Distributed Access Control For Social Networks

Adnan Ahmad, Brian Whitworth
Institute of Information and Mathematical Sciences
Massey University
Auckland, New Zealand
[A.Ahmad, B.Whitworth]@massey.ac.nz

***Abstract*—Access control is the process by which access to information is granted to users for certain actions based on their identity. Traditional access control models that map every system resource directly to every system user work for organizations with thousands of users but struggle for social network sites like Facebook with millions of users. The problems faced are firstly the technical complexity of mapping millions of users to billions of resources and secondly the social need of users to own the items they post and to control their access, so access policies beyond just public/private are needed. And finally, that if ordinary users are to manage their own access control, they need software support. This paper argues that only distributed access control can meet these challenges and proposes a model based on the socio-technical design paradigm: first define the social requirements then design a technical solution to fulfill them.**

Keywords: access control; complexity; social networking; socio-technical;

I. INTRODUCTION

Social networking sites (SNS) are online platforms where users form relationships with others by sharing resources (like photos and comments) [1]. People use these sites for entertainment, dating or business purposes, and their defining characteristic is users connecting to others to access their posted information. This paper proposes a model to meet both the opportunities and challenges social networks create for access control.

Access control is the process by which authorized users are granted permissions to act upon resources in a computer-based information system. It is pervasive to every computer system, but its role has changed as systems have evolved. Its traditional security aim was to prevent unauthorized user access to the system as a whole, to avoid system failure or data theft. However today, individual user security and privacy has become an issue, so the effect of a SNS 'error' is public outrage not technical failure. Software errors have given way to social "errors", but the effect is the same - if no one uses a site it is just as bad as if the software hangs – as in both cases, the code does not run. In a SNS, access control is critical, as even a single mishandling of access allocation can cause an entire community to turn against the site. This is because an individual's SNS privacy settings can make them the victim of targeted marketing, money scams or even personal attacks, e.g. a 2010 Australian investigation linked a girl's murder to her Facebook privacy settings [2]. Systems that offer only basic access controls, like completely private or completely public, are struggling to meet the social needs of users [1]. The challenges that traditional access control faces in SNS can be summarized as follows:

A. Interaction complexity

If traditional access control is about who can enter a system, it grows linearly with the number of users, but if SNS access control is about who can connect to who, the relationship combinations, potential access permissions increase geometrically as a function of the number of users. So models that map system resources directly to subjects soon become over-complex, over-complex for an SNS with millions of users and billions of resources, e.g. Facebook reports 800 million active user accounts each adding many hundreds of photos and comments each year. The challenge of managing the access rights of so many users' contributions affects access control efficiency and processing spent unnecessarily on access control affects use factors like response delay.

B. Local control of relationships

Connecting to others satisfies relationship needs but also raises security and privacy concerns [3]. As users contribute SNS content, like family or friend photos, they naturally expect to control that content. In centralized access control, each user is allocated the same access control policy values, so variants must be requested from a central authority. Roles in traditional access control systems are system wide groups whose membership is set by a system administrator. The access rights over user's resource are allocated to a role the user has no control over. In contrast, users on SNS want to set their own values without reference to a central authority, e.g. to share their data with everyone or to restrict it to family and friends. Traditional access control systems struggle with the demand of today's individual user for a diverse range of privacy requirements. Currently, SNS users can make personal information available to set roles, like friends or 'friend of friend', but can't define their own local roles, e.g. to distinguish acquaintance "friends" from buddies, or close from distant family. Generic roles tend to reveal more than the user wants, as users can't specify local requirements using generic roles.

Currently, different SNS use different access control approaches to address these challenges. However, these solutions, that connect users individually to objects, have complexity issues, are platform dependent and lack some generic access control solution for SNS.

This paper now proposes that although SNS technology is new, people networking with people is not, so it makes sense to use the social "inventions" society has evolved over thousands of years to manage social interactions, i.e. let social ideas like creator ownership and relationships drive system design as well as technical principles like efficiency [11]. Only this will close the socio-technical gap between what online communities want and what technology does.

The remainder of this paper is organized as follows. Section II briefly outlines related work in SNS access control, section III discusses the SNS access control requirements, section IV presents the proposed model and section V theoretically assesses it.

