McGill University (2008).

[49] A. Squicciarini, M. Shehab, F. Paci, Collective privacy management in social networks, in: Proceedings of the 18th international conference on World wide web, WWW '09, 2009, pp. 521–530.

[50] A. Simpson, On the need for user-defined fine-grained access control policies for social networking applications, in: Proceedings of the workshop on Security in Opportunistic and SOCial networks, SOSOC '08, ACM, 2008, pp. 1:1–1:8.

[51] J. Park, R. Sandhu, The UCON$_{ABC}$ usage control model, ACM Transactions on Information and System Security 7 (1) (2004) 128–174.

[52] F. Salim, J. Reid, E. Dawson, An administrative model for UCON$_{ABC}$, in: Proceedings of the Eighth Australasian Conference on Information Security, Vol. 105 of AISC '10, 2010, pp. 32–38.

[53] L. González−Manzano, A. I. González−Tablas, J. M. de Fuentes, Security and Privacy Preserving in Social Networks, Springer, In press 2013, Ch. User-Managed Access Control in Web Based Social Networks.

[54] C. Grompanopoulos, A. Gouglidis, I. Mavridis, A Use-Based Approach for Enhancing UCON, in: Security and Trust Management, Springer, 2013, pp. 81–96.

[55] J. Gross, J. Yellen, Graph theory and its applications, CRC Press, Inc., Boca Raton, FL, USA, 1999.

[56] R. Sandhu, J. Park, Usage control: A vision for next generation access control, Computer Network Security (2003) 17–31.

[57] OWASP-members, Owasp top 10, `http://owasptop10.googlecode.com/files/OWASP\%20Top\%2010\%20-\%202013.pdf` (2013).

[58] R. S. Scowen, Extended bnf-a generic base standard, Tech. rep., Technical report, ISO/IEC 14977. http://www. cl. cam. ac. uk/mgk25/iso-14977. pdf (1998).

[59] NIST, American National Standard for Information TechnologyRole Based Access Control (2003).

[60] L. Zhang, W. Tu, Six degrees of separation in online society, in: Proceedings of the WebSci'09: Society On-Line, 2009, pp. 87 –90.

[61] J. Ugander, B. Karrer, L. Backstrom, C. Marlow, The anatomy of the facebook social graph, arXiv preprint arXiv:1111.4503.

[62] M. L. Mazurek, et al., Access control for home data sharing: Attitudes, needs and practices, in: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '10, ACM, 2010, pp. 645–654.

[63] J. S. Olson, J. Grudin, E. Horvitz, A study of preferences for sharing and privacy, in: CHI '05 Extended Abstracts on Human Factors in Computing Systems, CHI EA '05, ACM, 2005, pp. 1985–1988.

[64] N. Li, N. Zhang, S. K. Das, Preserving relation privacy in online social network data, Internet Computing, IEEE 15 (3) (2011) 35–42.

[65] G. Harman, Reasoning, Meaning, and Mind, Oxford University Press, 1999.

[66] S. Capitani di Vimercati, S. Foresti, P. Samarati, Authorization and Access Control, Security, Privacy, and Trust in Modern Data Management (2007) 39–53.

[67] X. Zhang, J. Park, F. Parisi-Presicce, R. Sandhu, A logical specification for usage control, in: Proceedings of the ninth ACM symposium on Access control models and technologies, SACMAT '04, ACM, 2004, pp. 1–10.

Appendix A. Inductive reasoning: generalizing access control policies

In order to determine the expressive power of ACMs, it is not enough analysing that a concrete policy, related to a feature, can be expressed by a specific model and not by another one. The chief question that comes up is, is it possible to assert that any model that expresses policies associated with a feature is able to define any other policy related to this feature? Inductive reasoning is applied [65] to address this issue and reach a general rule after having reasoned from a set of cases. Applying inductive reasoning it is established that if a particular policy (P), based on a concrete feature (F), can be expressed by a specific model (M), then it is possible to generalize that M can express any kind of P based on F. In other words, using this technique it is generalized the number of P that can be created by a particular M. Specifically, inductive reasoning is applied to *distance* (F1), *common-contacts* (F2), *clique* (F3) and *multi-path* (F4). On the contrary, it cannot be applied to *direction* (F5) and it would be extremely tedious in respect to *fine-grained* (F6) due to several reasons. Regarding F5, this feature exclusively requires the creation of directional and bidirectional relationships and consequently, generalization does not have to be applied. By contrast, according to F6, the amount of attributes that can be managed is extremely assorted and their generalization is unattainable.

Table A.7 presents the application of inductive reasoning to F1, F2, F3 and F4. For each feature a set of cases CZ (being Z $\in \aleph$) are established to reach a general rule (G), considering that both, CZ and G, describe particular and general access control policies respectively. An access control policy is structured following the pattern $X \Rightarrow Y$, where X refers to the set of conditions to satisfy in order that Y happens, that is, certain rights granted to the requester by the administrator. Moreover, it is noticeable that cases are incremental - the higher cases involve the previous ones. In other words, given C1, C2 and C3, expressing C3 means that C2 and C1 are satisfactorily expressed too because they are involved in C3. Then, it is expected, given the general nature of ACMs, that ACMs that express C3 are expressive enough to define any policy from G.

Regarding F1, C1 refers to a policy in which access is granted to users (D) who are at a distance of three hops from the administrator (A). Similarly, C2 presents a policy which grants access to users (E) who are at a distance of four hops from the administrator (A). Then, following this reasoning, the specification of C1 and C2 can be generalized as an access control policy that involves users connected by a number n of hops, $n \geqslant 3$. Then, P1 is proposed considering $n = 3$.

In respect to F2, C1 corresponds to a policy in which access is granted to users (C) who have a contact (B) in common with the administrator (A). Similarly, C2 corresponds to a policy which grants access to users (B) who have a pair of contacts (B and D) in common with the administrator (A). Following an analogous reasoning, G shows that access is granted to users who have a number n of common contacts with the administrator. Thus, P2 is proposed considering $n = 3$.

On the other hand, according to F3, C1 presents a policy that grants access to users that belong to a clique composed of three users (A, B and C, where A is the administrator and B or C the requester). Analogously, C2 corresponds to a policy that grants access to four users involved in a clique (A, B, C and D, where A is the administrator and B, C or D the requester). Therefore, G refers to a policy that grants access to users who belong to a clique composed of $n+1$ users, being $n \geqslant 2$. Specifically, it refers to the establishment of as many bidirectional relationships as existing edges in a complete graph composed of n nodes and, consequently, the number of created bidirectional relationships are $\frac{n\times(n-1)}{2}$. Then, P3 is proposed assuming $n=2$.

Finally, in relation to F4, C1 presents a policy that grants access to users (B) with whom the administrator (A) is directly or indirectly connected by a pair of different paths. Likewise, C2 refers to a policy that grants access to users (B) that are connected to the administrator (A) by three different paths. Following such reasoning, G presents a policy that grants access to users connected to the administrator by a number n of different paths. Therefore, P4 is proposed assuming $n=2$.

Appendix B. Expressive power analysis of WBSN ACMs

In order to evaluate the expressive power of WBSNs policy languages, a total of 23 proposals have been selected and classified under RBAC, RelBAC, ABAC, TBAC and OBAC models.

In the following Sections the chosen approaches are briefly introduced and access control policies proposed in Section 3.2 are, as far as possible, expressed by each of them. In some cases the definition of policies includes the creation of additional elements, such as predicates, attributes or relationship types, which are created according to each proposal specifications. Besides, some policies cannot be defined because the model has not got the appropriate level of expressive power. In particular, partially defined policies are marked with "( )" and unexpressed policies are not even mentioned.

Appendix B.1. Role based access control (RBAC)

Three proposals have been classified under RBAC, being remarkable the lack of expressive power of all of them.

Role based access control for Social Networks [20]. This approach bases on relationship management. Relationships are represented as the connection between a pair of users taking each of them one specific role.

A WBSN involves the management of multiple elements, for instance user attributes. Nonetheless, this model exclusively includes relationship management offering quite restrictive procedures. Relationship types are the only relationship property that the model comprises.

