An ABAC Model with Trust and Gossiping (ABAC–TG) for Online Social Networks

Adi Swissa, Ehud Gudes

An ABAC model with Trust and Gossiping (ABAC–TG) for online social networks

Adi Swissa[1] and Ehud Gudes[1,2]

[1] The Open University, Department of Mathematics and Computer Science, Raanana, Israel
[2] Ben-Gurion University of the Negev, Department of Computer Science, Beer-Sheva, Israel
adi.swissa123@gmail.com, ehudgu@openu.ac.il

Abstract. In this paper, we propose an attribute-based access control model called ABAC–TG for online social networks (OSNs). This model comprehensively considers user and object attributes and two main social attributes: trust and gossip, which are calculated based on the Ego-node (the user sharing the information) point of view. Each user is evaluated trust and gossip wise by several criteria, such as total number of friends, number of interactions between two users, and more. A new algorithm for calculating user gossiping value by graph clustering is defined, and this gossiping value can also be used for trust calculation. The ABAC model is formally presented, including rules and attribute definitions, and is demonstrated by several use case scenarios. The gossip and trust assessments provide more accurate and viable information-sharing decisions that serve the purpose of more precise and flexible authorizations.

This work is novel in two respects. First, we are using trust and gossip as dynamic attribute calculations. And second, we present a new algorithm for calculating the user's gossip value from the ego user point of view and use it either as part of the trust attribute calculation or as a separate attribute in the ABAC model.

Keywords: Attribute based access control (ABAC), gossip, trust, online social networks.

1 Introduction

As online social networks (OSNs) increase in size and more people use them as their primary Internet website, the volume of information shared in OSNs keeps on growing.

The public accessibility of such networks with the ability to share opinions, thoughts, information, and experience offers great promise to people and communities. In addition to individuals using such networks to connect to their friends and families, governments and enterprises have started exploiting these platforms for delivering their services to citizens and customers [4]. Because of the sensitive and private information that is commonly stored in these networks, controlling access to this information is becoming very important and that depends largely on the level of trust that members have with each other. Several access control models for OSN based on trust have re-

cently appeared [4,5,11,14], but none of them uses the attribute of gossiping as a significant factor in the access control model. Zhang et. al [1] present an attribute-based access control (ABAC) model for OSN, but does not use either Trust nor Gossiping attributes. Since gossip is one of the oldest and most common means of information sharing among people, we consider it also very important for influencing access control.

This paper aims to demonstrate a new ABAC model called ABAC–TG, for an online social network, which combines privacy, trust, and a gossip model.

The general idea is to use the ABAC model with additional complex attributes such as user trust and gossip, calculated by clustering. The gossip attribute may be used as part of the trust calculation or as a separate attribute in an ABAC rule. The user selects attributes and defines rules for defining the access for a specific object. This model is extensible by adding additional dynamic attributes. The examples demonstrating the model use actions provided by a Facebook like network but are not limited to it.

This new model has three significant advantages: First, the model calculation is dynamic - the trust is calculated based on user selection and network parameters, and the gossip is dependent on specific network interactions.

The second is flexibility and scalability – we can add or remove attributes and decide on threshold values for trust and gossip calculations. The third is simplicity – the user will choose simple attributes and define the access to his objects in terms of these attributes. Thus, our model provides a solution to one of the most significant social network problems, the control and prevention of the spread of sensitive private information in the network.

The rest of this paper is structured as follows: Section 2 provides background information and an overview of related work. Sections 3 and 4 describe the new ABAC model, where Section 3 describes the model, trust, and gossip attributes calculations, and Section 4 discusses the rules and attributes used in detail and presents several use-cases of using the model. The last section presents the relevant conclusions and discusses future work.

2 Background and related work

As mentioned earlier, this paper's main goal is to propose a new ABAC model called ABAC–TG, which combines privacy, trust, and gossip attributes. In this section, the relevant background is provided.

An ABAC model relies upon evaluating attributes of the subject, attributes of the object, environment conditions, and the formal relationship or access control rule or policy defining the allowable operations for subject-object attribute combinations. All ABAC solutions contain these basic core capabilities to evaluate attributes and enforce rules or relationships between those attributes [1]. Examples of such rules for a social network are presented in this paper in section 4.3.

