2011 Sixth International Conference on Availability, Reliability and Security

ACON: Activity-Centric Access Control for Social Computing

Jaehong Park, Ravi Sandhu, and Yuan Cheng
Institute for Cyber Security
University of Texas at San Antonio
jae.park@utsa.edu, ravi.sandhu@utsa.edu, ycheng@cs.utsa.edu

***Abstract*—With increasing amount of sensitive user data stored in social computing systems (SCSs) and lack of consensus on how it should be protected under meaningful control by the average user, security and privacy has become a pressing problem that must be addressed. We propose the concept of user and SCS activity as a natural aspect of social computing which influences access control in a manner distinct to SCSs. We propose an activity-centric access control or Activity CONtrol (ACON) framework for social computing to facilitate both privacy setting from user side and administration from SCS side. We further propose an $ACON_{user}$ model for user activity control and session management. We illustrate how the model captures the user activities using several SC examples.**

***Index Terms*—Security; Privacy; Social Computing;**

I. Introduction

In typical access control systems, the main focus is to control a user's access to stored content. In social computing systems (SCSs), there are additionally other kinds of activities that need to be considered for access control. In a SCS, a user performs activities not only on shared content but also against target users (e.g., a user pokes another user or recommends other users to be a friend to each other, a buyer rates sellers, etc.). Moreover, when a SCS makes control decisions, the decisions are likely to be influenced by related users' control activities. For example, consider the sample social graph in Figure 1. Here, Bart may not want his activity to be notified to Homer or may not want to receive any notification of Homer's activity. Likewise, Homer may not want Bart to access any violent content, to share personal information with others or to become a friend of Homer's co-workers. We say a user performs *control activities* to express these preferences.

Furthermore, in SCS, there are activities performed by the system to provide services and resources that can promote user interactions or sustain the SCS provider's business. For example, Amazon collects and aggregates users' ratings on products to generate best rated products or Facebook notifies a user's activity to the user's friends or recommends friendships to users who share a common friend. These SCS's activities also need to be evaluated for control decision since users may not want their shared information to be used for SCS's analysis or may not want to receive some of these SCS services. From access control point of view, both users' control activities and systems' automated activities are rather unique to SCSs and seldom considered in traditional access control models.

In this paper, we propose the notion of *activity* as a key concept for access control and identify various activities found in SCSs. We distinguish *access (or usage) activities*, such as viewing a picture on a friend's page, from control activities, such as limiting access to a picture on a user's page to user's friends only.

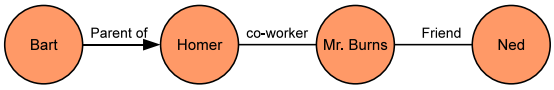

Fig. 1. A User Relationship Example

A distinguishing feature of SCSs is the diffused nature of control activities and their interactions. This requires a more sophisticated approach to control activities than found in traditional access control models. For this purpose we develop an access control framework called ACON (Activity CONtrol) that supports controls on access and control activities in SCS. The ACON framework supports personalized user privacy control by utilizing individualized user policies/attributes and resource policies/attributes. It also supports automated management of SCS. The scope of ACON is beyond the scope of traditional access controls in that ACON allows users' individualized controls on their own or other users' activities, as well as on SCS's automated management activities. As such it provides a unique perspective on access control suitable for SCSs.

II. Motivation and Related Works

In discretionary and lattice-based access controls, the main focus is on controlling users' access activities on objects. In role-based access controls (RBAC), controls on administrators' control activities have also been considered [9]. Usage control [5], [6] adds attribute mutability to cover a system's automated attribute management that is required to reflect the result of users' usage activity such as decreasing a user's credit balance per movie play. However these models do not provide the means for a user to influence control decisions on other users' activities or system's activities as often found in SCSs.

Online social networking is a prominent subset of social computing that utilizes a user relationship based social graph to control user activities. While recent access control literature on online social networking such as [1], [2], [3], [4], [10] deals with various issues that are distinct from traditional access control, they mainly focus on issues related to user relationship based social graph. Our framework is independent of social graphs and deals with fundamental aspects of access control in SCSs, including online social networks.

