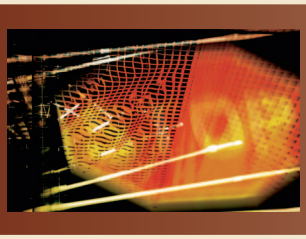
# Security and Privacy Challenges in Open and Dynamic Environments

**Lalana Kagal,** Massachusetts Institute of Technology
**Tim Finin and Anupam Joshi,** University of Maryland, Baltimore County
**Sol Greenspan,** National Science Foundation



**Achieving secure open and dynamic environments requires shared vocabularies, behavioral norms, and trust models.**

Information system security and privacy, once narrow topics primarily of interest to IS designers, have become critically important to society at large. The scope of associated challenges and applications is broadening accordingly, leading to new requirements and approaches.

Challenges arise as information systems evolve into distributed systems that are *open* in that they don't preidentify a set of known participants, and *dynamic* in that the participants change regularly, not just due to occasional failures. Such systems include peer-to-peer networks, grid computing environments, ad hoc networks, Web services, pervasive computing spaces, and multiagent systems.

In addition, as applications become more sophisticated and intelligent, they require greater degrees of decision making and independence. The long-range vision is of systems that let people, agents, services, and devices seamlessly interact as autonomously as possible while preserving appropriate security and privacy policies.

## SECURITY AND PRIVACY CHALLENGES

Consider a hospital emergency facility, which contains a wide range of devices—such as defibrillators, x-ray machines, a computed tomography scanner, screens, and dialysis machines—and numerous users including doctors, nurses, specialists, and paramedics. As these people move about, agents on their personal devices detect, and are detected by, the pervasive infrastructure.

The devices must discover the services and information of interest from the infrastructure and other devices in the vicinity, negotiate for access, control information exchange, and monitor for suspicious events to be reported to the community. For example, a doctor's agent might retrieve a patient's first aid information from a paramedic's PDA.

However, not everyone should have access to all devices, services, and information available in the space. Appropriate security policies must be enforced, such as

- specialists can only access information on a patient they're treating,
- defibrillators can only be used on patients without a do-not-resuscitate (DNR) designation, and
- paramedics can't access patient insurance data.

Privacy policies must also be considered. For example, a doctor who discovers that a patient has a drug dependency could be prohibited from disclosing this information to anyone including the nurses attending the patient.

An environment of this kind presents several security and privacy challenges. Agents belonging to different people and organizations have various identities as well as distinct enforcement mechanisms. This implies that agents might not be able to understand each other's security and privacy requirements or determine how to fulfill them.

Another problem is that people's identities might not be predetermined, making authentication difficult. Commonly used mechanisms such as role-based access control, access control lists, and public-key infrastructure require participants to be predetermined and generally can't adapt to evolving requirements.

Achieving secure open and dynamic environments requires shared vocabularies, behavioral norms, and trust models for communicating and cooperating applications, agents, and devices. Drawing on diverse areas within computer science as well as various social sciences, researchers must explore new languages for sharing knowledge models and data, declarative policies for information assurance and control, and trust-based approaches to security and privacy.

## KNOWLEDGE SHARING

As distributed information systems become more ubiquitous, autonomous, and complex, the need to ground them on common data models grows stronger. Agents in such systems must be able to exchange information, queries, and requests with some assurance that they have a common meaning. To facilitate cooperation and prevent misunderstandings, better languages are needed for sharing knowledge about individuals, events, and situations.

One possible approach is to employ Semantic Web technologies for modeling and reasoning about information. The Semantic Web (www.w3.org/2001/sw) is an enhancement of the World Wide Web that deals primarily with data instead of documents.

Semantic Web technologies include languages such as the Resource Description Framework and the Web Ontology Language (OWL) for defining ontologies—vocabularies that represent knowledge domains—and describing metadata, as well as tools for reasoning about these descriptions. Some commonly used ontologies include OWL-S for describing Web services and Friend of a Friend for describing people, their relationships and affiliations, and other information typically found on human-readable homepages.

Developers could use these same languages to define ontologies for describing system information such as participants' characteristics and the context as well as security, privacy, and trust requirements and values. Using any one of several reasoning engines, all agents who understand Semantic Web languages could understand one another's ontologies and system data and thereby exchange information and services while preserving the security and privacy requirements.

## COMPUTATIONAL POLICIES

While clients, services, and mediators in distributed systems are physically separate and subject to disconnection, most security and privacy models assume that such entities are predetermined and relatively static, sharing the same domain knowledge and infrastructure.

Developers could use a declarative-policy-based approach to describe the ideal behavioral norms of entities in such environments in a machine-understandable specification language. These policies would describe what an entity can or must do in a certain context and allow developers to modify an entity's behavior without affecting the underlying mechanisms and architecture.

**Explicit policies can define permissions, obligations, norms, and preferences for an agent's actions and interactions with other agents and programs.**

Along with providing the required openness, this approach also provides greater autonomy as entities can choose whether to accept a particular norm.