The access control model in Facebook is based on roles. It does a reasonable job of access control while handling millions of operations/seconds from its billion users. The mechanism of Facebook is a function of communication history among users (for instance, the existence of friendship is necessary for certain policies), however even though it's a quite simple model, users often do not use it properly. An analysis of Facebook access control model and its privacy problems has recently appeared in [2, 8, 11]. However, Facebook access control does not use trust or gossip and lacks reliance on users' specific attributes and objects for Access control.

Recently, a trust-based model for a social network called RTBAC was presented in [5]. The RTBAC model is a combination of User-Trust attributes, based on real OSN characteristics, within an RBAC model that usually grants permissions solely to roles.

The trust value is defined on a scale of 0 to 1 since the decision of sharing information with a certain user is defined as a probability variable, 0 being no sharing willingness at all, 1 being definite sharing willingness. The trust model is based on several criteria such as quality of Friendship, connection strength, and users' similarity. We will use this model in our new ABAC-TG model to calculate the trust value described in section 3.3. Another relevant work [14] describes the way trust can be used to identify adversaries and limit information flow to them.

The fast spread of information is a common and essential feature of social networks. A simple model of diffusion shows how bounded rational individuals can, just by tracking gossip about people, identify those who are most central in a network according to "diffusion centrality." [7, 9]. Gossip can essentially be defined as information passed from one individual (originator) to another (gossiper) about an absent third individual [13].

In [6], an algorithm is given for provably finding the clusters, provided there is a sufficiently large gap between internal density and external sparsity. This clustering is used to build knots of trust between users in [12]. Knots of trust are groups of community members having overall "strong" trust relations between them. In order to provide a member with reputation information relative to her viewpoint, the system must identify the knot to which that member belongs and interpret its reputation data correctly. Such clustering can be used to identify and measure the amount of "gossipness" of users and groups of users since users tend to gossip with other users they trust on. We'll use this clustering to define a "gossipness" measure in our proposed ABAC model.

The most relevant article to this work is by Zhang et. al. [10]. They present an ABAC model for social networks with many examples for rules involving various attributes. Our model is similar but more general than [10] since it explicitly includes two important new attributes: Trust and Gossiping attributes. Our model is dynamic and extensible and can be used for any online social network.

3 A new ABAC model with trust and gossip for online social network

This section presents the main theme of this paper. The aim is to define a new ABAC model for online social networks (for instance, Facebook, Twitter, etc.) and to incorporate user trust calculation and gossip calculation in the ABAC model.

3.1 Establishment of ABAC-TG model for online social network

Today, Facebook, the world's most popular social network, uses the RBAC model to manage user roles on a specific object or action [2]. Our new ABAC-TG model will replace the RBAC model. The key difference with ABAC is the concept of policies that express a complex Boolean rule set that can evaluate many different attributes [1].

In the ABAC-TG model, the user will choose attributes from a predefined list and define rules. For instance, the user doesn't want the specific objectA to be visible to users who are mostly gossiping; or the user doesn't want the particular objectA to be visible to teenagers or to users who have low trust value.

Relevant attributes may be: a number of friends, age, education, job title, work, family status, friends type (e.g., in comparing to ego user's age, education, work, city, etc.), action types (e.g., comment to friends with the same age, work, city), and attributes of the objects themselves. More attributes will be described in chapter 4.

Figure 1 demonstrates the access decision-making of the ABAC-TG model. For each object, the model checks if the object has roles restrictions, if yes, the model checks if the user has fitting attributes and grants access accordingly.

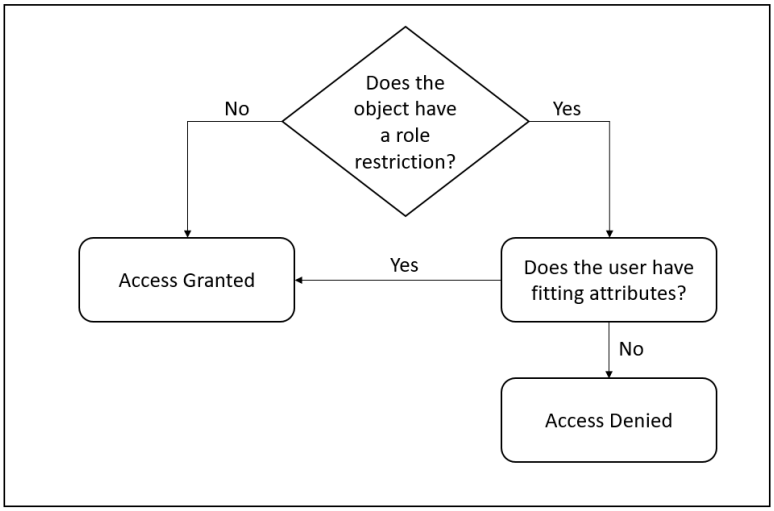

Fig. 1. Access decision in the ABAC-TG model

The new ABAC model includes user trust and user gossiping. Therefore we define two new attributes: GossipingAtt and TrustAtt.