978-0-7695-4485-4/11 $26.00 © 2011 IEEE
DOI 10.1109/ARES.2011.42

III. Activities in Social Computing Systems

Next, we classify SCS activities into different categories.

A. User's Activities

User's Usage Activities (UA). Just as in traditional access control systems, in SCSs a user can read, write or modify resources. However in SCSs, a user also performs certain activities on target users. For example, in Facebook, a user may poke another user, recommend a friendship, invite another user as a friend or accept a friendship invitation.[1]

User's Control Activities (CA) on Resources. Users in SCS perform control activities on resources by changing attributes and policies of resources. Similar to discretionary access control users can control access to their own resources, although in SCSs this control can be influenced by other users.[2] For example, providing access to friends of friends conveys the ability to a user's friends to further control who are friends of friends, thereby controlling access to the user's resources. As another example it may be possible for a parent to control access to a child's resources. Some form of social control for shared content, such as pictures with multiple people, may also be desirable [10].

User's CA on Users. In SCSs, a user needs to control her own or other users' activities without reference to a concrete resource. Often a user wants to control certain users' activities even though the actual resource may not yet be available. In discretionary access control, typically control policy is defined based on a concrete specific resource. In an SCS, for example, Homer may want to prohibit Bart from uploading any personally identifiable information such as a photo that includes Bart's face; or Homer may want to make sure he does not receive any notification of friends activities. In RBAC, although a user can change other users' privilege by delegating some or all of her role, the user is not allowed to change user- or permission-role assignments or role hierarchy.[3] On the other hand, for example, in eBay and Amazon like e-commerce systems, a user is allowed to rate sellers to change their trustworthiness which in turn may change sellers' privilege. This kind of user's control activities on users is rarely discussed in traditional access control policies and models.

User's CA on Sessions. A session represents an active user who has logged into the SCS.[4] In SCS, the user-session distinction is crucial to allow a user's session to carry attributes and policies that are different from those of the users. For example, Bart may not want to reveal a complete set of his friend information when he chats with Homer; or Bart may not want to receive any activity notification from the SCS on occasion. Typically, a session inherits all attributes and policies of a user who created it. However, a user may want to disable certain user attributes or policies in a particular session or add some temporary attributes or policies for a session. From the privacy protection point of view, user's ability to control session attributes and policies is crucial to realize users' privacy preference.

[1] A friendship invitation acceptance can be viewed as a user's control activity since it modifies the user's attributes. As a rule of thumb, although a user activity results in an attribute or policy change, if the activity is not intended to manage attributes or policies directly, we consider it as a usage activity. Though some ambiguity still remains, the goal of this classification is not to make a tight distinction between activity types but to identify various kinds of activities not typically found in traditional access control literature.

[2] Some theoretical models for discretionary access control allow ownership to be delegated from one user to another, but these features are not common in most prevalent commercial implementations.

[3] Technically, a user can be assigned an administrative role which enables changes to these assignments. This would not be practical in SCS since such an administrative role would need to be very specific to each individual user.

[4] The term session is borrowed from role-based access control models [8].

B. SCS's (Automated) Activities

SCS performs automated activities that are triggered either by user activities or other system attributes such as time, location/platform of accessing device, and other system status. Traditional access controls rarely recognize this. Although usage control includes mutable attributes, there is no facility for other kinds of system activities such as providing services (e.g., location-based coupons, friends recommendation) or value-added resources (e.g., most viewed video-clips, product ratings) that are generated by utilizing user resources. The SCS activities comprise service, control and decision activities.

SCS's Service Activities. SCSs perform various service activities to provide functionality and information to promote users' social interactions and information sharing. To do so, a SCS typically utilizes users' shared resources and generates value-added resources and services. For example, each time a user logs in, a SCS collects information about nearby friends and shows who is available to the user. If a user opts out, she will not receive this kind of notifications or will not be included in notification information sent to other users. A SCS may collect user information to provide a recommendation for potential friends. Also a SCS may notify one user's activity to all of her friends. Because these SCS's activities utilize user-provided resources, controls on these activities are likely to be influenced by control activities of providing users or users who are related to the provider or the provided resource.