Before defining access control policies, a set of roles and permissions have to be specified:

Table A.7: Inductive reasoning

Distance (F1) $(C1)A \xrightarrow{D_a} B \ \wedge \ B \xrightarrow{D_b} C \ \wedge C \xrightarrow{D_c} D \ \Rightarrow \ D \longrightarrow DAT(A)$ $(C2)A \xrightarrow{D_a} B \ \wedge \ B \xrightarrow{D_b} C \ \wedge C \xrightarrow{D_c} D \ \wedge D \xrightarrow{D_c} E \ \Rightarrow \ E \longrightarrow DAT(A)$ ——————————- $(G)X \xrightarrow{D_x} Y_1 \ \wedge \ Y_1 \xrightarrow{D_{y1}} Y_2 \ \wedge \ ... \ \Rightarrow \ Y_{n+1} \longrightarrow DAT(X)$ Given that this feature can be generalized for $n \geqslant 3$ where n corresponds to the number of hops, P1 is proposed assuming $n = 3$.
Common contacts (F2) $(C1)A \xrightarrow{D_{a_1}} B \ \wedge \ C \xrightarrow{D_{a_1}} B \ \Rightarrow \ C \longrightarrow DAT(A)$ $(C2)A \xrightarrow{D_{a_1}} B \ \wedge \ C \xrightarrow{D_{a_1}} B \ \wedge \ A \xrightarrow{D_{a_1}} D \ \wedge \ C \xrightarrow{D_{a_1}} D \ \Rightarrow \ C \longrightarrow DAT(A)$ $(C3)A \xrightarrow{D_{a_1}} B \ \wedge \ C \xrightarrow{D_{a_1}} B \ \wedge \ A \xrightarrow{D_{a_1}} D \ \wedge \ C \xrightarrow{D_{a_1}} D \ \wedge \ A \xrightarrow{D_{a_1}} E \ \wedge \ B \xrightarrow{D_{a_1}} E \ \Rightarrow \ C \longrightarrow DAT(A)$ ——————————- $X \xrightarrow{D_{x1}} Y_1 \ \wedge \ ... \wedge \ X \xrightarrow{D_{xn}} Y_n \ \wedge \ Z \xrightarrow{D_{x1}} Y_1 \ \wedge \ ... \wedge \ Z \xrightarrow{D_{xn}} Y_n \ \Rightarrow \ Z \longrightarrow DAT(X)$ Given that this feature can be generalized for $n \geqslant 1$ where n corresponds to the number of common contacts, P2 is proposed assuming $n = 3$.
Clique (F3) $(C1)A \longleftrightarrow B \ \wedge \ A \longleftrightarrow C \ \wedge \ B \longleftrightarrow C \ \Rightarrow \ A \longrightarrow DAT(B,\ C) \ \wedge \ B \longrightarrow DAT(A,\ C) \ \wedge \ C \longrightarrow DAT(A,\ B)$ $(C2)A \longleftrightarrow B \ \wedge \ A \longleftrightarrow C \ \wedge \ A \longleftrightarrow D \ \wedge \ B \longleftrightarrow D \ \wedge \ B \longleftrightarrow C \ \wedge \ C \longleftrightarrow D \ \Rightarrow \ A \longrightarrow DAT(B,\ C,\ D)$ $\wedge \ B \longrightarrow DAT(A,\ C,\ D) \ \wedge \ C \longrightarrow DAT(A,\ B,\ D) \ \wedge \ D \longrightarrow DAT(A,\ B,\ C)$ —————————— $(G)X \longleftrightarrow Y_1... \ \wedge \ X \longleftrightarrow Y_n \ \wedge \ Y_1 \longleftrightarrow Y_2... \ \wedge \ Y_1 \longleftrightarrow Y_n \ \wedge \ ... \ \wedge \ Y_{n-1} \longleftrightarrow Y_n \ \Rightarrow \ X \longrightarrow DAT(Y_1, ..., Y_n)$ $\wedge \ Y_1 \longrightarrow DAT(X, Y_2, ..., Y_n)... \ \wedge \ Y_n \longrightarrow DAT(X, Y_1, ..., Y_{n-1})$ Given that this feature can be generalized for $n \geqslant 2$ where $n + 1$ corresponds to the number of contacts in the clique, P3 is proposed assuming $n = 2$.
Multi-path (F4) $(C1)A \xrightarrow{D/I_{b1}} B \ \wedge \ A \xrightarrow{D/I_{b2}} B \ \wedge \ b1 \neq b2 \ \Rightarrow \ B \longrightarrow DAT(A)$ $(C2)A \xrightarrow{D/I_{b1}} B \ \wedge \ A \xrightarrow{D/I_{b2}} B \ \wedge \ A \xrightarrow{D/I_{b3}} B \ \wedge \ b1 \neq b2 \ \wedge \ b1 \neq b3 \ \wedge \ b2 \neq b3 \ \Rightarrow \ B \longrightarrow DAT(A)$ —————————— $(G)X \xrightarrow{D/I_{y1}} Y \ \wedge \ ...X \xrightarrow{D/I_{yn}} Y \ \wedge \ y1 \neq ....yn \ \Rightarrow \ Y \longrightarrow DAT(X)$ Given that this feature can be generalized for $n \geqslant 2$ where n corresponds to the number of different paths that connect the administrator and the requester, P4 is proposed assuming $n = 2$.
Applied predicates – X is direct or indirectly connected with Y by the relationship z: X $\xrightarrow{D/I_z}$ Y – X accesses to data, DAT, of contact Y: X $\longrightarrow$ DAT(Y) – X accesses to data, DAT, of contacts {Y$_1$, Y$_2$,...,Y$_n$}: X $\longrightarrow$ DAT(Y$_1$, Y$_2$,...,Y$_n$) – X and Y are bidirectional contacts: X $\longleftrightarrow$ Y

- *Roles* R={friend}
- *Permissions* P={read}
- $R \times P \ \Rightarrow$ the defined permission is assigned to all roles

According to [20], access control policies are identified as social relations ($SR(s)$) that a particular user (s) owns. They consist of three elements, {User1, User2, <User1's role, User2's role>}, where User1 and User2 corresponds to the couple of users involved in the relationship. Assuming that a is the administrator and s the requester, the proposed access control policies are defined as follows:

P5 $SR(a) = \{a,\ s,\ < friend,\ friend >\}$
$SR(s) = \{s,\ a,\ < friend,\ friend >\}$

P6 $SR(a) = \{a,\ s,\ < friend,\ friend >\}$

Due to the simplicity of the model just a couple of policies can be specified. Regarding P5, it is satisfactorily defined through a pair of social relations. It is assumed, but not explicitly mentioned, that an access control policy canf be composed of more that a single SR. Analogously, P6 is properly defined.

Tie-RBAC [27]. A relation is defined as a set of ties of the same type between senders and receivers. Each relation involves the sender's assignation of the receiver to a role with permissions. Besides, ties are non-reciprocal, that is, they are unidirectional and having a tie with a particular user does not imply the existence of a tie on the other way round.

The specification of policies is quite limited. Concerning described examples, just a pair of policies can be created. The administrator is a, s is the requester and the tie used is "Friend" which has the read permission linked to an object:

P5 $Tie_1\ =\ Friend\ from\ a\ to\ s$ and $Tie_2\ =\ Friend\ from\ s\ to\ a$

P6 $Tie_1\ =\ Friend\ from\ a\ to\ s$

Therefore, P5 and P6 are satisfactorily expressed.

Distributed access control [25]. Looking for the decentralization of access control in WBSNs, this model bases on sharing resources regarding relationship type or closeness. Besides, roles (called Local Roles, LR), permissions (called Attestation Certificates, AC) and objects that belongs to users (called Namespace, NS) are managed.

Regarding the WBSN definition, this model manages users, relationships and objects but not in a fine-grained way because just a single policy, P6, can be specified.

The following elements are required:

- LR={friend}
- AC={r is granted to all roles in LR}
- NS={o and other data the administrator has}

In respect to previous elements, considering i a particular domain (e.g. a particular set of photos), τ a set of security labels attached to objects and OC an object group (e.g. privacy labels), access control policies, called conditions (con), are the following:

P6 $con(s,o,i)\ =\ isactive(s,\ NS)\ \wedge\ hasLR(s,\ i)\ \wedge\ hasOC(o,\ i,\ \tau)\ \wedge\ hasAC(AC,\ LR)$

Considering that the aim of this model is the development of distributed access control, P6 is the only policy expressed by it. However, it is successfully defined.

