Dynamic Rights Reallocation in Social Networks

A. Ahmad[1], B. Whitworth[1] and L. Janczewski[2]

[1]Massey University, Auckland, New Zealand
[2]The University of Auckland, Auckland, New Zealand
e-mail : {A.Ahmad, B.Whitworth}@massey.ac.nz; Lech@auckland.ac.nz

Abstract

Access control, as part of every software system, has evolved as computing has evolved. Its original aim was to limit unauthorized access to centralized systems, but the rise of social networks like Facebook has changed that. Now each person wants to control who sees photos or makes comments on their local wall by making and unmaking friends, i.e. dynamic, distributed rights control. Social networks already have access control, but there is currently no agreed logical model for their rights, no consistent scheme for allocating and re-allocating permissions to create, edit, delete and view social objects and entities. A socio-technical approach based on social and technical requirements can give the basics of a model. Various rights reallocations like multiply, divide, transfer and delegate are explored. It suggests a theoretical base for access control beyond its security parent.

Keywords

Social Networks, Rights analysis, Rights reallocation

1 Introduction

The need for access control arose with multi-user computing, as users sharing the same system came into conflict (Karp *et al.* 2009). As computing evolved, access control logic developed to offer domain access control for distributed systems and roles for systems with many users. With variations, the traditional access control approach has worked for military and commercial applications, organizational structures, contextual decisions, distributed applications, medical data, peer-to-peer networks and the grid environment (Lampson, 1969; TCSEC, 1985; Clark and Wilson, 1987; Ferraiolo and Kuhn, 1992).

The last decade has seen extreme multi-user systems emerge – social networks (SNs) where millions of users share billions of resources and grant each other access rights (Carminati *et al.* 2008). As access control now depends on the number of interactions, its complexity increases geometrically with size, not linearly. Mapping millions of subjects directly to billions of resources is unwise, as each account adds hundreds or thousands of photos and comments a year. The world population is at seven billion and growing, if Facebook's current 800 million active accounts is just the beginning, traditional access methods may be ending their useful life.

As social networks are here to stay, and growing in number and size, a logical model of distributed rights reallocation is needed. This includes rights multiplication,

division, transfer and delegation. The aim is to identify software patterns that embody social principles as well as technical principles like efficiency (Ahmad and Whitworth, 2011). The result would be a consistent scheme to allocate and reallocate distributed rights in a socially acceptable way. The rest of the paper is organized as follows: Section 2 reviews previous work, Section 3 gives the specifications, Section 4 presents the model, and Section 5 concludes it.

2 Review

Carminati et al. presented a semi-decentralized access control model for social networks (Carminati *et al.* 2008), where users are categorized in terms of relationship type, depth and trust level. Likewise dRBAC manages trust in coalition environments by decentralized access control (Freudenthal *et al.* 2002). Additionally, some other access control solutions use trust (Ali *et al.* 2007), reputation (Carminati *et al.* 2006) and relationships (Tapiador *et al.* 2011) to manage access rights between users. However, there is no access control model for SN that supports rights reallocation.

There are delegation models for traditional access control, but other types of right reallocation, like multiplying and dividing, have received little attention. Existing delegation models can be categorized into machine to machine – one object acting on other's behalf (Varadharajan *et al.* 1991), user to machine – objects acting on user behalf (Gasser and McDermott, 1990), and user to user role delegation – user assigning roles to other users (Barka and Sandhu, 2000).

Traditional models cannot be mapped to current SN for the following reasons:

1. Traditional solutions do not give local control over user contributions like family photos, and so struggle with privacy demands. Central access control gives each user the same policy, so variants must be requested from a central authority who sets system wide roles. The user has no local control over their resources as friends are based on generic roles.

2. The delegation models presented in literature are based on system wide entities and are hard to apply on SN local autonomous domains, where domain based delegation is required rather than role based. Current models provide single user, multi-level delegation, but SN require multiple user, single level delegations to maintain domain accountability.

3. Current access control models for SN do not specify the dynamic reallocation of distributed rights found in social networks (Carminati *et al.* 2008; Simpson, 2008), where everyone can give rights away. In dynamic, distributed control, each person can fully administer their own domain.

The above expands on previous work aimed at developing a general and logical rights framework for online social interactions (Ahmad and Whitworth, 2011; Whitworth, deMoor and Liu, 2006; Whitworth and deMoor, 2003). It addresses

rights reallocation because in social networks friends are regularly made and unmade, i.e. managing rights allocations is a critical success criterion.