SCS's Control Activities. In addition to service activities, SCS also performs automated control activities to control users, resources and sessions. This is done through managing attributes and policies of users, resources and sessions. For example, a recommendation system like eBay regularly collects user activity (such as sales) information to generate trustworthiness of each user. In online sales or review system like Amazon.com and Edmunds.com, the system regularly collects users reviews on products to update popularity or reliability level of the products. Such SCS control activities are likely to be influenced by users' usage and control activities. Also, when a user creates a session, often the SCS requires parts or all of user attributes and policies to be inherited to the session or the SCS adds additional attributes and policies to the session as necessary. For example, a SCS may add a user's access device information so some user activities (e.g., account information updates) can be restricted for security purposes.

SCS's Decision Activities. One of the main functions of activity control systems in SCSs is decision making activities. In most traditional access control systems, only users' usage activities are considered and evaluated for permission. In SCS environment, users' requests for usage activities and control activities as well as SCS's requests for service activities and

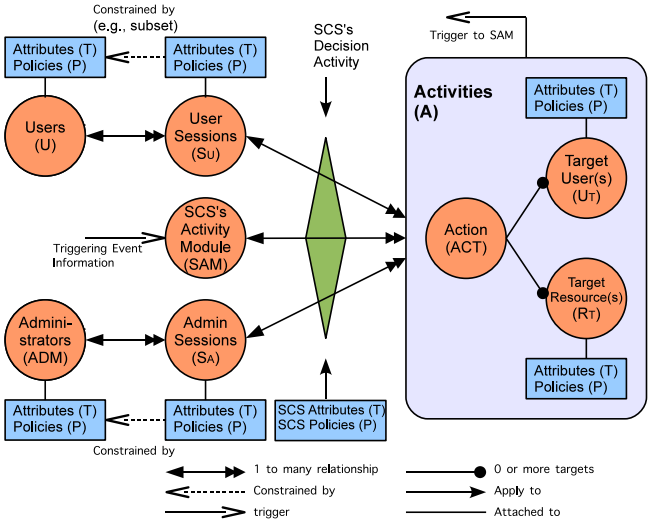

Fig. 2. ACON Framework

control activities are evaluated by SCS for control decisions. This is discussed further in the next section.

C. SCS Administrator's Activities

SCS administrators' need to perform manual activities to manage overall system, users and their activities. Controls on administrators' activities is important especially for highly distributed computing systems. In RBAC, such controls have been discussed extensively to manage administrator's roles and permissions. In SCS, most research has focussed on controlling user activities, but how administrators' activities should be managed is not typically considered. Applying user activity control approaches to SCS administrators' activity may lead to interesting research, but is beyond the scope of this paper.

IV. ACON Framework

We believe access control system for SCS needs to control various types of activities identified earlier not only for better security and privacy protection but also for better functionality and management. To support controls on these activities, we propose a conceptual framework of activity control or ACON for SCSs. In this section we identify components of ACON framework and discuss characteristics of the framework.

A. ACON Components

Figure 2 shows a conceptual diagram for ACON framework.[5]

Users. A user represents a human being who performs activities in an SCS. Users carry attributes and policies. User attributes are information about the user, e.g., address and friends list. User policies are user preferences and limits set by either herself, other users or SCS. User attributes/policies can be managed by the user, other related users (e.g., parents), or the SCS. These policies/attributes may be changed indirectly as a result of user activities (e.g., consumable attributes like a credit balance and a reputation based on other users' ratings).

Sessions. A session represents an active user who has logged into the SCS. In SCS, the user-session distinction is crucial to facilitate a session with different policies and attributes from those of users. Thereby a user can create a session that carries different degree of privacy preference. A user can have multiple sessions concurrently if permitted by the SCS, while a session belongs to exactly one user. In Figure 2 this is shown with the single and double arrowheads.

In the simplest case, all user attributes/policies are inherited by the user's session(s). More generally, the session may inherit only some of these. In Figure 2, "constrained by" relation depicts this. A session may have additional attributes (e.g., IP address, device's platform, its location, or its trustworthiness) and policies (e.g., limited privileges for a mobile device, or a session invisible to other users). It may have a sanitized version of user attributes (e.g., "over 18", not an actual age).