Appendix B.2. Trust based access control (TBAC)

This category involves a total of five proposals which are far from being the most expressive ones.

Social access control (SAC) [28]. SAC bases access control management on exploiting trust relationships between users. Policies are established regarding the trust placed in users (τ) and the confidentiality attached to each object ($t_c(o)$), such that if $\tau > t_c(o)$ the access is granted and denied otherwise.

SAC allows the definition of the following policy:

(P4) $\tau = high$

In this model trust is placed in users instead of relationships. Nonetheless, it is assumed that if a user gets access to an object with a certain kind of trust attached, it can be compared with the establishment of a trust relationship with this user. As a result, P4 is defined to some extent. In addition, relationships are inherently bidirectional and then, P5 is implicitly defined.

Rule based access control [21]. Carminati *et al.* propose a rule-based access control model to allow users the specification of access rules for their contents. Policies are expressed as constraints on the type, depth, and trust level of existing relationships. Moreover, certificates are applied to attest the authenticity of authenticity.

This model is specially focused on the relationship types, trust and depth management. Besides, relationships are unidirectional and, even considering a bidirectional relationship as a pair of unidirectional ones, provided specifications do not detail the management of bidirectional relationships in access control policies.

In order to define proposed access control policies, the following relationships types are required:

- *Relationship Types*={FriendOf, RelativeOf}

The last pair of types has been created to meet the goals of the analysis presented herein. Types are non-fixed and can be deliberately defined. According to this model, access control rules are composed of the *oid* of the requester resource identification and a set of predicates composed of four relationship elements {Node, Relationship Type, Jumps, Trust level} where "Node" refers to the administrator (a), that is, the owner of the requested data.

Policies are defined as follows:

(P1) $(oid,\ \{(a, RelativeOf, 3, *)\})$

(P4) $(oid,\ \{(a, FriendOf, 1, 10)\})$

P6 $(oid,\ \{(a, FriendOf, 1, *)\})$

Only three policies are expressed and just P6 is satisfactorily defined. On the one hand, regarding P1, the indirect relationship is not completely expressed. An indirect relationship composed of three hops can be defined but neither different roles in each hop nor particular relationship preferences can be pointed out. On the other hand, P4 expresses the fact that the relationship is completely trusted (considering a level of trust from 0 to 10) but it does not specify multiple paths.

Reachability-based access control model [23]. This model expresses access control rules as reachability constraints which encode the path between the requester and the administrator of the requested object. More specifically, similar to [21], this model bases on the specification of access control policies regarding the trust and distance of WBSN users.

Regarding elements managed in a WBSN, this model focuses on relationships and users. Nonetheless, relationships management is quite limited, trust and distance are the only managed attributes and there are not tools to deal with features such as common friends, cliques or multiple paths (P2, P3 and P4 respectively).

In this work, the following relationships and user properties, all of them described in [23], are applied:

- *Relationship*={Friend}
- *User properties*={age, gender, studies}

Specifically, access control policies consist of a tuple of three elements, {Requested object, Relationships path, Trust threshold}. Furthermore, "Relationships path" consists, at the same time, of four elements, {Starting user, Relationship ('+' refers to outgoing relationships and '-' to incoming), Relationship depth, User properties}. Also, "Trust threshold" is 0 when it is not considered in a particular rule.

Policies are constructed as follows, considering that a is the administrator:

(P1) $(o,\ \{(a,\ Friend^{+}[1,2,3])\},\ 0)$

(P4) $(o,\ \{(a,\ Friend^{+}[1])\},\ 10)$

(P5) $(o,\ \{(a,\ Friend^{+}[1]),\ (o, Friend^{-}[1])\},\ 0)$

P6 $(o,\ \{(a,\ Friend^{+}[1])\},\ 0)$

(P7) $(o,\ \{(a,\ Friend^{+}[1][age = 30][gender = female])\},\ 0)$
$(o,\ \{(a,\ Friend^{+}[1][age = 40][gender = female][studies = c.science])\},\ 0)$
$(o,\ \{(a,\ Friend^{+}[1][studies = c.science])\},\ 0)$

A great set of policies can be expressed using this model, though not reaching completely successful results. Indeed, P6 is the only policy properly specified. By contrast, P1 is defined to some extent. It grants access to users, located at three hops without pointing out relationship preferences. Similarly, P4 specifies the existence of a highly trusted relationship but without specifying multiple paths. Likewise, differing from the proposed P5, the created policy is simultaneously satisfied by unidirectional and bidirectional relationships. Similarly, P7 is not satisfactorily specified and it requires the definition of a set of three policies. In particular, the only operator applied in the specification of attributes values is "=" and thus, it is unattainable the definition of granting access to users under a certain age. Analogously, according to the model specifications, attributes are uni-valued and it cannot be defined granting access to users who have multiple degrees.

Personal Data Access Control (PDAC) [31]. This model manages access computing the trusted distance (d_{trust}) between users. A particular trust is linked to each user who is located at a certain distance in the social network graph. Subsequently, once a data is requested, d_{trust} is calculated regarding the trust and distance of the requester.

Specifically, assuming that a is the administrator and o the requested object, policies can be identified as the establishment of a trust interval such that (*accept limit*, *reject limit*). The *accept limit* ($C_a(a,d)$) refers to the largest trusted distance and the *reject limit* corresponds to the smallest trusted distance ($C_r(a,d)$). As a result, access is granted if the calculated d_{trust} is within the established interval. Moreover, it is considered that trust distance for a *friend* is 1, for a *friend-of-friend* is 2 and for *friend-of-friend-of-friend* is 3, as well as intermediate trust levels are managed and, for instance, 0.8 refers to a very good friend and 0.9 to a good friend.

Concerning above specifications proposed access control policies are defined as follows:

(P1) (3,3)

(P4) (0,0.5)

P6 (0,1)

PDAC achieves the complete definition of P6, establishing that access is granted if requesters are friends of any kind of trust between 0 and 1, that is, if they are good, very good, extremely good friends, etc. Nonetheless, P1 and P4 are slightly expressed. P1 defines that only those users who are located at a distance three from the administrator get access, leaving aside the specification of relationship types and the relationship creation time. Likewise, P4 specifies granting access to users who are highly trusted but the definition of multiple paths is not achieved.

Trust in Collaborative Open Social Networks [29]. Wang *et al.* present a fine-grained access control scheme for WBSNs focused on managing access control through a purpose-based approach. Data is related to a set of purposes that form a hierarchy and can change dynamically.

Access control policies are constructed as rules composed of seven elements: *Data* which identifies the requested data; *Sub* that refers to requesters to whom the access is granted; *RelT* which corresponds to the type of the relationship between the requester and the data owner; *Purp* that corresponds to the right that users request; *Dmax* which corresponds to the maximal relationship depth; *Tmin* that refers to the minimal trust; and *Obli* that refers to requirements that have to be satisfied before granting access. As a result, a policy rule is defined as: (*Data, Sub, RelT, Purp, Dmax, Tmin, Obli*).

Considering that o is the requested object, r is the requester, relationship types are $friend$, *relative* and *neighbour*, *read* is the purpose, trust rates from 0 to 10 and there are not obligations ($\emptyset$), the definition of proposed policies is the following:

(P1) $(o,\ s,\ relative,\ read,\ 3,\ 5,\ \emptyset)$

(P4) $(o,\ s,\ friend,\ read,\ 1,\ 10,\ \emptyset)$

P6 $(o,\ s,\ friend,\ read,\ 1,\ 5,\ \emptyset)$

This approach allows the definition of three policies but just P6 completely. P1 is slightly defined as neither roles at each distance can be different, nor the specification of relationship attributes is possible. Similarly, P4 is partially defined because it only allows the establishment of high trust relationships but not multiple paths.

Appendix B.3. Relationship based access control (RelBAC)

Relationships are an inherent element of WBSNs, being identified a total of six proposals within this category.