II. RELATED WORK

Traditional access control models can be categorized into discretionary access control (DAC), mandatory access control (MAC) and role-based access control (RBAC). DAC [4] systems assume that objects belong to owners who can manage their access, while MAC [4] and RBAC [5] assume a central trusted computing authority to support an organization security policy. As ownership is fundamental in SNS, DAC driven models are the only choice, however system wide groups and their centralized management in DAC don't suit the SNS environment. RBAC is centralized in nature and has proved to be quite expensive to implement ownership [19].

Past SNS security research has mainly focused on statistical analysis techniques while preserving members' privacy [6]. Some relatively new approaches to SNS access management are based on trust and relationships, e.g. D-FOAF (Friend of a Friend) is an ontology based distributed identity management system for SNS that manages access rights in terms of trust level and path length between two users [7]. In another SNS access control approach, centralized trust management is used to determine the security level of users and resources [8]. Such solutions do not address the SNS complexity problem raised above.

A semi-decentralized access control model is presented in [9] where users are categorized in terms of relationship depth and trust level, and dRBAC [10] manages trust in coalition environments by decentralized access control. Such solutions view the requirements of SNS users through a security lens, so do not address the local control of friends problem raised above.

In current SNS access control systems, complexity remains an issue and support for local control of relationships is questionable. This paper proposes an SNS access control model based on the socio-technical paradigm, by first defining the social requirements then designing a technical solution to meet them.

III. ACCESS CONTROL REQUIREMENTS

Socio-technical systems in general involve a community of largely equal users making contributions [12]. They differ from simple technical systems in having to satisfy the social demands of a community as well. If they do not, then people will not contribute, and the system will fails because it has no users community. Socio-technical systems fail not only by hardware and software errors, but also by social errors.

Fig. 1 shows a simple social network where nodes are network users and lines are the relations between them, which are of different types, e.g. A(lice) sees G(reg) and F(rank) as friends, D(avid) and E(ric) as family, and B(ob) and C(arl) as colleagues. Also, D(avid) sees H(arry) as a friend and B(ob) sees I(an) as a friend.

In social terms, if Alice *owns* say a photo of her, only she should be authorized to manage its access. If she then shares it based on her social connections, it is not necessary to map that resource to any users beyond those she knows, i.e. her social circle. Also, as in physical communities, social relationships have degrees of closeness, i.e. people relate not only to their best buddies but also to family, coworkers, teachers and acquaintances on SNS. Each type of relationship differs in whether a user wants to share a status, activity or photo album. The social requirement is for the system to let users share resources based on relationship type or closeness. In contrast, most previous work on access control for SNS focuses on level one (Friends) and level two (friends of a friend) relations [e.g., 1, 7, 9]. The *social requirements* of SNS access control can be outlined as follows:

A. Ownership

In physical society people see objects in terms of who owns them. In particular it is considered right that one should own what one creates, e.g. a painting or poem, a principle first established by Locke [13]. In an SNS, people create information objects by posting them, so in a socio-technical design they would own them, i.e. be able to manage their access control. Essentially the privacy requirements of SNS are that if people cannot control access

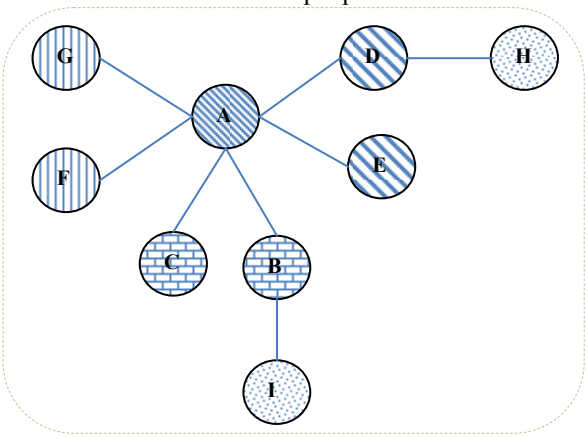

Figure 1. A simple social network

to what they post, they will not post, and so the system will collapse. Socio-technical systems are built upon the social concept of ownership, i.e. every resource in them must be owned by at least one person. Ownership can also add complexity, as in physical society it can be sold, delegated and shared.