A *policy* is an explicit representation of constraints and rules that govern an agent or system's behavior. For example, the policy of not using defibrillators on patients who've signed a DNR causes doctors to treat patients differently.

Explicit policies can define permissions, obligations, norms, and preferences for an agent's actions and interactions with other agents and programs. Such policies, especially those expressed in high-level declarative languages, can form the basis for electronic contracts and provide a sublanguage useful for negotiating agreements and commitments.

Rei (http://rei.umbc.edu) is an example of a declarative policy language that uses Semantic Web technologies to describe policies as constraints over allowable and obligated actions on resources in a distributed environment. Rei can describe policies over heterogeneous domain information as defined in a Semantic Web language, thereby providing a common understanding among participants who might not use the same information model.

Rei is suitable for open and dynamic environments because it describes policies in terms of user attributes, actions, contextual data, and other domain knowledge instead of identities; it also provides greater extensibility because it can describe policies at different abstraction levels.

Rei also supports sanctions and conditional permissions, which are common in human societies.

Users are often overconstrained and might not completely satisfy all security and privacy policies, resulting in undesirable consequences such as loss of reputation and penalties. Rei allows modeling such consequences as *sanctions* so that autonomous entities or providers can reason about them to decide whether to deviate from a certain policy.

Consider, for example, the policy that doctors mustn't disclose that a patient has a disease such as HIV if being treated for an unrelated ailment. Physicians who do reveal this information can be sanctioned by losing their license or being sued.

*Conditional permissions* let an entity perform a certain action or set of actions under the condition that it will assume certain additional responsibilities. For example, doctors who consult a specialist must delegate to that person the right to access their notes on the patient.

## TRUST-BASED SECURITY

Authentication-based security and privacy schemes are inadequate in open systems, where principals might be able to provide authentication but are otherwise unknown to the system and thus not authorizable for specific actions. Traditional role-based approaches also fare poorly.

One solution is to base security and privacy decisions on trust-related attributes for which a principal can provide evidence. Examples of this approach include self-evident properties that any observer can reliably

sense, such as a request originating at an IP address assigned to a .gov host; proof of key attributes; signed statements from a trusted source delegating permission; or undertaking an obligation in return for access (http://doi.ieeecomputersociety.org/10.1109/2.970591).

Human societies use trust and reputation to empower persons without preestablished rights, and social networks are an important way of transferring these attributes. However, societal norms, constraints, and rules frequently overlap. Providing effective support for information sharing and control in open environments requires developing computational analogs for these complex social mechanisms similar to Rei's sanctions and conditional premissions.

Consider, for example, a hospital privacy policy that lets doctors at affiliated hospitals access a patient's records as well as share those records with other doctors as long as they first notify the patient.

The first part of the policy requires doctors to prove that they are on staff—for example, via a signed statement from the hospital, a certificate from the hospital board, or delegation from the hospital administrator—and then verify that the hospital is an affiliated one. The second part of the policy is a conditional permission that lets doctors share patient information as long as they meet the associated obligation in the future.

An example of a system that uses trust-based security is the Policy Aware Web project (http://policyawareweb.org). PAW is developing a general-purpose policy framework for the Web that lets users define trust-based policies in their own policy languages—or reuse/extend existing languages—and over their own domain information. PAW provides uniform mechanisms for reasoning over and enforcing access control policies for Web resources.

An example of a PAW policy is: "All specialists I've worked with in the past can access my personal notes on any patient about whom I've consulted them." Specialists who want to access these personal notes must prove to the PAW framework that they have satisfied these conditions.

Information networks are evolving into more open and dynamic systems. Security and privacy enforcement is problematic in these systems due to the lack of a common understanding of requirements and information as well as user unpredictability. Shared ontologies, declarative policies, and trust models offer the most promising approaches to meet these challenges.

*The authors received support for this work from the National Science Foundation. Any opinions, findings, conclusions, or recommendations expressed in this article are those of the authors and do not necessarily reflect the views of the NSF.*

***Lalana Kagal** is a postdoctoral associate in the Computer Science and Artificial Intelligence Laboratory at the Massachusetts Institute of Technology. Contact her at lkagal@csail.mit.edu.*

***Tim Finin** is a professor in the Computer Science and Electrical Engineering Department at the University of Maryland, Baltimore County. Contact him at finin@cs.umbc.edu.*

***Anupam Joshi** is a professor in the Computer Science and Electrical Engineering Department at the University of Maryland, Baltimore County. Contact him at joshi@cs.umbc.edu.*

***Sol Greenspan** is a program director at the National Science Foundation and a research affiliate at the MIT Computer Science and Artificial Intelligence Laboratory. Contact him at sgreensp@nsf.gov or sgreensp@mit.edu.*

**Editor: Jack Cole, US Army Research Laboratory's Information Assurance Center, jack.cole@ieee.org; http://msstc.org/cole**