In current SCSs attributes and policies of a session are likely to be fully controlled by the SCS. For example, some SCSs facilitate a session to have different privilege if the user is logged in from a mobile device or public machine. This may be to constrain functionality from a limited device or may involve discrimination amongst security-sensitive operations. We believe it is highly desirable to support sessions with user controlled attributes and policies especially for user privacy purpose. For instance, Ned may not want to reveal his friends information to SCS users occasionally. This can be achieved by creating a session that does not convey his friends information. However, if Ned's session that carries no friends list attempts to be a friend of Bart, this might create a conflict with Homer's policy that says Bart cannot be a friend of anyone who is Homer's coworker or a friend of Homer's coworkers. This means some user policies and attributes must be carried by a session for certain activities. We believe research on the relationship between session and user attributes and policies will provide better foundations for more nuanced access control and privacy in future SCSs.

Activities. As discussed, the activity notion encompasses user's activities, SCS's activities and SCS administrator's activities. Each activity is initiated via a session on behalf of that session's user or SCS administrator, or by SCS's Activity Module (SAM) which is a conceptual entity that performs automated SCS activities. The SCS decides whether or not an activity is permitted. Some activities send a trigger to SAM if additional SCS activities are required. A session and SAM can have multiple activities whereas each activity is initiated by a single session or SAM. Each activity consists of an action, zero or more target resources and zero or more target users.

Action. Action is an abstract function available in SCS. Examples are a user *reads* and *writes* a comment; a user *rates* a product or a seller; a user *invites* other users as a friend; or a user's activity triggers an *activity notification* action to be delivered to her friends. Actions can be against target resource(s), target user(s) or both target resource(s) and target user(s). For example, a user rates target resource(s) or target

[5] In [7], we introduced a framework that only covers user activity controls in its scope. In this paper we significantly extend this prior work to include SCS activities and SCS administrator activities in the picture, as well as developing a formal model for user activities in section V.

user(s); a friendship recommendation require two or more target users; and typical notification actions require both target user(s) (e.g, friends) and resource(s) (e.g., activity log info).

Target Users. Target users are recipients of an action. To be accurate, although an action is made to a target user, it is the target user's session(s) that actually receives the action. For example, if Ned invites Homer as a friend or for a chat, Homer is the target user while Ned is called the acting user. Here it is Homer's session that actually receives Ned's invitation. If Homer's session has a policy that says it does not want to have any chat, Ned's attempt to chat will not succeed.

Target Resources. Target resources are the resources involved in the action. These include users' shared content, SCS's shared content, user policies and attributes, resource policies and attributes, session policies and attributes and any information that users, SAM or SCS administrators can access or manage in SCS. Considering policies and attributes as part of the resource abstraction allows the framework to support the ability for users to partially control their own attributes and policies as well as attributes and policies of other related users. This also allows SCS (i.e., SAM) and SCS administrators to control user policies and attributes in a single framework. Moreover, this allows additional policies and attributes to apply to these policy and attribute resources. For example, if Bart has a policy that says no access to violent content, there could be an attribute on this policy (such as the policy creator) or a policy on the policy (stipulating that only Homer can change that policy). Also if a photo has a provider information as an attribute, this attribute can have a policy that says only friends of the provider can read the provider information. While theoretically, this chaining can be continued indefinitely, we believe practical SCSs are not likely to provide policies and attributes on policies and attributes beyond a couple of levels.

SCS's Decision Activity. When an activity is attempted, SCS consolidates all the necessary individual policies and attributes together with the SCS's own policies and attributes, and then utilizes them to make a decision on whether or not to permit the attempted activity. For instance, let's say Homer sets a policy that says Homer's co-workers and their direct friends cannot be a friend of Homer's children. When this policy is specified by Homer, the SCS makes sure activity decisions on all the friendship requests from and to Bart reflect Homer's policy (either by updating Bart's policy at the time of the Homer's policy creation or evaluating policies of Bart's parents at each time Bart attempts to be a friend with someone). If Bart (in a session) tries to send a friendship invitation (an action) to Ned (a target user), the SCS evaluates Bart's policy and possibly policies of Homer, then verifies whether any of Ned's friend (the target user's attribute) is one of Homer's co-worker (see Example 2 for a formal specification).