B. Local roles

In traditional access control systems roles are system wide with fixed access associations, so they manage relationships in flat order without any closeness hierarchy. In contrast, the social requirement is for every SNS user to define their own local roles, e.g. a person may trust their friends more than their family and so give them more privileges. Roles and access rights should vary by local domain, i.e. let users define local roles for their domain to decentralize resource management. It reduces complexity and lets users manage access rights for their roles over their resources.

C. Local object classes

Currently, resources are managed independently regardless to their disclosure level, but SNS resources can be put into groups, like a photo album, that can then be given a privacy classification, e.g. to let only family view the family photo album. Creating object classes to define privacy levels reduces rights management complexity and increases usability.

D. Usability and flexibility

Local roles and object classifications give users both the simplicity of high control and the flexibility of low level control. In high level control, users link abstract roles to abstract object classes that make user sense, e.g. whether to let friends view the family photo album. In low level control, users can define a new role of one person or a new object class of one object, to define exactly who can see what, allowing a degree of fine grained access control not currently offered.

E. Delegating view rights

Current SNS lets users share information at level one or level two, i.e. with friends or friends of friends. In social terms, it is delegating view rights, e.g. does lending a book to a friend mean they can lend it to their friends? Or should they ask the lender if they want to further lend it to another? If delegating an object doesn't delegate the meta-right to delegate that object, then the social answer is that a delegatee cannot further delegate. So agreeing to level two viewing amounts to giving anyone who views your data the right to pass it on to anyone else. This is not required by SNS users, e.g. if Alice is friend with David and Bob, and wants to share a photo album with David's family but not with Bob's family, it is impossible in current settings. One may wish to connect to the families of two friends differently. Only access control based on local relations can give users the fine grained social control over resources they require.

IV. PROPOSED MODEL

A distributed access control model supported by a client-side enforcement mechanism must satisfy both social and technical requirements.

A. Definitions

Table 1 defines the constructs of the model.

TABLE 1. ABBREVIATIONS AND THEIR DEFINITIONS

	Definition
SH	*Stakeholder:* A user who posts system resource objects, e.g. photos, videos, comments or votes.
NS	*Namespace:* The set of objects a stakeholder creates.
VU	*Virtual user:* A user, from the social circle of stakeholder, seeking a NS resource access.
LR	*Local role:* A VU group with defined access to NS resources.
OC	*Object class:* An object group, based on security clearance, whose access is mapped to LRs.
AC	*Attestation certificates:* Permission objects encapsulated various access rights and map LR to OC objects.

B. Components

This model has the following components:

1) Namespace

Dividing the system into autonomous namespaces reduces processing costs and enhances user trust. It lets stakeholders control their own domain. They can classify virtual users by relationship and objects by who can see them. Mapping local roles to object classes manages access control.

2) Local roles:

The model allows local roles to restrict access to resources. A virtual user seeking access to a resource must acquire a role in the namespace with the requested access rights. Local roles have namespace wide scope and do not exist beyond that. They are dynamic, as different stakeholders can implement different roles and one virtual user can concurrently acquire various roles in multiple namespaces.

3) Object classes:

In this model, objects present in one namespace are classified according to their privacy clearance. They are first put in an object class, or container, and then the class is given a privacy clearance depending on its objects. Access mappings are between object classes and local roles, not between objects and users.

4) Mapping roles between namespaces:

If access control is between *VU* and *NS*, to give access to some *LR* from another *NS* needn't create a local name but can use the same list of *VU*. This introduces a flexible system for mapping access among multiple *LR* in multiple *NS*.

5) Use-Conditions:

Use-conditions are the conditions that a *VU* must satisfy to get access to a resource. These use-conditions (as a subset

of policy) are stored in the *NS* that only the *SH* has access rights to. The general use-conditions are:

- The requestor *VU* belongs to some *LR* in the *NS*.
- The requested object *O* is classified into an *OC*.
- The *LR* has an *AC* with access to that *OC*.
- The certificate defines an action, in this case view.