Privacy preservation model [24]. This model bases on the generalization of Facebook access control mechanism. It demonstrates that the expression of several policies not currently supported by Facebook can be carried out, such as the sharing of data between common friends or friends involved in a clique.

According to this ACM and the WBSN definition, users and relationships are the main managed elements. Similar to previous proposals, it does not deal with attributes management and policies such as P7 cannot be defined. Furthermore, as it is based on Facebook, the establishment of unidirectional relationships like the one proposed in P6 is unreachable.

Assuming that u and v refer to a pair of WBSN users, s is the requester, G is the complete social network represented as a graph and γ refers to a particular communication state that describes the communication history between the administrator (a) and s access control policies are defined as follows:

(P1) $\textit{only-me} = \gamma(u,\ a,\ G,\ \gamma).u = a$
$\textit{only-friends} = \textit{only-me}\ \vee\ (\gamma(u,\ v,\ G,\ \gamma).\{u,\ v\} \in E(G))$
$\textit{friends-of-friends} = \textit{only-friends} \vee (\gamma(v,\ s,\ G,\ \gamma).(\exists v' \in Sub\{s,\ v'\} \in E(G)\ \wedge\ \{v',\ v\} \in E(G)))$

P2 $common-friends_3 = \textit{only-friends} \vee (\gamma(s,\ a,\ G,\ \gamma).|N_G(s) \cap N_G(a)| \geqslant 3)$

P3 $clique_3 = only\text{-}friends \vee (\gamma(s, a, G, \gamma).(\exists G'.G' \subseteq G \wedge G' \cong K_3 \wedge \{s, a\} \subseteq V(G')))$

(P4) $path_2 = friends\text{-}of\text{-}friends \wedge friends\text{-}of\text{-}friends = (only\text{-}friends = only\text{-}me \vee (\gamma(u, v, G, \gamma).\{u, v\} \in E(G))) \wedge (only\text{-}friends = only\text{-}me \vee (\gamma(z, x, G, \gamma).\{z, x\} \in E(G)))$

P5 $only\text{-}friends = only\text{-}me \vee (\gamma(u, v, G, \gamma).\{u, v\} \in E(G))$

Due to the similarity with Facebook, results are quite interesting. Regarding P1, it is slightly expressed because indirect relationships have a maximum depth of two and relationships fine-grained management is not considered either. An interesting policy is P4 which, considering the possibilities offered by this model, is defined through the combination of existing access control policies. Then, even not currently supported by Facebook, the establishment of policies that involve multiple paths is supported. However, contrary to the proposed P4, the created one requires the specification of each path length and also, it is missing the definition of being different paths. By contrast, P2, P3 and P5 are satisfactorily defined.

Relation based access control [16, 47]. This model focuses on capturing the idea that an authorization decision, a policy, bases exclusively on the relationship between the administrator and the requester. The main issue runs towards the specification of policies capable of expressing WBSN features such as having common friends or the establishment of cliques.

This ACM, as its name suggests, focuses on relationship without considering attributes management, either relationships, objects or users attributes. Then, P7, which is particularly focused on user attributes, cannot be defined.

Regarding proposed policies, the following set of relationships identifiers are applied:

- I={friend, neighbour, relative}

Considering that the administrator (a) is identified by the prepositional symbol $@p$ [47] and v and u refer to WBSN users and s refers to the requester, access control policies are defined as follows:

(P1) $(@p.\langle relative\rangle(\neg u \wedge \neg v \wedge \langle neighbour\rangle v)(\neg u \wedge \neg v \wedge \neg s \wedge \langle friend\rangle s))$

P2 $s \vee \langle friend\rangle s \vee (((\langle friend\rangle\langle friend\rangle\langle friend\rangle s) \oplus (\langle friend\rangle\langle friend\rangle\langle friend\rangle s))$

P3 $s \vee (\neg s \wedge \langle friend\rangle s \wedge @p.\langle friend\rangle(\neg p \wedge \neg s \wedge \langle friend\rangle s))$

(P4) $(\langle friend\rangle(\neg s \wedge \langle neighbour\rangle s)) \wedge (\langle friend\rangle(\neg v \wedge \langle neighbour\rangle v))$

P5 $\langle friend\rangle s \wedge \langle -friend\rangle s$

P6 $\langle friend\rangle s$

This model is one of the most expressive models attaining the successful definition of four policies, P2, P3, P5 and P6. However, in respect to P1, given that this model is not focused on attributes management, the specification of the proposed relationship creation time is infeasible. On the other hand, according to P4, even being possible the definition of multiple paths, the path length has to be established. Indeed, the created P4 specifies the existence of a pair of different paths of two hops.

Relation based access control through hybrid logic [36]. This proposal presents a RelBAC model that uses hybrid logic to express access control policies. Indeed, as aforementioned, this is one of the first contributions which particularly mentioned and work to achieve expressive power. The model is similar to the one proposed in [47] but applying other type of logic.

In general, binary relationships are managed, tagged with a set of labels in I, being S the set of principals, subjects, involved in them. Moreover, it is distinguished *own* to refer to the owner, the administrator, and *req* to refer to the requester and the symbol @ is used to define a policy in regard to both principals. Specifically, labels applied herein are the following:

- I={friend, neighbour, relative}

As a result, proposed policies are defined as follows:

(P1) $@_{own}\langle relative\rangle((req \wedge \langle neighbour\rangle req) \wedge req \wedge \langle friend\rangle req$

P2 $@_{own}(req \vee \langle friend\rangle req \wedge \langle friend\rangle_3 \langle friend\rangle req$

(P4) $@_{own}(\langle friend\rangle req \wedge \langle neighbour\rangle req$

P5 $@_{own}\langle friend\rangle req \wedge @_{req}\langle friend\rangle own$

P6 $@_{own}\langle friend\rangle req$

It is identified the possibilities offered by hybrid logic. Nevertheless, the difficult task of expressing cliques is pointed out [36], as well as it is just briefly mentioned the management of user attributes. These attributes are applied in terms of types (called labels). They are quite similar to relationships attributes and thus, P1 is partially expressed. Similarly, P4 defines a pair of different paths but their length has to be pre-defined. On the contrary, P2, P5 and P6 are successfully defined.

User relationship-based access control (UURAC) model [26]. UURAC is an online social network model focused on existing WBSN relationships and the establishment of policies through regular expressions. Its main challenge goes towards the definition of policies that involve direct and indirect relationships with different types in each hop.

This novel model specially bases on direct and indirect relationships management, leaving aside attributes highlighted in the proposed WBSN definition. Consequently, P1, P5 and P6 are the policies that can be specified by UURAC.

Before defining policies, a pair of elements have to be noticed, $\sum$ corresponds to the set of managed relationship types and *action* refers to the set of actions that can be requested:

- $\sum$=$\{f,\ n,\ r\}$ where f corresponds to friend, n to neighbour and r to relative.
- *action*=$\{rd\}$ where rd refers to the read permission.

This model proposes a non-fixed set of $\sum$ and consequently, n and r have been created. Furthermore, according to policies in UURAC, there are different types of them. In this regard, as the analysis performed herein focuses on policies specified by an administrator in regard to a resource (o), policies called ***target resources policies*** are the ones applied. These policies consist of three elements $<$***action***, ***resource***, (***starting node***, ***path rule***)$>$, where ***path rule*** corresponds to a set of predicates connected by disjunctions and conjunctions. Besides, each predicate is composed of (***relationship path***, ***hopcount***) where ***hopcount*** refers to the maximum number of edges on the relationship path.

Proposed policies are defined as follows:

(P1) $(r, o, (r*, 1)\ \wedge\ (n*, 2)\ \wedge\ (f*, 3))$

P5 $(r, o, (f*, 1))$

P6 $(r, o, (f*, 1))$

Alluding to this model's name, relationships are the main managed elements. On the one hand, P1 is quite defined, being remarkable a significant issue. The lack of attributes management prevents from detailing the proposed relationship creation time. On the other hand, P5 and P6 are properly defined. Surprisingly, the syntax of both policies is analogous because UURAC explicitly mentions that relationships are unidirectional and their bidirectional nature exists simultaneously.