SCS's Activity Module (SAM). SAM is a conceptual abstraction of functions that performs SCS's automated service and control activities.[6] Specifically, user activities or system status (e.g., time and CPU load) trigger SAM. SAM then initiates necessary service and control activities to support SCS services and management. SCS administrator's activities or SAM's own activities can also trigger SAM.[7] By introducing SAM, we can conveniently capture the fact that even SCS's activities should be controlled in SCS. SCS typically facilitates various automated activities to provide services or to manage attributes and policies of users and resources. However, decisions on these activities are likely to be influenced by users' control activities (e.g., configuring user or resource policies). Thus, if a user doesn't want her activity information to be used for SCS's statistical analysis and SCS honors it, SAM's automated access on the information will be denied while other users' activity information is granted for SAM's access.

One may argue SAM can initiate only allowed activities at first place hence doesn't require decision process. However, this approach still needs a selection/decision process to reflect target users' or resources' policies regardless of the actual decision point. ACON is a conceptual framework and includes SAM to capture the necessary decision process on the system's service and control activities. It does not suggest any implementation level design specifications.

SCS Administrators. An SCS administrator represents a human being with a management role in SCS. Just like users, an SCS administrator may carry attributes and policies that apply to the SCS administrator. If SCS is highly distributed and managed by various SCS administrators who have various levels or kinds of administrative privileges, it could be beneficial to have personalized attributes and policies for each SCS administrator. On the other hand, if SCS relies on a small number of administrators, having personalized attributes and policies may not be meaningful or desirable. Controls on SCS administrator's activities need further research and are therefore not discussed extensively in this paper. Also, one can consider a target SCS administrator for certain activities but this is not covered here, since our current focus is mainly on user and SCS activities.

B. ACON Framework Characteristics

One of the design characteristics of our framework is **policy individualization**. Unlike lattice- and role-based access control where there is one system-wide access control policy, ACON allows users to carry their own policies that includes privacy preferences and activity limits. These policies are collectively used by SCS for control decision on activities. Unlike typical discretionary access control, individualized policies in ACON allow users to configure related users' policies. We believe this characteristic is crucial for access control in SCS.

Another characteristic in ACON is a **separation of user and resource policies**. By utilizing user policies, a system can control activities more effectively if the activity controls are specific to a user without knowing any particular resource. For example, if Homer does not want to receive any violent content, it should be more effective to add the policy rule in Homer's policies rather than adding Homer (and all other users' names who do not want violent content) into policies of each item of violent content.

[6]Note that SAM does not include SCS's decision activities as shown in Figure 2 since SAM's activities are evaluated for a decision by SCS.

[7]Our framework allows multiple levels of SAM's activities to be triggered by an activity of SAM. Although we think this is not likely to occur frequently, the model allows it.

