Moving from Security to Distributed Trust in Ubiquitous Computing Environments

Lalana Kagal, Tim Finin and Anupam Joshi
University of Maryland Baltimore County
email : lkagal1,finin,ajoshi@cs.umbc.edu

1 Introduction

Traditionally, security for stand-alone computers and small networks was handled by physical security and by logging into computers and domains. With open networks like the *Internet* and *pervasive environments*, issues concerning security and trust become crucial. There is no longer the physical aspect of security due to the distributed nature of the networks and the concept of user authentication to a domain is not possible. Imagine a scenario where a user, with a portable device, walking through a building, switches on the lights in the corridor and lowers the temperature of the room that he/she enters. This is an example of pervasive/ubiquitous environments that will soon be a reality. In these *ubiquitous computing* environments users expect to access resources and services anytime and anywhere, leading to serious security risks and problems with access control as these resources can now be accessed by almost anyone with a mobile device. Adding security to such open models is extremely difficult with problems at many levels. We can not assume an architecture with a central authority and access control is required for foreign users. The portable hand-held and embedded devices involved have severe limitations in their processing capabilities, memory capacities, software support and bandwidth characteristics. Moreover, there is currently a great deal of heterogeneity in the hardware and software environments and this is likely to continue for the foreseeable future. Finally, in such an open, heterogeneous, distributed environment there is a great likelihood that inconsistent interpretations will be made of the security information in different domains.

Existing security infrastructures deal with authentication and access control. These mechanisms are inadequate for the increased flexibility required by distributed networks. We suggest enhancing security by the addition of *trust*, which is similar to the way security is handled in human societies. A person is trusted if someone we trust, says that the person can be trusted. In terms of distributed computing, a user is allowed to access a service or information, if the user has the access right to do so, or if the user has been *delegated* the ability by a trusted authority. Trust management can be viewed as developing of security policies, the assignment of credentials to entities, checking if the credentials fulfill the policy and the delegation of trust to third parties [8, 3]. We propose a lightweight solution for trust management that is applicable for the *Internet*, which we are tailoring for pervasive computing environments.

2 Pervasive Computing

Pervasive Computing strives to simplify day-to-day life by providing the means of carrying out personal and business tasks via portable and embedded devices. These tasks could be as simple as switching

This research was supported in part by the IBM EECOMS program, the DARPA DAML program under contract F30602-97-1-0215, NSF CCR0070802, 1159875433. To appear in IEEE Computer, December 2001