3 Specifications

A socio-technical system is a social system on a technical base, as a socio-physical system is a social system on a physical base. Socio-technical design involves technical and social requirements, to model not just what can be done but what should be done.

3.1 Overview

An information system has *entities* and *operations*, where:

1. ***Entity***. Stored as static information, with properties.
    a. *Actor*. An entity that can participate in a social interaction.
        i. *Persona*. Represents an accountable offline person or group.
        ii. *Group*. A set of personae acting as one.
        iii. *Agent*. An actor that represents another actor.
    b. *Object*. Conveys information and meaning.
        i. *Item*. A simple object with no dependents, e.g. a bulletin board post.
        ii. *Space*. A complex object with dependents, e.g. a bulletin board.
    c. *Right*. A system permission for an actor to operate on an entity.
        i. *Simple rights*. Rights to act on object or actor entities.
        ii. *Meta-rights*. Rights to act on right entities, e.g. transfer.
        iii. *Role*. A variable right (a set of rights).
2. ***Operations***. Stored as a program or method that processes entities.
    a. *Null operations* don't change the target entity, e.g. view, enter.
    b. *Use operations* change the target in some way, e.g. edit, create.
    c. *Communication operations* transfer data from sender(s) to receiver(s), e.g. send.
    d. *Social operations* change a right or role, e.g. delegate.

3.2 Reallocating Rights

The ability to reallocate social rights is the key to meeting social requirements. It allows socio-technical systems to evolve from an initial state of one administrator with all rights to a community with delegated rights. Re-allocation can change the actors in a right or role as follows:

1. *Transfer.* Allocate use and meta-rights and is irrevocable.

2. *Delegate.* Allocate use rights only and is revocable.

3. *Divide.* Allocate rights jointly to an actor set.

4. *Multiply.* Allocate rights severally to an actor set.

If a right is owned *jointly,* all must agree to allow the act, while if it is owned *severally,* any party alone can activate it. The above can act in combination, e.g. to transfer joint ownership. Table 1 shows the details, as follows:

1. *Transfer*. Transfer gives all entity rights, including meta-rights (Gaaloul *et al.* 2010). Rights are irrevocably given to the new owner, e.g. after selling a house, the old owner has no rights to it.

2. *Delegate.* Delegate gives use rights but not meta-rights, so can be taken back, e.g. a system administrator who delegates rights can take back the top system priority (Gaaloul *et al.* 2010).

3. *Divide.* Those who divide ownership jointly own an entity, e.g. a couple who jointly own a house must both agree to sell it. In joint ownership, any party can stop an act.

4. *Multiply.* In multiply, the entire right is given completely, so any party can act alone as if they owned it exclusively, e.g. a couple's bank account where both can withdraw all the money.

	Allocated by (Actor)		*Allocated to (Actor)*	
	Meta	Use	Meta	Use
Transfer			√	√
Delegate	√			√
Divide	√	½√		½√
Divide all	½√	½√	½√	½√
Multiply	√	√		√
Multiply	√	√	√	√

Table 1: Allocating use and meta-rights

For example, a many author paper submitted online can let *one* author alone edit it (transfer), let *one* author edit as allowed by the primary author (delegate), let edits proceed only if confirmed by *all* authors (divide), or let *any* author do any edit (multiply). The model covers all these social options.

If a delegatee gets no meta-rights, they can't pass rights on, e.g. renting an apartment gives no right to sublet. Similarly, lending a book to another doesn't give them the right to on-lend it, though as with all social requirements, it happens. Yet being consistent maintains accountability, e.g. if one loans a book to a person who loans it to another person who then loses it, who is accountable to the original owner? This gives the operational principle:

P1. Delegating doesn't give the right to delegate.

A right reallocation is *revocable* if the initiating party keeps the meta-rights, so delegation is revocable but transfer is not. Dividing use rights is revocable but dividing all rights is not, as reverting would require joint agreement. Multiplying use rights is revocable but multiplying meta-rights is a dictator's dream case as anyone can allocate all rights to anyone, which is likely unstable.

To allocate a right to an existing object makes one accountable for it, so by fairness requires consent, e.g. one doesn't add a paper co-author without their agreement. The principle is:

P2. Allocating use rights to existing objects requires consent.