TABLE I
$ACON_{user}$ MODEL DEFINITIONS

$ACON_{user}$	Definitions
User Activity Control	U, S, ACT, R, T, P, SCS and D (users, sessions, actions, resources, attributes, policies, social computing system and decision predicate, respectively); $U_T \subseteq U$ and $R_T \subseteq R$ (target users and target resources, respectively); dot notation: we understand $e.T$ and $e.P$ to respectively denote the set of attributes and set of policies associated with entity e; A, the set of activities is defined as $A \subseteq ACT \times (2^{R_T} \times 2^{U_T} - \emptyset)$; Let $A = \{a_1, a_2, ..., a_n\}$, we denote the components of each individual element as $a_i = (a_i.ACT, a_i.R_T, a_i.U_T)$; $AP\_R_T : A \rightarrow 2^{R_T \times P}$, $AP\_U_T : A \rightarrow 2^{U_T \times P}$, $AT\_R_T : A \rightarrow 2^{R_T \times T}$, $AT\_U_T : A \rightarrow 2^{U_T \times T}$, mappings of activity to a set of target resources and policies, a set of target users and policies, a set of target resources and attributes, and a set of target users and attributes respectively defined as: $AP\_R_T(\{a_1, .., a_n\}) = AP\_R_T(\{a_1\}) \cup ... \cup AP\_R_T(\{a_n\})$, $AP\_R_T(\{a_i\}) = \{(r_t, p) \mid r_t \in a_i.R_T, p \in r_t.P\}$ $AP\_U_T(\{a_1, .., a_n\}) = AP\_U_T(\{a_1\}) \cup ... \cup AP\_U_T(\{a_n\})$, $AP\_U_T(\{a_i\}) = \{(u_t, p) \mid u_t \in a_i.U_T, p \in u_t.P\}$ $AT\_R_T(\{a_1, .., a_n\}) = AT\_R_T(\{a_1\}) \cup ... \cup AT\_R_T(\{a_n\})$, $AT\_R_T(\{a_i\}) = \{(r_t, t) \mid r_t \in a_i.R_T, t \in r_t.T\}$ $AT\_U_T(\{a_1, .., a_n\}) = AT\_U_T(\{a_1\}) \cup ... \cup AT\_U_T(\{a_n\})$, $AT\_U_T(\{a_i\}) = \{(u_t, t) \mid u_t \in a_i.U_T, t \in u_t.T\}$; $AP(a) = AP\_R_T(a) \cup AP\_U_T(a)$, $AT(a) = AT\_R_T(a) \cup AT\_U_T(a)$; $allowed(s, a) \Rightarrow D(s.P, s.T, a, AP(a), AT(a), scs.P, scs.T)$, where $s \in S$ and $a \in A$.
Session Management	$user\_sessions : U \rightarrow 2^S$, $session\_users : S \rightarrow U$; $user\_added\_sessionT : S \rightarrow 2^T$, $user\_removed\_sessionT : S \rightarrow 2^T$; $scs\_added\_sessionT : S \rightarrow 2^T$, $scs\_removed\_sessionT : S \rightarrow 2^T$, $scs\_required\_sessionT : S \rightarrow 2^T$; $user\_added\_sessionP : S \rightarrow 2^P$, $user\_removed\_sessionP : S \rightarrow 2^P$; $scs\_added\_sessionP : S \rightarrow 2^P$, $scs\_removed\_sessionP : S \rightarrow 2^P$, $scs\_required\_sessionP : S \rightarrow 2^P$; $user\_removed\_sessionT(s) \subseteq \{t \in T \mid t \in session\_users(s).T \wedge t \notin scs\_required\_sessionT(s)\}$; $user\_removed\_sessionP(s) \subseteq \{p \in P \mid p \in session\_users(s).P \wedge p \notin scs\_required\_sessionP(s)\}$; $assignS\_T : S \rightarrow 2^T$, $assignS\_P : S \rightarrow 2^P$, assignment of attributes and policies to sessions respectively; $assignS\_T(s) \subseteq \{t \in T \mid (t \in session\_users(s).T) \vee (t \in user\_added\_sessionT(s)) \vee (t \in scs\_added\_sessionT(s)) \wedge \neg((t \in user\_removed\_sessionT(s)) \vee (t \in scs\_removed\_sessionT(s)))\}$; $assignS\_P(s) \subseteq \{p \in P \mid (p \in session\_users(s).P) \vee (p \in user\_added\_sessionP(s)) \vee (p \in scs\_added\_sessionP(s)) \wedge \neg((p \in user\_removed\_sessionP(s)) \vee (p \in scs\_removed\_sessionP(s)))\}$.

As discussed, characteristics such as **user-session distinction**, **user relationship independent access control**, and **SCS's automated service and control activity** are also novel and make ACON applicable to today's and future SCSs.

V. $ACON_{user}$, A USER ACTIVITY CONTROL MODEL

In this section we formally define a user activity control model called $ACON_{user}$. This model does not include SAM's and SCS administrators' activity controls. Although these are important, we believe a user activity control model should be developed first and will provide a basis for models of other activities in SCSs. Table I gives the definition of the $ACON_{user}$ model for user activity and session management.