C. The approach

These features define an SNS access control model independent of the policy. Each *SH* manages its access control policy by allocating *VU*s to *LR*s with known access to *OC*s. No global administration is needed, as *SH*s administer their *NS* resources by mapping *LR*s to *OC*s.

The *VU* are not directly mapped to the resources rather the entry point to a *NS* is the abstraction of roles. All the *VU* in *SH NS* are assigned some *LR* and access to objects is granted on the basis of *LR* membership. Also, the objects *O* in *SH NS* are categorized into some security labeled *OC* with respect to their disclosure level. To introduce an additional security layer between *OC* and *LR*, the concept of attestation certificates *(AC)* is introduced similar to [14]. Every *LR* is assigned an *AC* and the access decision is made on the encapsulation of requested right in *AC* for the requested *OC* label. The system architecture of distributed access control model is illustrated in Fig. 2.

D. Security

While this access control model is driven by social rather than security aims, it must still satisfy the latter requirement. If SNS access control is managed by attestation certificates, how can a stakeholder ensure that a requestor has not maliciously forged one? So, a foolproof mechanism is needed to avoid forgeries. One way to do this is client-side distributed certificates. When Alice establishes a relationship with Bob, she creates a local certificate stating that Bob is a colleague of Alice. This certificate is locally stored in her *NS,* accessible to her only, so cannot be forged.

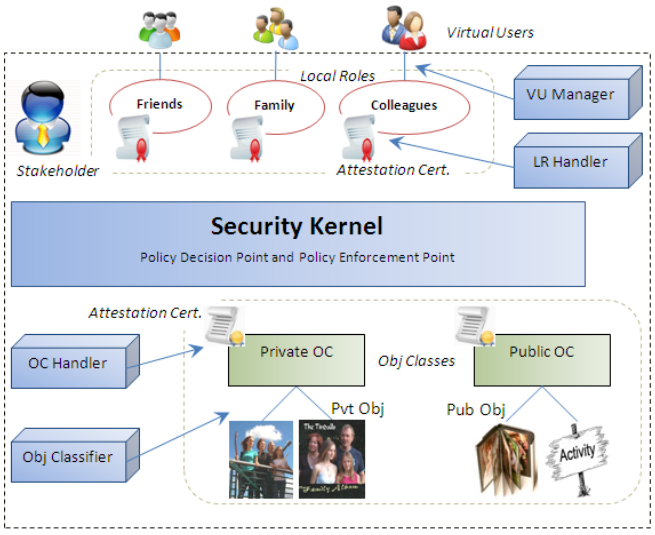

Figure 2. Distributed access control model system architecture

A similar but independent certificate is generated and stored by Bob as well. Note that the relationships are two way but access control permissions are one way only. Alice may consider Bob a friend but Bob may consider Alice just a colleague. When Bob requests a resource from her domain, Alice's access control module matches Bob's userid against her roles to decide the request outcome. Having one way certificates provides flexibility and also resolves the problem of certificate forgery. In contrast, centralized certificates need more processing and also expose a single point of failure.

Further it is liable to security attacks, in that the resource owner and the requestor both have certificates in the same repository and can access and modify them.

E. The model

An access control model is a state transition system {δ, γ, σ, Λ} where δ is a set of states, γ is a set of rights that include privileged requests considered by the system, σ is the entailment relation that determines whether a given right request is true or not in a given state, and Λ is the set of state-transition rules.

The goal of an implementation is to follow the scheme as closely as possible by supporting each global state which in turn made up of local states $(s_1, s_2 \ldots s_n)$, where all $s_i \in \delta_i$, the set of all possible states for domain *i*. All the rules in Λ for domain *i* should be testable with the relation σ_i. The implementation computes a function σ_i: $\delta_i \times \gamma \rightarrow$ {true, false}, where δ_i is the set of local states of domain *i*, and γ is the set of specific access requests being considered. In general, δ comprises of five different states: *Virtual users (VU)*, *Member (M)*, *Non-Member (Nm)*, *Allow (a)* and *Deny (d)*. σ has four set of functions, first is for mapping of *VU* to *M* or *Nm*, second is for mapping of *objects* to *OC*, third is for allocation of *AC* to *M* and *OC*, and fourth is for mapping of *LR* to *OC* to decide the outcome of request γ_i.