One can't make someone the owner of something unless they agree. The access control system would have to put a question like: "*Martin wants to transfer edit rights over xyz to you, do you agree?*" In contrast, rights with no accountability for existing objects can be allocated without permission, as the other can use them if they wish, e.g. view and enter. The principle is:

P3. Rights that imply no existing objects or null acts can be allocated freely.

So space owners can delegate entry and view rights without inconsistency. These are social requirements not technical necessities. As technical requirements express technical good practice, so social requirements express social good practice. For best effect, they should be applied consistently.

3.3 Social networks

The model both clarifies how social networks operate and suggests alternatives, e.g. social networks send messages like:

"*X wants to be friend with you*"

In this model, it is a social trade: X will add you to their friend role if you add them to yours. It can be handled as a two-step social transaction, but the steps need not be linked. A tit-for-tat is assumed, but one can *befriend* another, i.e. add them to a friend

role, without their permission (P3). One could make another a friend, with view rights, whether they return the favor or not. So, one could receive messages like:

"*X has made you a friend* "

This is an offer to *be a friend,* not a request to *be my friend*. As one can love another who doesn't return the favor, so friendship needn't be mutual. Systems that axiomatize friendship as always a duality limit it, as friendship is given as well as received.

4 The formal model

An access control matrix can be expressed using function Grant-Right (A, O, R) which holds whenever the access control matrix gives right R to actor A over object O. So function of the form

Grant-Right (Alice, abc.txt, View)… (i)

states that Alice can view the abc.txt file. This kind of simple function can also be used to assign various rights to roles instead of individual actors, e.g. the function to give edit right to family role over abc.txt will look like

Grant-Right (Family, abc.txt, Edit)… (ii)

Also, some rights can contain others as their subset, e.g. allocating an edit right implies a view right to the same actor:

Grant-Right (A, O, Edit) ⊨ *Grant-Right (A, O, View)… (iii)*

Apart from the basic Grant-Right function, which is a nice way to represent the rights stored in the access control matrix, access control logics also include formulae of the form A says Ω, where A is an actor and Ω is a right statement (Genovese *et al.* 2010). The formula represents that actor A makes statement Ω, which can be a request, assignment of rights to some actor or role, or as a part of the security policy, e.g. the owner of an object grants a friend the right to view it, this assertion can be represented as:

Owner says Grant-Right (Friend, Object, View)… (iv)

The precondition of using this say function is that the authorizing actor holds the meta-right to the right that is given away. In the above statement, the owner is by definition the person with meta-rights to that object, so can re-allocate its rights.

In a typical SN instance, suppose Alice sets David as family, George as a friend and Bob as a colleague, where David has a friend Harry and George has a friend Frank. If Alice posts a photo collage of her family on her wall (space) as entity O1, she may wish to let David view and edit (add a photo to it), George to view it, but to not let Bob see it at all. This can then be done by granting those rights to those roles. Yet David might post his family photos O2 on his wall but not to let his friends view it, as his domain has different rules from Alice. This can be modelled in a consistent way, as shown in Table 2.

This model lets Alice delegate or transfer *view* rights to her family photo. If she delegates view to her friend George, he can't show it to his friend Frank (by P1), but if she transfers view to David, then David can show it to Harry. So delegate means you can't pass it on.

If Alice delegates *edit* rights to David, only he can add photos. If she multiplies edit rights to him, both her and David can add photos. If she divides the rights, photos can only be added if they both approve. If Alice transfers edits rights to David, he can on-delegate, to let his friend Harry add a photo.

Alice can delegate, divide or multiply *delete* rights, but to transfer them would be to give up ownership of her space, e.g. another could post a picture she disapproves of which she could not delete. If the right to delete, or at least to say who can delete, is basic to the ownership of any local domain space, it can't be given away without giving away the space.

This access control model not only supports existing rights, as granted by systems like Facebook, but also suggests new ones, e.g. to let SN actors set their role allocations different from the general template, or create new local roles like colleague. Local domain control involves decentralizing meta-rights, not just rights.