GossipingAtt – this attribute describes the gossiping level of the user. The value will be calculated using a network-directed graph containing the interactions (messages, likes, comments, shares, etc.) between the user to other users, applying the clustering algorithm, and deciding if the user is gossiping. The gossiping value is on a scale of 0 to 1, where 0 means a total gossiping user, and 1 means not gossiping at all. The gossiping algorithm is described in section 3.4.

TrustAtt – this attribute describes the social trust level of the user. The value is calculated by the model presented in [5]. The trust calculation is described in section 3.3. Note that in [5] the main goal is to compute the user's trust, which is the base for access control, while in our model, trust is just one attribute among several attributes of the ABAC model. Besides, we can adjust the trust calculation by using the GossipingAtt attribute. In case that the user chooses TrustAtt and GossipAtt in the rules, we do not use them twice in the trust model calculation.

Therefore, our model is dynamic due to the dependency on the user selection in calculating trust and ABAC attributes that may change in time. This capability defines a new perspective on the trust model from [5] with the new GossipingAtt and user selection, which affects the trust calculation.

3.2 Dynamic ABAC-TG model

The decision algorithm of ABAC-TG is the same as the original ABAC model, by evaluating different attribute conditions and getting a Boolean value as a result for "allow access" or "deny access" (see figure1).

If the user chooses the TrustAtt and an attribute1 that is already part of the trust formula, we have two options: first is to adjust the trust formula and remove attribute1 in order not to use it twice. The second is not to change the trust formula and let the user influence the weight of the attributes. In this paper, we choose the first option by removing it from the trust formula and use it in the ABAC rules.

For example, if the user defines a rule of – "I want that only my trusted friends (0.8) and my non-gossiping friends (0.8) will be able to see my image1". In this case, we'll adjust the trust formula and remove the gossiping parameter and use it only once in the rule. As a result, the user affects the trust value as a dynamic trust value and changes the gossiping value's weight as a more effective attribute (due to the 50% of the rule and not just a small part in the trust value). This use case is demonstrated in section 4.3 on use case 3.

3.3 Trust Calculation

In our ABAC-TG model, we're using the trust attribute (denotes as TrustAtt) as yet another attribute. In order to calculate the TrustAtt value, we'll use the formula described in [5],

with an extension to include the new friendship characteristic of gossiping value. The original formula from [5] to compute the user trust value (denotes as UTV) is shown in Figure 2.

The different factors and their corresponding weights are explained in detail in [5]: taken into consideration user credibility factors (knowledge factors such as the total number of friends) denotes as u, and connection-based factors (friendship characteristics such as mutual friends) denotes as c. In order to add the gossiping value into the formula we just add another factor and its corresponding weight, such that the total sum of weights (denotes as w) is still 1.

The gossiping value will affect the trust value in the same manner of every friendship characteristic. The gossiping value calculation is described in section 3.4, and the value is the probability from a scale of 0 to 1, 0 mainly being gossiping, 1 being not gossiping at all.

The trust value is also expressed on a scale of 0 to 1, 0 being no sharing willingness at all, 1 being definite sharing willingness.

$$u = \langle WiUi \rangle = \frac{\sum_{i=1}^{|u|} WiUi}{\langle W \rangle |u|} \quad (8)$$

$$c = \langle WiCi \rangle = \frac{\sum_{i=1}^{|c|} WiCi}{\langle W \rangle |c|} \quad (9)$$

$$UTV = \frac{c \cdot |c| + u \cdot |u|}{|c+u|} \quad (10)$$

Fig. 2. The formula from [5] to calculate the trustAtt value. User Trust Value (UTV) is calculated as the weighted average of user credibility and connection strength. The weight is set according to the relative number of attributes in each category.

For example, in table 1, we have two users: UserA and UserB, which are friends of Alice. Alice gave them different scores regarding what she thinks, based on her friendship experience. In this case, UserA and UserB have different gossiping values that affect the calculated trust value.