A. $ACON_{user}$ Model Definition

$ACON_{user}$ consists of users (U), sessions (S), actions (ACT), resources (R), attributes (T), policies (P), social computing system (SCS) and decision predicate (D). Target users (U_T) is a subset of U and target resources (R_T) is a subset of R. Activities (A) comprise an action that is performed against either U_T, R_T, or both U_T and R_T. Decision is a functional predicate that examines activity requests for control decision by utilizing session policies ($S.P$) and attributes ($S.T$), the requested activity (A), activity policies ($AP(a)$) and attributes ($AT(a)$), and system policies ($SCS.P$) and attributes ($SCS.T$). Here, $AP(a)$ and $AT(a)$ are subsets of policies and attributes of target users and target resources of the action. We write $allowed(s, a)$ to express that a session s is allowed to perform activity a. $allowed(s, a)$ is formulated as a necessary condition to allow for inclusion of other rules that might be necessary for finer controls.[8]

Note that a user initiates an activity through a session, hence the model utilizes session policies and attributes rather than the user's for an activity decision.[9] As discussed, session policies and attributes are constrained by user policies and attributes. When a user creates a session, a user may indicate her privacy preference for that session by configuring session policies and attributes as within the boundary allowed by a SCS. Also, the SCS may add, remove, or change some policies and attributes. This session management is defined in Table I.

Functions $assignS.T$ and $assignS.P$ map a session to a set of attributes and policies respectively, where $assignS.T$ and $assignS.P$ include all the policies and attributes of the initiated user and policies and attributes added or removed as requested by the user and the system. We assume a user can only do so as allowed by the SCS. In absence of any changes by the user of the session and by the SCS, the session carries an identical copy of user attributes and policies.

[8] A similar approach can be found in the classic Bell-LaPadula (BLP) model and $UCON_{ABC}$ usage control model [6].

[9] Activity policies and attributes ($AP(a)$, $AT(a)$) may include policies and attributes of target users (not a target session's). Although some activities like a chatting request require policies and attributes of target sessions, our current model utilizes target users' policies and attributes for the sake of simplicity.

The examples below show how typical user activities can be realized within $ACON_{user}$. Here, we assume a session carries all policies and attributes of the related user.

Example 1 A buyer can rate a seller only if the buyer bought a product from the seller ($SCS.P$).
N: a list of users, $sellerList : S \rightarrow 2^N$
$allowed(s, rate, u_t) \Rightarrow u_t \in sellerList(s)$

Example 2 A user can invite another user as a friend if not already a friend. If the user < 18, the system must check parents' policy to ensure that one parents' policy says that children cannot be a friend of the co-workers of the parents.
L: a list of users, N: natural numbers, $age : S \rightarrow N$,
$f : S \rightarrow 2^L$, $pt : S \rightarrow 2^L$, $cw : U \rightarrow 2^L$ (a mapping of a session/user to friends, parents and coworkers, respectively)
$SCS.P$ = {A session must have a "pt" and an "age" attributes; If a user under 18, check the parents' policies}[10]
$pt(s).P$ = {My child cannot be a friend of my coworker}[11]
$allowed(s, invite, u_t) \Rightarrow$

$$\begin{cases} u_t \notin f(s), & \text{if } age(s) \geq 18; \\ u_t \notin f(s) \wedge cw(u_t) \cap pt(s) = \emptyset, & \text{if } age(s) < 18. \end{cases}$$

Example 3 A user can change own policy. If a user is under 18, only parents can change the user's policy ($SCS.P$).
N: a list of users, $age : S \rightarrow N$, $children : S \rightarrow 2^N$
$allowed(s, change, u_t.P) \Rightarrow$

$$\begin{cases} u_t \in session\_user(s), & \text{if } age(s) \geq 18; \\ u_t \in children(s), & \text{if } age(s) < 18. \end{cases}$$

Example 4 A user can recommend a friendship between two friends if they are not a friend to each other($SCS.P$).
N: a list of users, $friends : S \rightarrow 2^N$
$allowed(s, f.recommend, u_t1, u_t2) \Rightarrow (\{u_t1, u_t2\} \in friends(s)) \wedge (u_t2 \notin friends(u_t1)) \wedge (u_t1 \notin friends(u_t2))$