The proposed framework can handle this because it doesn't distinguish the administrative authority. The condition formula *A says* Ω can be used by many as well as by one, as both *A* and Ω are arbitrary. In this approach:

Access Control Model Instance *

Subject (type : subject, identifier : number, name : string)	*User (subject, number, user)*
Right (type : right, identifier : number, name : string)	*Role (subject, number, role)*
Object (type : object, identifier : number, name : string)	
User (subject U, #1, Alice)	*User (subject U, #2, Bob)*
User (subject U, #3, David)	*User (subject U, #4, George)*
Object (object O, #5, O1)	
Right (right R, #6, View)	*Right (right R, #7, Edit)*
UserRoles (subject UR, #8, Family)	*UserRoles (subject UR, #9, Friend)*
UserRoles (subject UR, #10, Colleague)	
ActiveRole (subject AR, ar#11, U : #3 (David), R: #8 (Family))	
ActiveRole (subject AR, ar#12, U : #4 (George), R: #9 (Friend))	
ActiveRole (subject AR, ar#13, U : #2 (Bob), R: #10 (Colleague))	
Auth (S : ar#11 (Family), O : #5(O1), R : #6 (view), Υ: +)	*subject (U, #3, David)*
Auth (S : ar#11 (Family), O : #5(O1), R : #7 (edit), Υ: +)	*subject (U, #3, David)*
Auth (S : ar#12 (Friend), O : #5(O1), R : #6 (view), Υ: +)	*subject (U, #4, George)*

Authorization Set

(ar11, O1, View)	*(UR: #8, O : #5, R: #6, Υ:+)*	*(ar11, O1, Edit)*	*(UR: #8, O : #5, R: #7, Υ:+)*
(ar12, O1, View)	*(UR: #9, O : #5, R: #6, Υ:+)*		
(David, O1, View)	*(U: #3, O : #5, R: #6, Υ:+)*	*(David, O1, Edit)*	*(U: #3, O : #5, R: #7, Υ:+)*
(George, O1, Edit)	*(U: #4, O : #5, R: #6, Υ:+)*		

**Owner has all the rights over the object so owner role is not considered*

Table 2: Access control model instance and its authorization set

1. *Transferring* changes the actor property of all rights to an object, including meta-rights. It changes the owner, as formalized by

   OldOwner says Grant-Right (NewOwner, Entity, AllRights) ... (v)

   By this access matrix change, the *NewOwner* has all rights to the object and the *OldOwner* has none. The *says* method is the means by which that occurs. As it is one atomic operation, the rights of the *OldOwner* and the *NewOwner* cannot conflict.

2. *Delegating* a right reallocates all rights except the meta-rights. Again, the *says* operation changes object rights from the delagator to the delegatee in one atomic step, so at each point it is clear who is responsible for acting upon it. In this case, the delegatee is responsible for the object, but the delagator is responsible for the delegatee, and can revoke their permission at any time.

3. *Multiplying* a right replaces its actor by an OR set, where the formula $A \vee B$ *says* Ω to mean that principal A or B says Ω. This explicitly lets any actor execute the given operation on the defined object alone.

4. *Dividing* a right replaces the actor by an AND set, where the formula $A \wedge B$ *says* Ω to mean that principal A and B jointly says Ω. This would require the consent of both A and B to execute a function Ω, where Ω can be any arbitrary operation legal in the settings of access control model instance.

Note that to automatically make the friends of my friends also my friends is to not recognize the difference between delegate and transfer rights re-allocations. It contradicts P1, that giving a use right doesn't give the meta-right. The attempt to make friends of friends also friends illustrates a technical option that failed because it had no social basis. Designing social interactions without regard to social requirements is how we got the current spam problem (Whitworth and Liu, 2009). It is a social fact that liking someone doesn't guarantee that one will like their friends. In the above, if a role delegates a right, it can't be passed on.

5 Conclusions

This paper suggests how to allocate and reallocate access control rights to satisfy social requirements like local ownership. It arises because those who add photos of their family won't add them if they can't control them. The semantics of this model satisfies social requirements P 1-3 and its formal syntax is consistent (Table 2). It not only defines what SNs like Facebook currently do, and what they should not, but also suggests new options not yet tried (Table 1).

Access control began in the shadow of security but socio-technology will make it a new discipline. While security needs secrecy for obvious reasons, access control today is more about access than control. It is about getting participation rather than

stopping it, as a socio-technical system without an active community isn't a socio-technical system at all.

The next phase of this project is to develop a distributed, dynamic access control plug-in for a NSF granted open knowledge exchange (OKE) system and evaluate it with respect to both social criteria like fairness and technical criteria like storage efficiency. This "rights module" will also give human readable reports to actors of granted rights, i.e. be transparent. The goal is that social rights are not only applied but also seen to be applied, as this is critical for trust and synergy.