Multiparty Access Control (MPAC) for Online Social Networks [30]. MPAC focuses on capturing multiparty authorization requirements. It makes possible the collaborative management of shared data in WBSNs.

Access control policies are composed of five elements: *controller*, who is the user who manages access control; *ctype*, that refers to the type of the controller; *accessor*, who is the user to whom the access is granted and it may consist of the user name, the relationship type and the group name; *data*, which corresponds to the identifier of the requested data and the level of data sensitivity; and *effect* refers to the permission granted, that is, permit or deny. Therefore, assuming that the administrator a permits access to an object o with a sensitivity level sl, proposed policies are defined as follows:

(P1) $< a,\ OW,\ \{< friend - of - friend,\ RN >\}, < o,\ sl >,\ permit >$

P5 $< a,\ OW,\ \{< friend - of,\ RN >\}, < o,\ sl >,\ permit >$

Applying this model just a couple of policies can be defined, being P5 the only one completely specified. Also, it should be noticed that relationships have a maximum length of two and they are considered inherently bidirectional.

A reachability-Based Approach [32]. This novel ACM bases on connection characteristics between WBSN users. It tries to generalize access control policies in terms of users properties, indirect relationships and complex relationship composed of direct relationships of different types.

Access control policies, called access rules, consist of the tuple (rid, ACS) where rid is the identifier of the requested resource and ACS refers to the set of access conditions (ac) to satisfy. Besides, each ac is composed of (o, p) where o is the starting node, that is, the resource administrator, and p refers to a path of ordered steps. Each step is also composed of four elements (r, dir, I, C) where r is the type of the relationships, dir is the orientation of the relationship edge ($+, -$ or $*$ in case of bidirectionality), I is the set of authorized distances and C the set of conditions regarding user properties. Then, it is assumed the existence of elements:

- Relationship types= {Relative, Neighbour, Friend}
- User properties= {age, gender, trust}

As a result, assuming that the administrator is a and the requested resource is ro, proposed access control policies are defined as follows:

(P1) $(ro, (a, (Relative,^+, 1), (Neighbour,^+, 1), (Friend,^+, 1))))$

(P4) $(ro, (a, (Friend,^+, 1, (trust = 1))))$

P5 $(ro, (a, (Friend,^*, 1)))$

P6 $(ro, (a, (Friend,^+, 1)))$

(P7) $(ro, (a, (Friend,^+, 1, (gender = female, age < 30))))$

This ACM is significantly expressive as it allows the partial specification of five policies. P1 is slightly defined since the relationship creation time is not specified. Similarly, P4 is defined to some extent. It is specified a trust relationship but not the existence of multiple paths. Likewise, even being possible the definition of user attributes, disjunctions cannot be specified in P7. By contrast, unidirectional and bidirectional relationships are appropriately expressed.

Primates [33]. The ACM proposed in this approach is quite similar to the one presented in [32]. Access control is managed through reachability constraints based on paths between WBSNs users and user properties.

Concerning policies, called access rules, they consist of four elements such that (u, r, P, C) where u is the resource owner, r the requester resource, P the path and C the set of constraints on the attributes of the requester. Besides P consist of constraints on the path that connects the resource owner and the requester and each constraint is, simultaneously, composed of the tuple (l, dir, I) where l is the type of the relationships, dir the direction of the relationships ($\rightarrow$, $\leftarrow$ or $\Leftrightarrow$) and I the minimum and maximum depth of the path. Therefore, assuming the same relationship types and user attributes as those defined in *A reachability-Based Approach* [32] and considering a the administrator and o the requested object, proposed policies are the following:

(P1) $(a, o, (`Relative', \rightarrow, (1,1)), (`Neighbour', \rightarrow, (1,1)), (`Friend', \rightarrow, (1,1)), -)$

(P4) $(a, o, (`Friend', \rightarrow, (1,1)), [trust = 1])$

P5 $(a, o, (`Friend', \Leftrightarrow, (1,1)), -)$

P6 $(a, o, (`Friend', \rightarrow, (1,1)), -)$

(P7) $(a, o, (`Friend', \rightarrow, (1,1)), [gender = female, age < 30])$

Due to the similarity with [32], drawn conclusions are equivalent. The definition of P5 and P6 is complete and the definition of P1, P4 and P7 is partial.

Appendix B.4. Attribute based access control (ABAC)

In this category five proposals are involved. Given that they focus on attributes management, policies that involve F6 (*flexible attributes*) can be defined, at least partially, by most of them.

$UCON_{ABC}$ for social networks [10]. $UCON_{ABC}$ for social networks bases on $UCON_{ABC}$ [1, 52] usage control model. It is developed under the perspective of Attribute Based Access Control (ABAC) models [66]. The definition of policies bases on the application of subjects, resources and the environment. Besides, this model can be used to model MAC, DAC and RBAC access control policies, as well as certain authorization processes of Digital Rights Management (DRM).

In regard to the proposed WBSN definition, the main managed elements are subjects, objects, context and subject and objects attributes (ATT(S) and ATT(O)). Therefore, P3 and P4 cannot be defined because they base on complex relationships management which is not included in this model. Furthermore, it is noticeable that WBSNs access control policies are defined by data owners, referred as administrators. Then, in this model, according to [52], it is assumed that the administrator, who is considered a subject, is the user who administrates requested objects and thus, he manages ATT(O).

Specifically, access control focuses on the specification and management of predicates that express relationships between subjects and objects [67, 52]. According to proposed access control policies, the following ATT(S), ATT(O) and predicates are defined. Notice that except for the predicates *permit* and *in*, which are presented in [67], the rest of predicates and attributes have been created herein following the model specifications.

Attributes:

- $ATT(S)$={Age, Gender, Studies, Friends, Neighbours, Relatives} where Friends, Neighbours and Relatives are the lists of friends, neighbours and relatives, respectively, that the user has. Moreover, each list is composed of a set of attributes:
  - Friends=$\{\{User1_{Id}, relationshipTrust, ...\}$
  - Neighbours=$\{\{User1_{Id}, relationshipTrust, ...\}$
  - Relatives=$\{\{User1_{Id}, relationshipTrust, ...\}$
- $ATT(O)$=$\{Object1_{Id}, ...\}$

Predicates:

- permit($s \in S,\ o \in O,\ r$): it grants access permission (r) over an object (o) to a particular subject (s).
- in($s \in S,\ Friends/Neighbours/Relatives\ of\ v \in S$): it returns the existence of not of a friendship/neighbour/relative relationship between a pair of users (s and v). Notice that s has to be within the list of friends/neighbours/relatives of v.
- commonFriends($Friends\ of\ s \in S,\ Friends\ of\ v \in S,\ n$): it returns a positive or negative value regarding if the list of friends of a subject (s) has n subjects in common with the list of friends of another subject (v), being $n \in \aleph$.

Policies have been constructed following the process described in [67]. Moreover, the administrator of the requested object o is referred to as a and s corresponds to the requester. Given that in this work policies are inherently unidirectional, access control policies established by a are described below:

(P1) $in(a, s.Neighbours) \wedge a.Neighbours[s].creationTime < 2001 \longrightarrow permit(s, a.o, r)$

P2 $commonFriends(a.Friends, s.Friends,\ 3) \longrightarrow permit(s, a.o, r)$

P5 $in(a, s.Friends) \wedge\ in(s, a.Friends) \longrightarrow permit(s, a.o, r)$

P6 $in(s, a.Friends) \longrightarrow permit(s, a.o, r)$

P7 $(s.gender = female\ \wedge\ s.age < 30\ \vee\ (s.gender = female\ \wedge\ s.age < 40 \wedge s.studies = \{C.Science\}) \vee (s.studies = \{C.Science\} \wedge s.studies = \{Physics\}) \longrightarrow permit(s, a.o, r)$

A great set of access control policies are satisfactorily defined. Conversely, in respect to P1, the relationship creation time is specified but the proposed indirect relationship is not because, as aforementioned, this model does not focus on relationships management. By contrast, the rest of policies are satisfactorily expressed.

Content-based access control for social networks [22]. This proposal presents an automatic ACM that selects a particular policy for a post of an added message according to its content. Then, the main characteristic of this ACM is the identification of the content of each particular object.