Table 1. An example of trust value including gossiping attribute

Username	Knowledge Value (u)	GossipingAtt	Friendship Value (c)	TrustAtt (UTV)	Result
UserA	0.51	0.23	0.35	0.43	The trust value is lower with a gossiping user
UserB	0.77	0.98	0.92	0.845	The trust value is higher with non-gossiping user

3.4 Gossiping Calculation

This section defines the algorithm for calculating the gossiping value for users in a social network. The gossiping attribute's goal is to identify a set of friends who would leak the shared information to an adversary. The gossiping value is the probability from a scale of 0 to 1, 0 being mostly gossiping, 1 being not gossiping at all.

The GossipAtt is calculated by taking into consideration connection-based factors: the number of human interactions between two users. The gossiping calculation is in the context of an Ego user perspective (the user sharing the information) to his friends and not general to the whole network. For example, from Alice's perspective, Bob's gossiping value is 0.8, but from David's perspective, it's 0.2, which leads to that Alice will share more information with Bob than David will share with Bob.

We assume, for this paper, that gossip serves to strengthen the relationship between gossipers and weakens the relationship between the victim and each gossiper [13]. Therefore, the friends with which the Ego user has fewer interactions will have the potential to gossip about him. Thus, in our algorithm, we focus on gossiping friends and exclude the "Ego's best friends" from the graph. We define "best friends" as users who have more than R interactions with each other. For example, a relationship of 100 interactions and above is with a high probability of being a best friend.

In our model, we consider only two levels of ego user's friends and friends of friends, for three reasons: the first reason is to have a comprehensive perspective on user's interactions and an extensive network graph. The second reason is to enable the ego user to share his post object with a wide enough forum (but not to a huge subnetwork) and limit it to non-gossiping users. The third reason is to restrict the network size to a reasonable amount of nodes to achieve better running time.

To compute the gossiping value, we use graph clustering based on the logic described in [12,6]. A knot [12] is a subset of community members identified as having overall strong interaction relations. Two members i and j should belong to the same knot if i has high direct interaction in j denoted I_M (i, j). Knots are groups of members with strong interactions, sharing the same gossiping value from the Ego user perspective. Ego is an individual focal node, which is the specific user from which we consider the gossiping flow. It is, therefore, plausible that the gossiping of the same user may differ significantly between different Ego users.

A community is modeled as a directed graph G = (V, E) that describes a social network, where V is the set of network's users, and E is the set of directed and weighted edges representing the users' interactions. The weight on a directed edge from vertex i to vertex j is the level of direct interactions i has in j at time t and is denoted by I_Mt(i, j). Since we deal with the state of the graph at time period t, we omit the time indicator for simplicity. An edge (ui ,uj) ∈ E exists only if ui has interactions with uj.

We refer to the task of identifying knots as graph clustering. In a social networking graph, these clusters could represent users with similar interactions. More specifically,

we aim to find a partition of the community graph based on the direct interactions between pairs of members. For this purpose, we replace the interaction relations between any two members I_M(i, j) and I_M(j,i) with a weaker relation named Mutual Interactions in Member (MIM) which is the minimum of the above two values, that is, the two directed edges (i, j) and (j,i), are replaced by a single, undirected edge whose weight is MIM(i, j) = MIM(j,i) = min{IM(i, j),I M(j,i)}.

This way, we can use the edge relation as the input for the clustering algorithm, which must decide if its two end-vertices should reside in the same cluster or not. Intuitively, the new relation is more stringent because it considers the minimum level of mutual interactions between any two members as the representing value of gossiping between them.

Gossip Algorithms:

Algorithm 1. calculates the gossiping value for Ego user's friends. The algorithm returns a map of clusters and their gossip value.

Algorithm 1. CalcGossipGraph(G, u_e)

Input: G = (V, E) an undirected graph that describes the social network of Ego's user, which vertices represent users and edges represent the interactions relations between the users at their end-point vertices

u_e the ego user.

Output: M: a map of clusters and their gossip value.

1: R = 100
2: **for** each ego's direct friends u_i **do**
3: **if** the MIM (u_e, u_i) >= R **then**
4: Remove u_i from graph G.
5: Add u_i to cluster "best friends" and set in the map M with gossipValue equal to 1.
6: **end if**
7: **end for**
8: Remove u_e from graph G
9: Create clusters based on the graph G.
10: **for** each cluster C **do**
11: set gossipValue = min {Sum of MIM in C / (number of vertices in C * r), 1}.
12: **end for**
13: Return M as map of clusters and gossipValues.