Λ comprises these rules for every namespace request :

- If a *VUid* is in a NS_i and maps to some LR_j, the *VU* state changes to M_i, else it becomes Nm_i.
- Object belongs to an object class under some label (default $L_1(\tau)$), i.e. $O \rightarrow OC_\tau$. Where, τ is the set of all security labels that are used for confidentiality levels. These labels are hierarchical and form a lattice under a partial order > such that $L_1 > L_2$ if and only if $L_2 \in L_1$.
- If *VU* is in *M* state and requests some object *O*, and there is some mapping of LR_i to OC_i then the request γ_i is allowed, else it is denied.
- If *VU* is in *Nm* state and requests some object *O*, then the request γ_i is always denied.

$$\sigma = \begin{cases} VU \rightarrow M/Nm \; \forall \; M \in \; LR \\ O \; \rightarrow \; OC_\tau \\ LR \; \rightarrow \; OC \; (\gamma_i) \\ M \rightarrow a/d_r \; \forall \; r \; \in \gamma \end{cases} \quad \ldots (i)$$

Given a social circle (namespace) *i* in SNS, an access condition *con* against NS_i is a tuple *(VU, LR, OC, AC)*, where $VU \in \sum_{i=1}^{N} M_{LR_i} \cup \{*\}$ is the requestor in domain *i*,

$obj \in OC_\tau$ is the object privacy clearance and $AC \in AC \cup \{*\}$ is the attestation certificate for *LR*. If *VU* =*, *VU* corresponds to any user in SNS but is not active in namespace *i*, whereas if *AC*=*, there is no mapping exist for LR_i to OC_τ.

To make the task of access decision easier, the equivalent formula translation can be expressed by the following assertion:

$$con(vu, obj, i) = isactive(vu, NS) \wedge hasLR(vu, i) \wedge hasOC(obj, i, \tau) \wedge hasAC(AC, LR) \ldots (ii)$$

F. Architecture

Traditionally, access control enforcement is done by a security kernel mechanism. A security kernel is a trusted software module that intercepts every access request call submitted to a system and decides if it should be granted or denied, based on some specified policy. Usually, a centralized approach is used for a security kernel, giving one *policy enforcement point* (PEP) and one *policy decision point* (PDP) to handle all resource requests. *PDP* manages the policies and evaluates them for concrete access requests, and *PEP* links directly to the protected resources and is responsible for querying the *PDP* and the enforcement of its returned access decisions. The user sees either a PEP executed action result or a PDP generated permission denied message. However, due to the number of users in SNS the centralized or semi-decentralized certificates, using centralized management [14], are bottle neck. This along with the social requirement of local ownership by content contributors motivates an alternative strategy for distributed certificates using distributed *PEP* and *PDP*. Now a user states desired access control policies using local PDP and a local *PEP* enforce those policies. This ensures complete user control over local resources. If distributed certificates are stored in the stakeholder's namespace, only he or she can access and modify them.

V. Theoretical Assessment

In social networks, access control models face a serious scalability problem, as potentially many more subjects must be mapped to many more resources, regardless of whether a subject has access rights over a resource or not. We can use *u* × *o* × *r* matrix *MAT* to estimate the relations between users, objects and permissions, where *u* is the number of users, *o* is the number of objects and *r* is the number of access permissions. The authorization matrix

$$|MAT| = subject \times object \times access \ldots (iii)$$

is therefore huge and diverse [15]. One often proposed answer is role-based access control (RBAC) [5], which succeeds in reducing complexity [16, 17] by dividing the authorization matrix using a level of indirection via the role concept:

$$subject \times role; \quad |MAT| = role \times object \times access \ldots (iv)$$