Concerning policies, they are composed of five elements: *priority* which corresponds to the relevance of the policy; *name* which refers to an unique identifier; *explanation* that points out how the system has concluded; *attributes* which refer to the list of managed attributes; and *rules*, that indicate how elements match in the destination profile. In sum, they are expressed as:

Priority | *Name* | *Explanation* | *Attributes* | *rules*

Supposing that *priority* is *p*, *name* is *id* and *explanation* refers to the description of each policy $expPX$ $(X = \{1 - 7\})$, proposed access control policies are defined as follows:

P6 p | id | $expP6$ | - | $is\text{-}friend$

P7 p | id | $expP6$ | $-gender$, $-age$, $-studies$ | $[gender(female)AND(age < 30)]$ OR $[gender(female)AND(age < 40)ANDstudies(C.Science)]$ OR $[gender(female)AND$ $studies(C.Science)ANDstudies(Physics)]$

Therefore, a pair policies, P6 and P7, are successfully expressed.

Persona [8]. This proposal bases on Attribute Based Encryption (ABE) cryptography and consequently, the ACM which lays the bases of this work is ABAC. This cryptographic technique focuses on creating a pair of keys, to encrypt and decrypt, in regard to an established group of attributes. ABE schemes can be divided into two groups, Ciphertext-Policy ABE (CP-ABE) and Key-Policy ABE (KP-ABE). The former corresponds to the association of policies with ciphertexts and attributes to user keys and it is a remarkable technique in applications in which data is managed by multiple profiles, such as in hospitals or in the army. By contrast, the latter corresponds to the attachment of policies to user keys and attributes to ciphertexts, being useful in applications like auditing logs. The main difference between both approaches is that in CP-ABE attributes of users keys are known, while in KP-ABE they are hidden. Specifically, *Persona* applies CP-ABE. Thus, users creates keys regarding a set of attributes, encrypt data using encryption keys and distribute decryption keys among their contacts.

The strength of this approach focuses on dealing with untrusted service storages. Nevertheless, policies expressive power is not its main goal. In this work, "friend" is the only attribute used within policies, though disjunctive and conjunctive operators can be applied. Consequently, proposed access control policies are defined as follows:

P6 friend

The set of access control policies that Persona allows to create is not flexible enough. Policies elements are limited to attributes connected by disjunctive and conjunctive operators, that is, it can be compared with the management of groups. Moreover, the necessity of delivering decryption keys to chosen users supportss the unidirectional nature of relationships and thus, P6 is properly defined.

EASiER [9]. Similar to *Persona [8]*, this proposal focuses on ABE and, specially on CP-ABE. Therefore, policies are constructed through attributes combined with disjunctive and conjunctive operators.

According to proposed policies, the attribute applied is "friend" and they are defined as follows:

P6 friend

This approach, as *Persona*, does not focus on the establishment of expressive policies. Besides, assuming that decryption keys are delivered from data owners to the requester, established relationships are unidirectional and P6 is satisfactorily defined.

Secure and Policy-Private Resource Sharing [34]. This proposal presents an ABE solution, thereby based on an ABAC model, that achieves the definition of expressive policies regarding the social network graph (users represented as nodes and edges as relationships). Specially, Distance-Based Revokable Attribute Encryption (DBRA) is applied. Links are established between users to exchange decryption keys and the specification of access control policies bases on resource attributes and the distance between the resource owner and the administrator.

Regarding access control policies, they are composed of a set of access rules (ar) composed of conditions ($cond$), such that $\langle ar_1\rangle, \langle ar_2\rangle, ..., \langle ar_n\rangle$ where $ar = cond_1, cond_2, ...cond_m$. A particular condition is $dist(u, d)$ being u the requester and d the maximum distance between the requester and the administrator. Assuming the existence of resource attributes "relatives", "neighbours" and "friends", the proposed policies are defined as follows:

(P1) $\{\langle$RelativeType= "relatives".$dist(u,1)\rangle$, $\langle$NeighbourType= "neighbours".$dist(u,2)\rangle$, $\langle$FriendType= "friends".$dist(u,3)\rangle\}$

P6 $\{\langle$FriendType= "friends".$dist(u,3)\rangle\}$

A key relevant point of this approach is the management of resource attributes. Consequently, just a pair of policies can be defined. In respect to P1, the indirect relationship is defined to some extent because the relationship creation time cannot be managed. Conversely, P6 is properly defined following the same bases as in *Persona* [8] and *EASier* [9].

Appendix B.5. Ontology based access control (OBAC)

Three proposals fall in this category in which it is identified one of the most expressive proposals.

An Ontology-based Access Control Model for Social Networking Systems (OSNAC) [15]. OSNAC focuses on the management of a semantic ontology for WBSNs. It captures the WBSN semantic and constructs a model to manage it. In particular, it is described as a rule-based access control policy model in which rules are specified at user and at system level. The former refers to personal authorization rules established by users regarding protected resources and the latter corresponds to rules that govern the overall privacy policy of the system.

According to the proposed WBSN definition, OSNAC mainly manages users, data, relationships and user attributes. By contrast, relationship attributes are left aside and together with the restrictive possibilities for creating rules, access control is not managed in a fine-grained way.

To express user policies the following properties are required, where *sn* and *ac* allude to relationships and actions respectively:

- $Properties$={sn:isFriendOf, sn:isNeighbourOf, sn:isRelativeOf, sn:hasGender, sn:isYoungerThan, ac:canRead}

This model provides an interesting set of properties opened to the inclusion of new ones. Except for *isFriendOf* and *canRead*, presented properties have being created according to the model specifications. In fact, *isRelativeOf* and *isNeighbourOf* follow the same bases as *isFriendOf*.

Considering that v, u and t refer to WBSN users, s corresponds to the requester and a is the administrator proposed access control policies are constructed as follows:

(P1) $sn: isRelativeOf(a,\ u) \wedge sn: isNeighbourOf(u,\ v) \wedge sn: isFriendOf(v,\ s) \wedge ac: canRead(s,\ o)$

P2 $sn: isFriendOf(a,\ t) \wedge sn: isFriendOf(a,\ u) \wedge sn: isFriendOf(a,\ v) \wedge sn: isFriendOf(s,\ t) \wedge sn: isFriendOf(s,\ u) \wedge sn: isFriendOf(s,\ v) \wedge ac: canRead(s,\ o)$

P3 $sn: isFriendOf(a,\ s) \wedge sn: isFriendOf(a,\ u) \wedge sn: isFriendOf(s,\ a) \wedge sn: isFriendOf(s,\ u) \wedge sn: isFriendOf(u,\ a) \wedge sn: isFriendOf(u,\ s) \wedge ac: canRead(s,\ o)$

(P4) $sn: isFriendOf(a,\ v) \wedge sn: isFriendOf(v,\ s) \wedge sn: isFriendOf(a,\ u) \wedge sn: isFriendOf(u,\ s)\ \wedge\ ac: canRead(s,\ o)$

P5 $sn: isFriendOf(a,\ s)\ \wedge sn: isFriendOf(s,\ a)\ \wedge\ ac: canRead(s,\ o)$

P6 $sn: isFriendOf(a,\ s)\ \wedge\ [r \leftarrow\ ac: canRead(s,\ o)]$

(P7) $sn: hasGender(s,\ Female)\ \wedge\ sn: isYoungerThan(s,\ 30)\ \wedge\ ac: canRead(s,\ o)$
$sn: hasGender(s,\ Female)\ \wedge\ sn: isYoungerThan(s,\ 40)\ \wedge\ sn: hasStudied(s,\ C.Science)\ \wedge\ ac: canRead(s,\ o)$
$sn: hasStudied(s,\ C.Science)\ \wedge\ ac: canRead(s,\ o)$

Applying this model all proposed policies can be defined to some extent. In particular, P2, P3, P5 and P6 are satisfactorily expressed. On the contrary, even defining the indirect relationship proposed in P1 and given the lack of relationship attributes management, the existence of a relationship established before 2,000 is not specified. Likewise, P4 is partially defined. Multiple paths can be established but all of them with a particular length. Thus, the presented P4 gives access to users connected to the administrator by a pair of paths of two hops. Furthermore, it is also unspecified the fact that paths are different. Finally, in respect to P7, it is quite successfully defined through the establishment of as many access control policies as conjunctions. Nevertheless, the model does not manage multi-valued properties and granting access to a user who has studied c.science and physics becomes infeasible.