In Algorithm1, we defined Ego's best friends as friends which MIM bigger than 100 interactions, denotes as R, (a parameter which obviously can be changed), as gossip serves to strengthen the relationship between gossipers and weakens the relationship between the victim and each gossiper [13].

Lines 1-8 remove the ego user and his best friends from the graph. Line 9 creates the clusters based on the algorithm described in [12]. Finally, line 11 sets the gossiping

value of each user to the average MIM in the cluster, normalized by the factor R. An example for this calculation is shown below.

Algorithm 2. returns the gossiping value for a specific user from the Ego user perspective. This algorithm is calls algorithm 1 to calculate the gossiping clusters.

Algorithm 2. GetUserGossipValue(G, u_e, u_i)

Input: G = (V,E) an undirected graph that describes the social network of Ego's user, which vertices represent users and edges represent the interactions relations between the users at their end-point vertices

u_e the ego user.

u_i – the user to evaluate the gossip value.

Output: V: U u_i 's gossiping value

1: M = CalcGossipGraph (G, u_e).

2: Find u_i in M.

3: V = u_i 's clustering gossiping value.

4: Return V.

Table 2 and figure 3 that appears below present an example of Algorithm 1, with ten users: friends and friends of friends of Alice. Based on their interactions, we calculate the gossiping value of each user from Alice's perspective. The edges represent the interactions with the value of MIM.

Users 1,2,3 have more than R interactions with u_e and defined as "Ego's user best friends" with a gossiping value of 1 (see cluster "best friends" in figure).

Users 4,5 has less than R MIM value, and they are sharing the same cluster1, with gossiping value of C1 = min {60/ (3 * 100), 1} = 0.2.

Users 7,8,10 sharing the same cluster2 with gossiping value of C2 = min {245/ (6 * 100), 1} = 0.4.

Users 9,6 sharing the same cluster3 with gossiping value of C3 = min {250/ (4 * 100), 1} = 0.62.

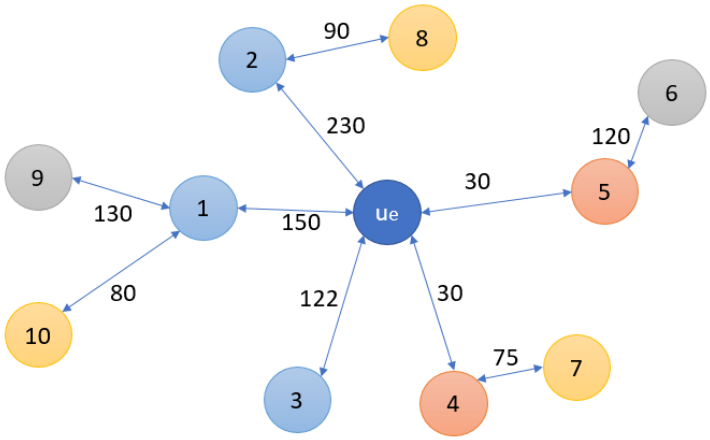

Fig. 3. An example of CalcGossipingGraph is described in table2. Nodes 1,2,3 are in cluster "best friends", nodes 4,5 are in cluster1, nodes 7,8,10 are in cluster2, and nodes 9 and 6 are in cluster3.

Table 2. An example of gossiping calculation for Alice's friends

Username	Is Friend of Ego User?	Is best friends of Ego User?	Part of Cluster #Number	GossipingAtt
User1	Yes	Yes	---	1
User2	Yes	Yes	---	1
User3	Yes	Yes	---	1
User4	Yes	No	1	0.2
User5	Yes	No	1	0.2
User6	No	No	3	0.62
User7	No	No	2	0.40
User8	No	No	2	0.40
User9	No	No	3	0.62
User10	No	No	2	0.40

Note that the gossiping value calculation may run once for each ego user and stored in cache memory. The recalculation of it, which reflects the model dynamics, may be a parameter that depends on the number of new interactions in the ego user subnetwork, which the network administrator can set.

4 Formalization and applications of the ABAC-TG model

This section describes ABAC-TG model's formalization by defining user's attributes, object's attributes, and rules. We also represent five main use cases for using this model in an online social network.

4.1 Definition and formal descriptions of ABAC-TG model components

We define a rule definition for our ABAC-TG model. Our syntax is based on article [10], but it's simpler and focuses on trust value, gossiping value, and the dynamic model, which the user defines.