However for an SNS with millions of users, the number of access control entries still remains a bottle neck, even with RBAC, e.g. currently Facebook reports over 800 million active users with 90 resources added by each every month [18]. In a traditional DAC access control model, this is over 172 trillion access control entries per month, where every request must traverse the whole list. And the condition with RBAC is even worse as implementing ownership with RBAC is even more expensive [19]. Some of the existing models of access control in SNS introduced the local visibility of objects by introducing friend levels. This reduces the authorization matrix, as the visibility of object is not across the whole system but limited to the social circle of each owner. This limits subjects having potential access over resources:

$$Potential\ Users = Subject \cap Social\ Circle \ldots (v)$$

Thus, the authorization matrix for the whole system is reduced to

$$|MAT| = \sum_{i=1}^{N} \left[\sum_{j=1}^{n} SocialCircle_j \times \sum_{j=1}^{n} Obj_j \times \sum_{j=1}^{n} Rights_j \right] \ldots (vii)$$

This visibility reduces the access control entries from 172 trillion to 28 trillion, which is illustrated in Fig. 3.

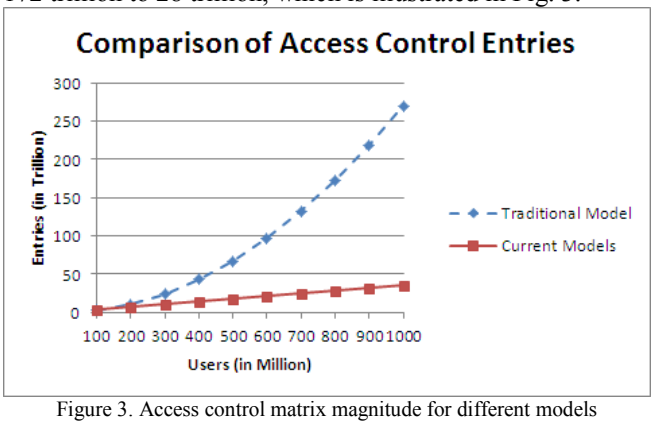

Figure 3. Access control matrix magnitude for different models

The proposed model refined the visibility from social circles to domain based local roles, introduced the object classes based on their privacy clearance and encapsulated various rights over object classes into one attestation certificate. The authorization matrix is distributed among various stakeholders and for one namespace it is reduces to

$$VU \times LR;\ Object \times OC; \quad LR \times OC \times rights \ldots (vi)$$

And the sum of the entire authorization matrices for the whole system under the proposed model is

$$|MAT| = \sum_{i=1}^{N} \left[\sum_{j=1}^{n} LR_j \times \sum_{j=1}^{n} OC_j \times \sum_{j=1}^{n} AC_j \right] \ldots (viii)$$

where *N* is number of users present in the whole system, *n* is the number of *VU* present in *NS j*, LR_j is the number of local roles in NS_j, OC_j is the number of object classes present in NS_j, and AC_j is the number of attestation certificates used for mapping of *LR* and *OC* in NS_j.

These settings under the proposed model give fewer access control entries. For the above case, the number of entries was reduced from 28 trillion to 3 trillion. Fig. 4

shows the number of access control entries generated by user number for a fixed object contribution in the last two cases - current access control models for SNS and the presented model.

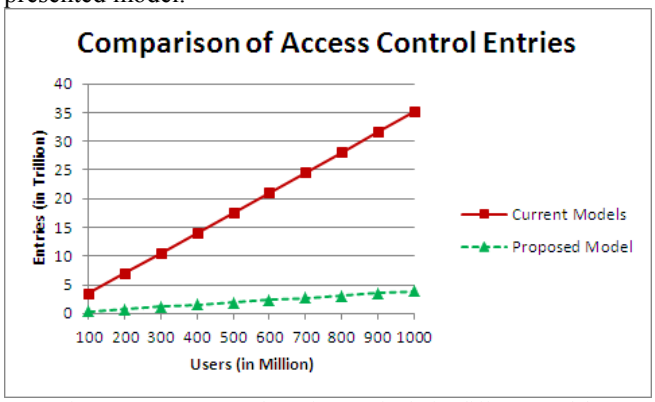

Figure 3. Access control matrix magnitude for different models

VI. CONCLUSION AND FUTURE WORK

This paper proposed a distributed model of access control to manage resources in social networking sites with millions of users. It aims to satisfy both the technical requirements of efficiency and scalability, and the social requirements of ownership and privacy. The latter arise because social network content is contributed by a community of users, who by social logic expect to own what they post. The aim of this design, as in all socio-technical design, is to satisfy both technical and social needs to avoid technical and social errors, with community outrage an example of the latter. The mathematical model was defined, and an architecture for a rights module to work with existing security modules was given, based on distributed certificate issuing and management. Finally, theoretical estimates of access control entries by user numbers were calculated to support the need for this approach for social networking sites.