Semantic web based framework [7]. The general idea is to define a WBSN in terms of an ontology based on users' profiles, resources, relationships between users and between users and resources. Using this ontology the social network is modelled as a Social Network Knowledge Base (SNKB). Specifically, three types of policies are distinguished: *authorization policies* that consist of granting users permissions to execute privileges on objects; *admin policies* that state users who may specify access control policies for a certain privilege on an object; and *filtering policies* that establish prohibitions. Relationships have a particular trust assigned to them and policies are established accordingly. Besides, relationships are unidirectional and the bidirectional nature is created as a pair of unidirectional ones.

Concerning policies, SWRL is the access control policy language applied to implement them. Nonetheless, it cannot be used to deal with bidirectional relationships and they have to be managed out of SWRL. In general, in SWRL, access control policies are represented as antecedents, that encode conditions included in policies, and consequents, that encode authorizations and prohibitions. Considering the ontology applied in this ACM, the following instances are applied:

- Instances: $Relative$, $Neighbour$, $Friend$ and $Data$

Assuming that the administrator a grants read access to an object o to the requester r, proposed access control policies are defined as follows:

(P1) $\mathit{Read} : \mathit{Relative}(a, ?targetSubject1) \wedge Neighbour(?targetSubject1, ?targetSubject2) \wedge$

$Friend(?targetSubject2, ?targetSubject3) \wedge \mathit{Data}(?o) \Rightarrow \mathit{Read}(?r, ?o)$

P6 $\mathit{Read} : \mathit{Relative}(a, ?targetSubject1) \wedge \mathit{Data}(?o) \Rightarrow \mathit{Read}(?r, ?o)$

As a result, a pair of policies can be defined, being P6 the only one completely specified. On the other hand, P1 lacks the definition of the duration of the relationship.

Online social networks using MKNF+ [35]. A prioritized ontology based on an ACM for protecting users' data is proposed. It consists of a MKNF formalism that combines Decryption Logic (DL) and rules created by Answer Set Programming (ASP). Furthermore, this model includes priority as an access control policy element to prevent conflicts caused by contradictions between each user's access control policies.

Concerning policies, they are composed of two types of predicates, DL-predicates and non-DL-predicates. The former bases on DL language and the latter focuses on unary or binary predicates. Specifically, the following predicates, already defined in [35], are the one applied herein:

- DL-Relationships=$\{IS{-}FRIEND{-}OF(Person,\ Person),\ ELEMENT(Object)\}$ where $ELEMENT$ may refer to a photo, a message or any other element in a WBSN.
- Non-DL-Concepts=$\{o(Object),\ s(Person)\}$

Assuming that src refers to the requested resource, sbj corresponds to the requester, a refers to the data owner and p refers to a certain type of priority, the following policy is defined:

P6 $K\ (?src),\ K\ s(?sbj),\ K\ IS{-}FRIEND{-}OF(a,\ ?sbj),\ K\ ELEMENT(?src) \rightarrow$ $K\ permit(a, ?sbj, READ, ?src, p)$

In sum, relationships are pointed out as directed label edges and then, P6 is properly defined.

Appendix C. Enforcement functions

The notation used to define each function corresponds to the name of the function, the input parameters (arguments), a set of predicates that refers to the establishment of variables or conditions and the returned value if required. It is based on [59] and it is formally represented as follows: $Function - Name(Arguments) \lhd\ Predicate1\ Predicate2\ ...\ [Return - Value] \rhd$

Moreover, symbol . is used to access to the content of an element. For instance, given a user (s), $s.id$ is used to access to the user's id. Besides, the expression $list[pos]$ refers to the access to an element located in position pos within the list $list$. For example, given the list i $=\{v, t, y\}$, $i[1]$ corresponds to t. Finally, it shlould be noticed that functions $MatchC$ and $MatchO$, that refer to the verification of conditions and obligations respectively, have to be implemented according to each particular case.

CheckAccess

$CheckAccess(s, o, r, \rho; out\ result : BOOLEAN) \lhd \rho_s \in \rho; \rho_o \in \rho; \rho_{rt} \in \rho; subAtt = GetSubAtt(s, \rho_s);$
$objAtt = getObjAtt(o, , \rho_o); a = GetAdmin(o); \partial_b = GetObligations(\rho); \partial_c = GetConditions(\rho);$

$rt = CreateRT(a, s, 1); openThread = ContinuityCheckAccess(s, o, r, \rho, \partial_b, \partial_c, rt)$
$result = (\forall\rho((if(\rho_s\ NOT\ \emptyset) \Rightarrow Match(subAtt, \rho_s)) \wedge (if(\rho_o\ NOT\ \emptyset) \Rightarrow$
$Match(objAtt, \rho_o)) \wedge (if(\rho_{rt}\ NOT\ \emptyset)$
$\Rightarrow MatchRT(\rho_{rt}, rt)) \wedge r = \rho.r \wedge MatchB(s, o_1, r, \rho, \partial_b)$
$\wedge MatchC(s, o, r, \rho, \partial_c)))\triangleright$

ContinuityCheckAccess
$ContinuityCheckAccess(s, o, r, \rho, \partial_b, \partial_c, rt; out\ \ result\ :\ BOOLEAN) \triangleleft$
$\rho_s \in \rho; \rho_o \in \rho; \rho_{rt} \in \rho; subAtt = GetSubAtt(s, \rho_s);$
$objAtt = getObjAtt(o, , \rho_o); a = GetAdmin(o)$
$result = (\forall\rho((if(\rho_s\ NOT\ \emptyset) \Rightarrow Match(subAtt, \rho_s)) \wedge (if(\rho_o\ NOT\ \emptyset) \Rightarrow$
$Match(objAtt, \rho_o)) \wedge (if(\rho_{rt}\ NOT\ \emptyset)$
$\Rightarrow MatchRT(\rho_{rt}, rt)) \wedge r = \rho.r \wedge MatchB(s, o_1, r, \rho, \partial_b)$
$\wedge MatchC(s, o, r, \rho, \partial_c)))\triangleright$

Match
$Match(ATT(\omega), \rho_\omega; out\ result : BOOLEAN) \triangleleft$
$att(\omega)_i \in ATT(\omega)$
$result = (\forall att(\omega)_i \Rightarrow (if\ \ (att(\omega)_i.type = \mathcal{FV}) \Rightarrow$
$VerifyFVAttTypes(\gamma^j_{att(\omega)_i}, \rho_\omega)\ \vee\ (if(att(\omega)_i.type = \mathcal{D}) \Rightarrow$
$VerifyDAttTypes(\gamma^j_{att(\omega)_i}, \rho_\omega) \vee\ (if(att(\omega)_i.type = \mathcal{B}) \Rightarrow$
$VerifyBattTypes(\gamma^j_{att(\omega)_i}, \rho_\omega)))\triangleright$

CreateRT
$CreateRT(v, s, hop; out\ result : rt) \triangleleft$
$result = ((\forall hop < 6) \longrightarrow (nC = GetNumContacts(v) \wedge (\forall i < nC \longrightarrow$
$(c = getConnectedUser(v, i) \wedge (Storeforward\ and\ backward$
$relationships\ and\ length) \wedge ((if(c = s) \Rightarrow (Path\_completed)) \vee (if(c\ NOT\ v) \Rightarrow$
$CreateRT(c, s, hop + 1)) \vee (if(c = v) \Rightarrow (Path\_broken))))))\triangleright$