Definitions:

Definition 1. (User set, U): User represents the entity that performs social network's user accesses. In the social network, the set U contains all users. The users can upload and access media resources and perform various operations on other users and resources available in the system.

Definition 2. (Object set, OB): a post entity. The entity includes many items such as images, texts, videos, comments, etc.

Definition 3. (Actions set, AC): a post action. The actions include different activities such as display, share, post, like, comment, etc.

Definition 4. (User attribute set, AU): The set AU includes user basic information attributes, user social relationships attributes, and user community attributes. The user basic information attributes include name, age, identity, hobbies, and user-level attributes (as described below in section 4.2).

Definition 5. (Object attribute set, OU): The set OU includes object basic information attributes. The object's basic information attributes include attributes such as publish date time, location, object type, related objects (e.g., comment on an object has a related object of the original object – post). Image object has corresponding objects of the users who appear in the image, check-in object has a location type attribute such as restaurant, work office, etc.).

Definition 6. (Attribute expression set, AE): The set AE includes AU and OU expressions, separated by the *and* (∧) and *or* (∨) operations. The not sign is allowed by adding '!' before an expression.

Definition 7. (Basic Rule set, BR): a rule definition in ABAC-TG model as < U, OB, AC, AE>. If the rule condition is true, the user can access the object. If the rule is false, the user cannot access the object.

Definition 8. An ABAC-TG instance is a tuple of < U, OB, AC, AE>, which is a combination of user, object, actions, and attributes. Figure 4 describes the ABAC-TG model and the connections between the various components.

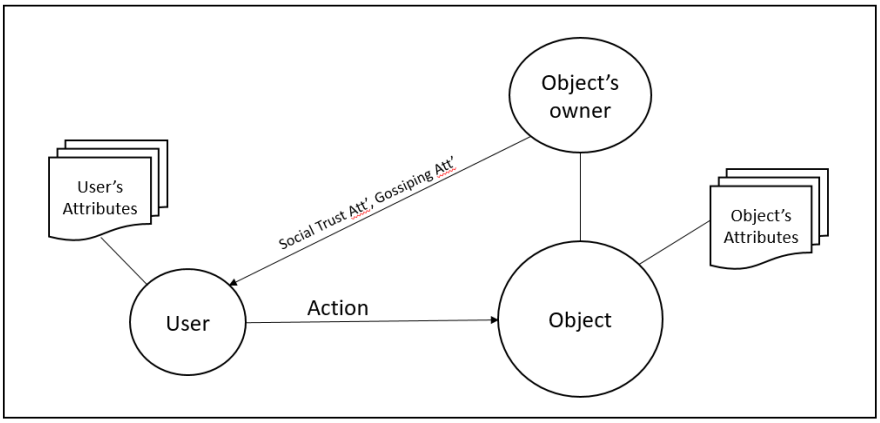

Fig. 4. ABAC-TG model components

4.2 Attributes definitions.

In this section, we define the main attributes that will be used in our ABAC-TG model. These attributes are examples, and we can add any attribute that is relevant to online social networks.

Attributes definitions:

Definition 1. (TrustAtt): this attribute describes the user's social trust level, calculated as described in section 3.3.

Definition 2. (GossipingAtt): this attribute describes the user's gossiping level, calculated as described in section 3.4.

Definition 3. (AgeLevel): this attribute describes the user's age level *(e.g. level1* is age 10-20, level2 is age 20-40, level3 is age 40-60, level4 is 60+).

Definition 4. (Education): this attribute describes the education of the user, for example: "Bachelor of Science in Computer Science," "Bachelor of Laws in Law", etc.

Definition 5. (Job title): this attribute describes the job title of the user, for example: "development manager", "software developer", "product owner", etc.).

Definition 6. (Work): this attribute describes the workplace of the user, for example: "Microsoft", "Google", "Amazon", etc.

Definition 7. (Family status): this attribute describes the family status of the user, for example, single, married, divorced, etc.

Definition 8. (Friends type): this attribute describes the "friends' type", for example: in comparison to a user's age, education, work, city etc.

Definition 9. (Action types): this attribute describes the "Action types" of the user, for example, comments to friends with the same age, work, city, etc.