In the near future, the system feasibility will be tested via a distributed access control plug-in for a NSF granted open knowledge exchange system project, as well as efficiency in terms of retrieval time and storage space. This "rights module" will also be responsible to provide readable error messages for guidance to what users can do, in terms of allowed namespace rights. In social terms, *transparency* of access control rules both lets users anticipate and avoid social errors and reduces community governance corruption as people see the permissions of others [20]. If a social requirement of access control systems is transparency, this sets access control apart from the security aim of system defense, which by definition requires secrecy. The evolution of access control to meet the needs of social networks opens it up to new research dimensions beyond its security origins.

Acknowledgement

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".

REFERENCES

[1] B. Carminati, E. Ferrari, R. Heatherly, M. Kantarcioglu, B. M. Thuraisingham, "A semantic web based framework for social network access control," pp. 177-186 SACMAT 2009.

[2] ABC News, http://www.abc.net.au/news/2010-05-17/teens-murder-sparks-facebook-privacy-plea/829850, 17th May, 2010.

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

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

[5] R. Sandhu, E. Coyne, H. Feinstein, and C. Youman, "Role-Based Access Control Models," IEEE Computer 29(2), 1996.

[6] B. Carminati, E. Ferrari, and A. Perego, "Security and privacy in social networks," In Encyclopedia of Information Science and Technology, 2nd Edition, volume VII, pages 3369–3376. IGI Publishing, Sept. 2008.

[7] S. R. Kruk, S. Grzonkowski, H. C. Choi, T. Woroniecki, and A. Gzella, "D-FOAF: Distributed identity management with access rights delegation," In proc. of the 1st Asian Semantic Web Conference (ASWC 2006), pages 140–154. Springer Verlag, 2006.

[8] B. Ali, W. Villegas, and M. Maheswaran, "A trust based approach for protecting user data in social networks," In proc. of conference of the center for advanced Studies on collaborative research (CASCON'07), pages 288–293, 2007.

[9] B. Carminati, E. Ferrari, and A. Perego, "Enforcing access control in web-based social networks" ACM Transactions on Information & System Security, 2008.

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

[11] B. Whitworth, "The social requirements of technical systems," In: Handbook of Research on Socio-Technical Design and Social Networking Systems, PA: IGI Global, 2009.

[12] E. Hippel, "Democratizing Innovation," MIT Press, Cambridge 2005.

[13] J. Locke, "An essay concerning human understanding", ed. P.H. Nidditch, 2. 25 and 2. 27. Oxford: Oxford University Press, 1975.

[14] M. Thompson, W. Johnston, S. Mudumbai, G. Hoo, K. Jackson, and A. Essiari, "Certificate-based access control for widely distributed resources," In proceedings of the 8th Usenix Security Symposium, pages 215–228, Aug 1999.

[15] F. Kerschbaum, "An access control model for mobile physical objects," Proceeding of the 15th ACM symposium on Access control models and technologies SACMAT 2010.

[16] H. Lee, K. Lee, and M. Chung, "Enterprise application framework for constructing secure RFID application," Proceedings of the 1st International Conference on Hybrid Information Technology, 2006.

[17] M. Wu, C. Ke, and W. Tzeng, "Applying context-aware RBAC to RFID security management for application in retail business," Proceedings of the IEEE Asia-Pacific Services Computing Conference, 2008.

[18] Facebook statistics page, http://www.facebook.com/press/info.php?statistics , accessed on 30th July 2011.

[19] R. S. Sandhu and Q. Munawer, "How to do discretionary access control using roles," In Proceedings of the Third ACM Workshop on Role-Based Access Control (RBAC 1998), pages 47–54, Oct. 1998.

[20] J. Kooiman, M. Bavinck, R. Chuenpagdee, R. Mahon, R. Pullin, "Interactive governance and governability: an introduction," The Journal of Transdisciplinary Environmental Studies, 7(1), 2008.