Online communities today can't survive without participation, so access control is increasingly about letting people in rather than keeping them out. This model follows the socio-technical paradigm: to first define the social requirements then design a technical solution to meet them. It tries to avoid social errors like the unfairness of spam but still addresses technical requirements like consistency and efficiency. The evolution of access control to include social requirements goes beyond the traditional physical security focus, into new research dimensions.

6 Acknowledgment

This work has been sponsored by National Science Foundation (NSF), USA, under award number 0968445. “OKES: An open knowledge exchange system to promote meta-disciplinary collaboration based on socio-technical principles”.

7 References

Ahmad, A. and Whitworth, B. (2011), "Distributed Access Control for Social Networks", *International conference of information assurance and security IAS’11*.

Ali, B., Villegas, W., and Maheswaran, M. (2007), ”A trust based approach for protecting user data in social networks”. *Conference of the Center for Advanced Studies on Collaborative research (CASCON’07),* pages 288–293.

Barka, E. and Sandhu, R. S. (2000), "Framework for role-based delegation models", *16th Annual Computer Security Applications Conference (ACSAC 2000)* New Orleans, La. Dec. 11–15). IEEE Computer Society Press, Los Alamitos, Calif., 168–177.

Carminati, B., Ferrari, E., and Perego, A. (2006), ”Rule-based access control for social networks”. *On the Move to Meaningful Internet Systems: OTM Workshops*.

Carminati, B., Ferrari, E. and Perego, A. (2008), “Enforcing access control in web-based social networks” *ACM Transactions on Information & System Security.*

Clark, D. D., and Wilson, D. R. (1987), “A Comparison of Commercial and Military Computer Security Policies,” *IEEE Symposium of Security and Privacy*, pp. 184–194.

Ferraiolo, D. and Kuhn, D. R. (1992), “Role-Based Access Control,” *NIST-NSA National (USA) Computer Security Conference,* pp. 554–563.

Freudenthal, E., Pesin, T., Port, L., Keenan, E., Karamcheti, V. (2002), "dRBAC: Distributed role based access control for dynamic coalition environments" In ICDCS '02: *22nd International Conference on Distributed Computing Systems (ICDCS'02)*.

Gaaloul, K., Zahoor, E., Charoy, F., and Godart, C. (2010) "Dynamic Authorisation Policies for Event-based Task Delegation", *Advanced Information Systems Engineering, 22nd International Conference, CAiSE,* Hammamet, Tunisia.

Gasser, M., and McDermott, E. (1990), "An Architecture for practical Delegation in a Distributed System". *IEEE Computer Society Symposium on Research in Security and Privacy.* Oakland, CA.

Genovese, V., Giordano, L., Gliozzi, V., Pozzato, G. L. (2010), "A constructive conditional logic for access control: a preliminary report". *19th European Conference on Artificial Intelligence*. pp.1073~1074.

Karp, A. H., Haury, H. and Davis, M. H. (2009), "From ABAC to ZBAC: The Evolution of access control models", *Technical Report HPL-2009-30,* HP Labs.

Lampson, B. W. (1969), "Dynamic Protection Structures," *AFIPS Conference Proceedings*, 35, pp. 27–38.

Simpson, A. (2008), "On the need for user-defined fine-grained access control policies for social networking applications," In SOSOC '08: *Workshop on Security in Opportunistic and social networks*, New York, USA, 2008.

Tapiador, A., Carrera, D. and Salvachúa, J. (2011), "Tie-RBAC: an application of RBAC to Social Networks". *Web 2.0 Security and Privacy*, Oakland, California.

TCSEC, Trusted Computer Security Evaluation Criteria (TCSEC) (1985), *DOD 5200.28-STD*. Department of Defense.

Varadharajan, V., Allen, P. and Black, S. (1991), "An Analysis of the Proxy Problem in Distributed systems". *IEEE Symposium on Research in Security and Privacy.* Oakland, CA.

Whitworth, B., and deMoor, A. (2003), "Legitimate by design: Towards trusted virtual community environments". *Behaviour and Information Technology Journal,* 22:1, p31-51.

Whitworth, B., deMoor, A. and Liu, T. (2006), "Towards a Theory of Online Social Rights", in R. Meersman, Z. Tari, P. Herrero et al. (Eds.): *OTM Workshops,* LNCS 4277, pp. 247 – 256, Springer-Verlag Berlin Heidelberg.

Whitworth, B. and Liu, T. (2009), Channel email: Evaluating social communication efficiency, *IEEE Computer,* July, p63-72.