Definition 10. (Gender): this attribute describes the gender of the user: female or male

4.3 Use cases and examples

This section demonstrates the expressiveness and usability of the ABAC-TG model. We design several real-life scenarios and give their corresponding rules in our logic. We can define additional scenarios as needed as this model is dynamic and extensible.

Use case 1. The user defines a rule which includes trustAtt and few attributes that are not part of the trust formula.

Scenario 1. Alice, a student 26 years old, post her locations and events, but she is suspicious. Therefore, she wants to share with users that she trusts them and at the same age and same university. The rule for this scenario is as follows:

S1: <R = {Alice}, {obj1}, {display}, {(trustAtt > 0.7) ∧ (ageLevel =myAgeLevel) ∧ (education =myEducation)}>.

Table 3 shows an example of two users with different social trust values, which leads to different access results in scenario1.

Table 3. An example of using scenario 1

Username	TrustAtt (including gossiping value)	GossipingAtt	AgeLevelAtt	Education	Result
UserA	0.6	0.5	27	Harvard U	Deny access
UserB	0.8	0.8	28	Harvard U	Allow access

Use case 2. The user defines a rule which includes gossipingAtt, and more attributes.

Scenario 2. Bob, a political man in the United States that doesn't care who will see his posts in his country. He wants to get the most likes and comments; therefore, he shares his post with gossiping users. The rule for this scenario is as follows:

S2: <R = {Bob}, {obj2}, {display, comment, like}, {(gossipingAtt < 0.7) ∧ (friendTypeCountry = mycountry)} >

Table 4 shows an example of two users with different gossiping values, which leads to different access results in scenario2.

Table 4. An example of using scenario 2

Username	gossipingAtt	friendTypeCountry	Result
UserA	0.5	USA	Allow access
UserB	0.8	USA	Deny access

Use case 3. The user defines a rule which includes trustAtt, gossipingAtt, and more attributes.

Scenario 3. Carlos, a COO of "FX" high-tech company, doesn't want to share his locations (check-in posts and event posts) with untrusted nor gossiping users but to share with employees who work with him. The rule for this scenario is as follows:

S3: <R = {Carlos}, {obj3}, {display}, {(gossipingLevel > 0.7) ∧ (trustLevel > 0.7) ^ (friendTypeWork = mywork)} >

Table 5 shows an example of two users with different gossiping values and social trust value, which leads to different access results in scenario3.

Table 5. An example of using scenario 3

Username	TrustAtt (without gossiping value)	gossipingAtt	friendTypeWork	Result
UserA	0.7	0.5	FX	Deny access
UserB	0.9	0.8	FX	Allow access

Use case 4. The user defines a rule which includes trustAtt and few attributes that are part of the trust formula.

Scenario 4. David, a seller in the Facebook marketplace, would like to share his new items for selling with trusted users and popular users that will share his items. The rule for this scenario is as follows:

S4: <R = {David}, {obj4}, {display, like, comment, share}, {(TrustAtt > 0.7) ∧ (numOfFriends > 300)} >

Table 6 shows an example of two users with different social trust values and a number of friend's values, which leads to different access results in scenario4.

Table 6. An example of using scenario 4

Username	TrustAtt (including gossiping value, without numOfFriends value)	numOfFriends	Result
UserA	0.55	85	Deny access
UserB	0.75	350	Allow access

Use case 5. The user defines a rule which includes simple attributes

Scenario 5. Erin, a teenager 22 old, looks for a girlfriend and wants all the girls in his city to see and like his post. The rule for this scenario is as follows:

S5: <R = {Erin}, {obj5}, {display, like}, {(friendTypeCity = myCity) ∧ (FamilyStatus =Single) ∧ (ageLevel =myAgeLevel)} >

Table 7 shows an example of two users with different city attribute values, which leads to different access results in scenario5.

Table 7. An example of using scenario 5

Username	City	Family status	AgeLevelAtt	Result
UserA	Palo Alto	Single	28	Deny access
UserB	San Francisco	Single	22	Allow access

5 Conclusions and future work

5.1 Conclusion

In this paper, we have presented a new Access-Control model for an online social network. Our ABAC-TG model's novelty is its combination of user-attributes that includes trust and gossiping values based on real online social network characteristics.

The algorithm for computing the trust attribute is based on [5] but enables the addition and removal of attributes in its formulation based on what appears in the ABAC rules. We described a new algorithm for calculating the gossiping value uses graph clustering, and this value may be either included in the trust calculation or treated separately in the ABAC rule. This makes the model very flexible and adaptive. The attributes of this model were carefully picked, but there could be flexibility in these choices and their values that are debatable. This model can help to make important permission decisions and prevent unwanted information leakage from users, making online social network privacy better in many ways.