In Example 1, a session carries a list of sellers as an attribute ($S.T$). The system policies include a rule that says a session can rate a target user if the target user u_t is found in the seller list of session s. In Example 2, since a session must have a "parents" attribute, $pt(s)$ is equal to $pt(u)$. The $cw(u_t)$ is an attribute of a target user. The system includes a policy that check parents' policies of the session if the session's age is under 18. Here, parents of a session should not be found in coworker list of the target user. In Example 3, $u_t.P$ (a target user's policies) is the target resource since policies and attributes are also resources in ACON. Example 4 shows an activity that need two target users for an action.

B. $ACON_{user}$ Model Discussion

Perhaps one of the most comparable access control model to $ACON_{user}$ is the $UCON_{ABC}$ usage control model [6]. Both models utilize user and resource attributes to make a control decision and both recognize the necessity of automated system activities. However, there are also some significant differences. Currently $ACON_{user}$ model deals with only pre-authorization of UCON while ignoring any continuous controls during the life time of activities. While UCON recognizes automated system activities, it mainly updates a user's attributes to reflect the result of that user's activities. In UCON, an update is a side effect of user activities. In ACON, we expand this to allow a user's both unintentional and intentional updates on her own or other users' attributes and policies.

While UCON controls a subject's access, ACON makes user-session distinction to enhance users' control capability and SCS's management. However, our current model considers only a static session where no policies and attributes of a session are changed during its life time. Further studies on dynamics of session will enhance our framework and models.

In this section we defined a user activity control models as well as a model for session management and further discussed these models using several examples. As mentioned earlier, the model covers only a part of the ACON framework. However we believe the proposed framework and models are essential for access control in SCSs and will provide a foundation for advanced security and privacy protection.

VI. Conclusion

While there are plethora of social computing services (including online social networking services), privacy protection and users' control capabilities in these services are still rudimentary. Although many studies on access controls for social computing have been done, most of these focus on user relationship based controls. In this paper, we propose a novel access control framework for SCSs and our initial models for user activity control and session management. We believe our activity centric access control framework provides a solid foundation for better security and privacy protection and SCS management. We also believe this line of work presents promising future research directions.

Acknowledgment

This work is supported by NSF and the state of Texas.

References

[1] B. Carminati, E. Ferrari, R. Heatherly, M. Kantarcioglu, and B. Thuraisingham. A semantic web based framework for social network access control. *Proceedings of the 14th ACM SACMAT*, 2009, pp. 177-186.

[2] B. Carminati, E. Ferrari, and A. Perego. Enforcing access control in Web-based social networks. *ACM TISSEC*, vol. 13, no. 1, 2009, pp. 1-38.

[3] P.W.L. Fong, M. Anwar, and Z. Zhao. A privacy preservation model for Facebook-style social network systems. *ESORICS*, 2009, Springer.

[4] P.W.L. Fong. Relationship-based access control: protection model and policy language. *ACM CODASPY* 2011.

[5] J. Park and R. Sandhu. Towards Usage Control Models: Beyond Traditional Access Control. *ACM SACMAT* 2002.

[6] J. Park and R. Sandhu. The $UCON_{ABC}$ Usage Control Model. *ACM TISSEC*, Volume 7, Number 1, February 2004, pages 128-174.

[7] J. Park, R. Sandhu and Y. Cheng. User-Activity-Centric Framework for Access Control in Online Social Networks. *IEEE Internet Computing*, online preprint 2011.

[8] R. Sandhu, and E. Coyne, and H. Feinstein, and C. Youman. Role-based access control models. *IEEE Computer*, V. 29, N. 2, Feb. 1996, pp. 38-27.

[9] R. Sandhu, V. Bhamidipati and Q. Munawer. The ARBAC97 Model for Role-Based Administration of Roles. *ACM TISSEC*, V. 2, Feb. 1999.

[10] A. Cinzia Squicciarini, M. Shehab, and F. Paci. Collective privacy management in social networks. *Proceedings of the 18th international conference on World wide web*, 2009, pp. 521-530.

[10] More formally, these will be representations of the stated policies.

[11] As above.