MatchRT
$MatchRT(\rho_{rt}, rt; out\ result : BOOLEAN) \triangleleft$
$result = ((if(\rho_{rt}.\sigma = \emptyset \wedge \rho_{rt}.\varpi = \emptyset \wedge \rho_{rt}.\delta = \emptyset) \Rightarrow true) \wedge (if(\rho_{rt}.\delta\ NOT\ \emptyset) \Rightarrow$
$VerifyClique(\rho_{rt}, rt)) \wedge (if(\rho_{rt}.\delta\ NOT\ \emptyset) \Rightarrow ((if(MatchPathPolicy(\rho_{rt}.\sigma, rt, \rho_{rt}.\varpi))$
$\Rightarrow true) \vee false)) \wedge (if((\rho_{rt}.\delta\ NOT\ \emptyset) \wedge (\rho_{rt}.\varpi\ NOT\ \emptyset)$
$\Rightarrow (pathsDivided = GetPathsPolicies(\rho_{rt}.\sigma) \wedge (\forall i < pathsDivided.paths$
$\longrightarrow (pathsSatisfaction = MatchPathPolicy(pathsDivided.paths[i], rt, \rho_{rt}.\varpi)))$
$\wedge ((if(VerifyPolicy(pathsSatisfaction, pathsDivided.listOp))$
$\Rightarrow true) \vee false)))))\triangleright$

MatchPathPolicy
$MatchPathPolicy(pathCond, rt, \varpi); out\ result : BOOLEAN) \triangleleft$
$pLength = GetLengthPath(pathCond);$
$rtPathsL = GetEnrichedPathsWithLength(pLength, rt); cont = 0$
$result = (\forall i < rtPathsL \longrightarrow (\forall j < pLength \longrightarrow$
$(if(MatchDirectPaths(GetDirectRelAtt(rtPathsL[i], j),$
$GetDirectRelAtt(pathCond, j))) \Rightarrow ((cont{+}1 \wedge (if(cont \geqslant \varpi) \Rightarrow true)) \vee$
$false))))\triangleright$

GetEnrichedPathsWithLength

$GetEnrichedPathsWithLength(length, rt); out\ result : EnrichedPaths) \lhd$
$result = (\forall i < rt.paths \longrightarrow ((if(rt.paths[i].length = length) \Rightarrow true) \vee false)) \rhd$

GetPathsPolicies

$GetPathsPolicies(length, rt); out\ result : \{EnrichedPaths, ListOpe\}) \lhd$
$result = (\sigma\ is\ processed\ storing\ \sigma.\psi_i\ in\ EnrichedPaths\ and\ operators\ that\ linked\ each\ \sigma.\psi_i\ in\ ListOpe)) \rhd$

GetDirectRelAtt

$GetDirectRelAtt(pathCon, pLength); out\ result : ListE) \lhd$
$cont = 1$
$result = (\forall i < pathCon.length \longrightarrow ((if(pathCon[i] = ";" \wedge cont = pLength) \Rightarrow (Store\ relationships\ between\ the\ previous\ identified\ ";"\ and\ this\ ";") \vee cont + 1))) \rhd$

GetErtDivision

$GetErtDivision(rels); out\ result : \{ListE, ListOpe\}) \lhd$
$result = (rels\ are\ processed\ storing\ rels.fert_i\ and\ rels.bert_i\ in\ ListOfErtand\ operators\ that\ linked\ each\ rels.fert_i\ and\ rels.bert_i\ in\ ListOfOperators)) \rhd$

MatchDirectPaths

$MatchDirectPaths(listPathsOfRT, pathsPolicy); out\ result : BOOLEAN) \lhd$
$ListPOPolicy = getErtDivision(pathsPolicy)$
$result = ((\forall i < ListPOPolicy.paths \longrightarrow (\forall j < listPathsOfRT \longrightarrow (resultMatch = Match(listPathsOfRT[j], ListPOPolicy.paths[i]) \wedge (if(resultMatch = true) \Rightarrow ListSatisfaction[i] = true)))) \wedge (\forall i < ListSatisfaction \longrightarrow ((if(ListSatisfaction[i] = true) \Rightarrow true) \vee false))) \rhd$

VerifyPolicy

$VerifyPolicy(listBoolean, listOpe\ result : BOOLEAN) \lhd$
$result = (Verify\ listBoolean\ against\ listOpe) \rhd$

VerifyClique

$VerifyClique(rt, \delta, \sigma); out\ result : BOOLEAN) \lhd$
$listlengthEP = CalculateCliqueUsers(\delta);$
$pathsDivided = GetErtDivision(\sigma)$
$result = ((\forall i < listlengthEP \longrightarrow (rtpaths[i] = GetEnrichedPathsWithLength(i, rt) \wedge ((if(rtpaths[i].length \geqslant listlengthEP[i]) \Rightarrow (pathsClique.ADD(rtpaths[i]))) \vee false))) \wedge$
$(\forall i < pathsClique.length \longrightarrow (\forall j < pathsClique[i].length \longrightarrow ((if(MatchDirectPaths(GetDirectRelAtt(pathsClique[i], j), pathsDivided) \Rightarrow acceptedPaths.ADD(pathsClique[i])) \vee false))) \wedge$
$listNodes.ADD(GetFirstNode(rt[i][j])) \wedge$

$listNodes.ADD(GetLastNode(rt[i][j])) \wedge (\forall i <$
$acceptedPaths.length \longrightarrow (\forall j < acceptedPaths[i].length$
$\longrightarrow (node = GetNode(acceptedPaths[i], j) \wedge$
$(if(node\ NOT\_IN\ listNodes) \Rightarrow listNodes.ADD(node)))))$
$\wedge ((if(ListNodes.length = \delta) \Rightarrow true) \vee false))\rhd$

CalculateCliqueUsers
$CalculateCliqueUsers(\delta); out\ result : ListINTEGER) \lhd$
$result = ((if(\delta = 2) \Rightarrow ListClique[0] = 1) \vee (if(\delta > 2)$
$\Rightarrow (\sum_{K=1}^{\delta} ListClique[K] = (P(K, N) + 1))))\rhd$

GetNode
$GetNode(path, pos); out\ result : STRING) \lhd$
$result = (return\ node\ located\ at\ pos)\rhd$

GetFirstNode/GetLastNode
$GetFirstNode/GetLastNode(path); out\ result : STRING) \lhd$
$result = (return\ the\ first/\ last\ node)\rhd$

GetLengthPath
$GetLengthPath(path); out\ result : INTEGER) \lhd$
$cont = 1$
$result = (\forall i < path.length \longrightarrow (if(epath[i] = ";") \Rightarrow cont + 1))\rhd$

MatchB/MatchC
$MatchB/MatchC(s_1, o_1, r, \rho, \partial_b/\partial_c; out\ result : BOOLEAN) \lhd$
$result = (-)\rhd$

GetConnectedUser
$GetConnectedUser(s, pos\ result : STRING) \lhd$
$result = (id\ of\ a\ user\ in\ position,\ pos,\ connected\ to\ s)\rhd$

GetNumContacts
$GetNumContacts(s\ result : INTEGER) \lhd$
$result = (s.contacts.length)\rhd$

VerifyBAttTypes
$VerifyBAttTypes(\gamma^j_{att_i}, policyAttPred; out\ result : BOOLEAN) \lhd$
$result = (\gamma^j_{att_i}\ is\ verified\ against\ policyAttPred\ )\rhd$

VerifyFVAttTypes
$VerifyFVAttTypes(\gamma^j_{att_i}, policyAttPred; out\ result : BOOLEAN) \lhd$
$result = (\gamma^j_{att_i}\ is\ verified\ against\ policyAttPred\ given\ all\ applied$
$\mathcal{T}\ and\ \mathcal{L}\ operators)\rhd$

VerifyDAttTypes
$VerifyDAttTypes(\gamma^j_{att_i}, policyAttPred; out\ result : BOOLEAN) \lhd$
$result = (\gamma^j_{att_i}\ is\ verified\ against\ policyAttPred\ given\ all\ applied\ \mathcal{X}\ operators)\rhd$

GetSubAtt/GetObjAtt

$GetSubAtt/GetObjAtt(-; out\ result : S/O/RTATTRIBUTES) \lhd$
$result = (s/o/rt\ attributes) \rhd$

GetAdmin

$GetAdmin(o; out\ result : SUBJECT) \lhd$
$result = (administrator\ of\ o) \rhd$

GetConditions/ GetObligations

$GetConditions/GetObligations(\rho; out\ result : partial_b/partial_c) \lhd$
$result = (conditions\ or\ obligations\ within\ \rho) \rhd$