Our ABAC-TG model is dynamic and extensible and can be used for any social network that would like to enable users to choose who will see their data. We decided to demonstrate the new model by five different use cases on Facebook, as it’s the world's most popular social network.

5.2 Future work

In future work, we intend to continue exploring in several directions.

First, we plan to conduct an extensive evaluation experiment. We like to evaluate the new ABAC model with Trust and Gossiping on a Facebook DB, as it’s the world's most popular social network. We plan to build a database that includes users, objects, and user actions and attributes. These items will be extracted from a real Facebook network of at least 100 users. We plan to do three experiments, in one we let the users define their own rules using our model and get their feedback on its usability. In the second, we'll set ourselves rules and threshold values and compare our model results with the user’s perceptions and expectations. Third, we like to evaluate the new gossiping algorithms on the same Facebook DB.

Second, we like to define an anonymity mechanism for Facebook objects for protecting shared objects by using summarization, filtering, blurring, and other techniques. Recently, initial work on this was published in [15]. We plan to extend it in several ways and integrate it with the ABAC-TG model. For example – define a filtering model – for sharing entities. The filter model will anonymize the data. For example, hide the username, hide the age and display range of ages, hide the gender, location, etc.

This mechanism will be part of the ABAC attribute definition by the user. For instance, this action is relevant for the “sharing" action on Facebook. Today the user adds a post, and his friends can share it, and the original user doesn’t know who will see his post. Therefore, we would like to protect the objects, so that the other users will not be able to see the full object, but only part of it (the anonymization result).

References

1. Hu, V.C., et al.: Guide to Attribute Based Access Control (ABAC) Definition and Considerations. NIST Spec. Publ 800, 162(2014).
2. Vishwas T. Patil, R. K. Shyamasundar: Undoing of Privacy Policies on Facebook. Proceedings of DBSec 2017: 239-255.
3. Tamir Lavi, Ehud Gudes: Trust-based Dynamic RBAC. ICISSP 2016: 317-324.
4. Wanita Sherchan, Surya Nepal, Cécile Paris: A survey of trust in social networks. ACM Comput. Surv. 45(4): 47:1-47:33 (2013).
5. Nadav Voloch, Priel Levy, Mor Elmakies, Ehud Gudes: An Access Control Model for Data Security in Online Social Networks Based on Role and User Credibility. CSCML 2019: 156-168.
6. Nina Mishra, Robert Schreiber, Isabelle Stanton, Robert Endre Tarjan: Clustering Social Networks. WAW 2007: 56-67.
7. Abhijit Banerjee, Arun Chandrasekhar, Esther Duflo, Matthew O. Jackson: Gossip: Identifying Central Individuals in a Social Network. CoRR abs/1406.2293 (2014).
8. Vishwas T. Patil, Nivia Jatain, R. K. Shyamasundar: Role of Apps in Undoing of Privacy Policies on Facebook. Proceedings of DBSec 2018: 85-98.
9. Abhijit banerjee, Arun G. Chandrasekhar, Esther Duflos, and Mattew O. Jackoson: Using gossips to spread information: theory and evidence from two randomized controlled trial, Review of EconomicStudies, 2016.
10. Z. Zhang, L. Han, C. Li, J. Wang: A novel attribute-based access control model or multimedia social netwroks. Neural Network World 26(6):543-557, 2016.
11. Jun Pang, Yang Zhang:A new access control scheme for Facebook-style social networks. Comput. Secur. 54: 44-59 (2015).
12. Nurit Gal-Oz, Ran Yahalom, Ehud Gudes: Identifying Knots of Trust in Virtual Communities. Proceedings of IFIPTM 2011: 67-81.
13. Allison K. Shaw, Milena Tsvetkova, Roozbeh Daneshvar: The effect of gossip on social networks. Complex. 16(4): 39-47 (2011).
14. Nadav Voloch, Ehud Gudes: An MST-based information flow model for security in Online Social Networks. Proceedings of ICUFN 2019: 460-465.
15. Nadav Voloch, Priel Nissim, Mor Elmakies, Ehud Gudes: A Role and Trust Access Control Model for Preserving Privacy and Image Anonymization in Social Networks. Proceedings of IFIPTM 2019: 19-27.