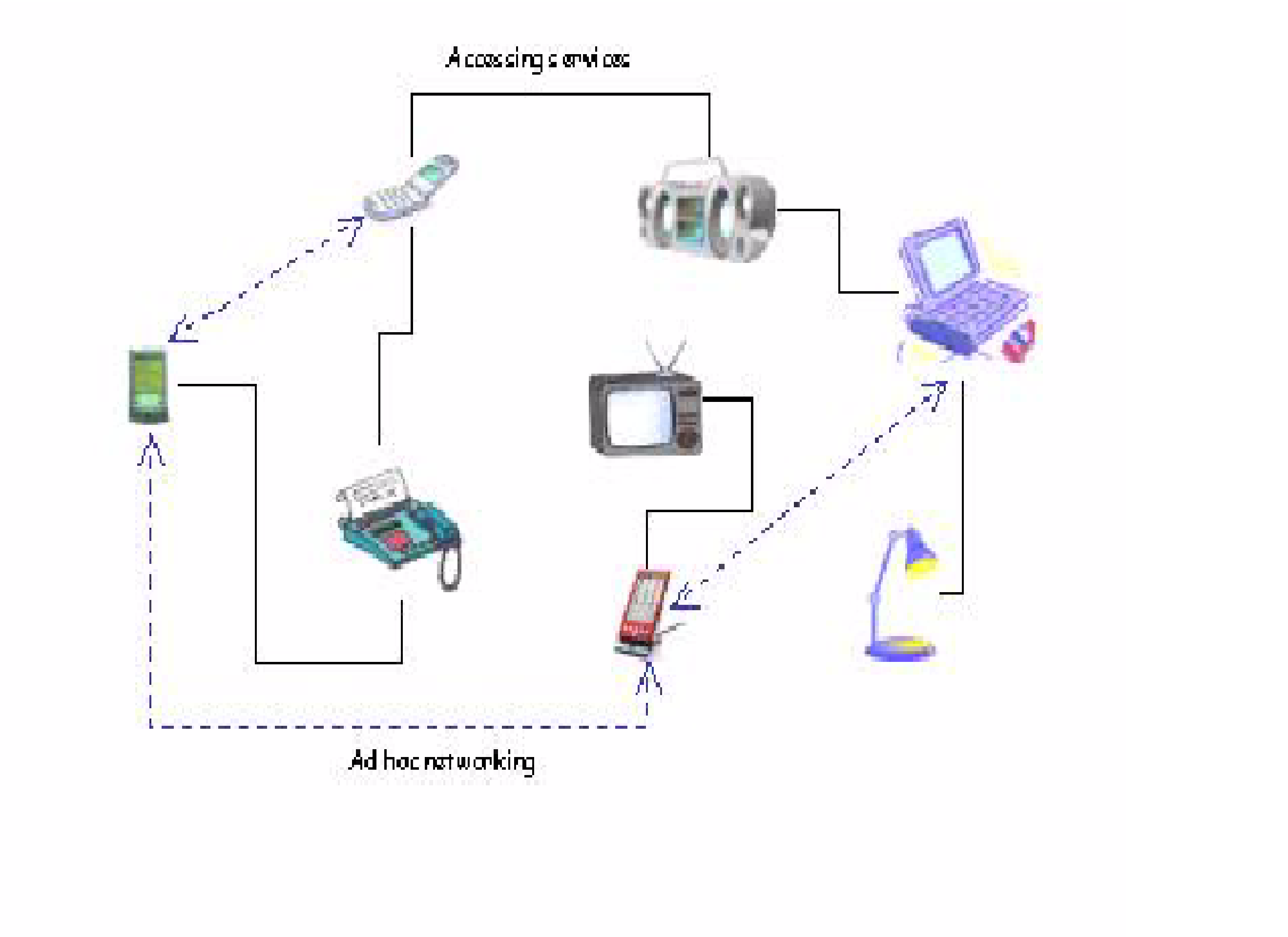

Figure 1. Pervasive Computing

on the lights in a conference room, checking email, organizing meetings, accessing services in a room, to booking airline tickets, buying and selling stock, and even managing bank accounts. As seen in Figure 1., hand-held and embedded devices work within a ubiquitous network infrastructure to provide more relevant information and services to the user.

Our research program* is aimed at realizing ubiquitous computing systems that are composed of autonomous, intelligent, articulate and social components. As part of this research project, we have developed *Centaurus* [7], which realizes the *Smart Office* scenario, where intelligent services are accessible to mobile users via hand-held devices connected over short range wireless links.

We encountered several problems with security for *Centaurus*. Firstly, it is not possible to have a central authority for a single building, or even a group of rooms. So we have to use a distributed model, where the *service managers*, each of which are responsible for a subset of services, are arranged in a hierarchy. It is also not sufficient to authenticate users because most users are foreign to the system, i.e. they are not known. So there is no means of providing access control. Consider a *Centaurus Smartroom* in an office, equipped with an MP3 player, fax machine, several lights, a coffee maker and a printer. If a user, John, walks, how does the room decide which services John has the right to access. Just authenticating John's certificate gives no information on access control because John is an unknown user. Unless it is known in advance which users are going to access the room and their access rights are also known, simple authentication and access control is not going work. Assume John does not work in the office, but in one of its partner firms. How will the system decide whether to allow him to use certain services ? *Distributed Trust* is the solution. Some authorized person in the office can *delegate* the use of

*web pages describing the *UMBC Ebiquity Group*, including descriptions of faculty, students, research projects and papers, can be found at http://research.ebiquity.org/.

the services in the room to John for the period during which he is in the office.

As simple security does not satisfy all the requirements of the pervasive model, we suggest the use of *distributed trust*.

3 Distributed Trust

The distributed trust approach involves articulating policies for authentication, access control and delegation, assigning credentials to individuals, allowing entities to delegate or defer their rights to third parties and providing access control by checking if the initiators credentials fulfill the policies. If an individual has the ability to access a certain service, the individual is said to have the *right* to access the service. If an individual defers a right, he/she possesses, to another individual, it is called a *delegation*, the former is called delegator and the latter delegatee. A user can access a service, if he/she has the right to do so, or if he/she has been delegated the right by an authorized user, who has the ability to delegate.

There has been some work in this area starting with well known models like Simple Public Key Infrastructure (SPKI) [5] and Pretty Good Privacy or PGP [14], to Blaze's Policy Maker [8, 3]. Blaze, who coined the term *Distributed Trust Management*, tries to solve the trust problem by binding public keys to access control without authentication [8, 3]. His PolicyMaker, given a *policy*, answers queries about trust. Though powerful, the policy definition is complicated and not easy to understand for non-programmers who are probably going to develop the policy. Delegation, such as copy/copy propagation issues, have been looked at in operating systems, but they generally dealt with a *known* user domain; all users were known in advance.

Distributed Trust is essentially the establishment of trust by interpreting policies to validate credentials, which can be delegated by authorized users. But how can *distributed trust* be used in pervasive computing ? Next we describe an architecture that uses trust to solve the previously discussed security issues.

4 Trust Architecture

A *security policy* is a set of rules for authorization, access control and trust in a certain domain. All services/users of the domain must enforce its policy and can impose a *local policy* as well. A service being accessed by a foreign user should verify that the user conforms to both its policies. The policy in each domain is enforced by special agents called *security agents*. These agents are part of the Centaurus Service Manager. Users/agents are identified by X.509 [1] authentication certificates. Delegations can be made by authorized agents in the form of signed assertions. Security agents are able to reason about these signed assertions and the security policies to provide access control to the services in their domain. In our system we view 'delegation' as a permission itself. Only an agent with the right to delegate a certain action can actually delegate that action, and the ability to delegate, itself can be delegated. Delegations can be constrained in the policy, by specifying whether an agent can delegate a certain right and to whom it can delegate.

Rights or privileges can be given to trusted agents, who are responsible for the actions of the agents to whom they subsequently delegate the privileges. So the agents will only delegate to agents that they trust. This forms a delegation chain. If any agent along this chain fails to meet the requirements associated with a delegated right, the chain is broken and all agents following the failure are not permitted to perform the action associated with the right [6].

Agents can make requests for a certain service to a security agent controlling the service, and while doing so they attach all their credentials, i.e. ID certificate, authorization certificates etc., to the request. The *security agents* generate authorization certificates, that can be used as 'tickets' to access a certain resource. An agent can also request another agent to delegate to it the right to access a certain service (refer to Figure 2). The latter agent, if satisfied with the requester's credentials may decide to send

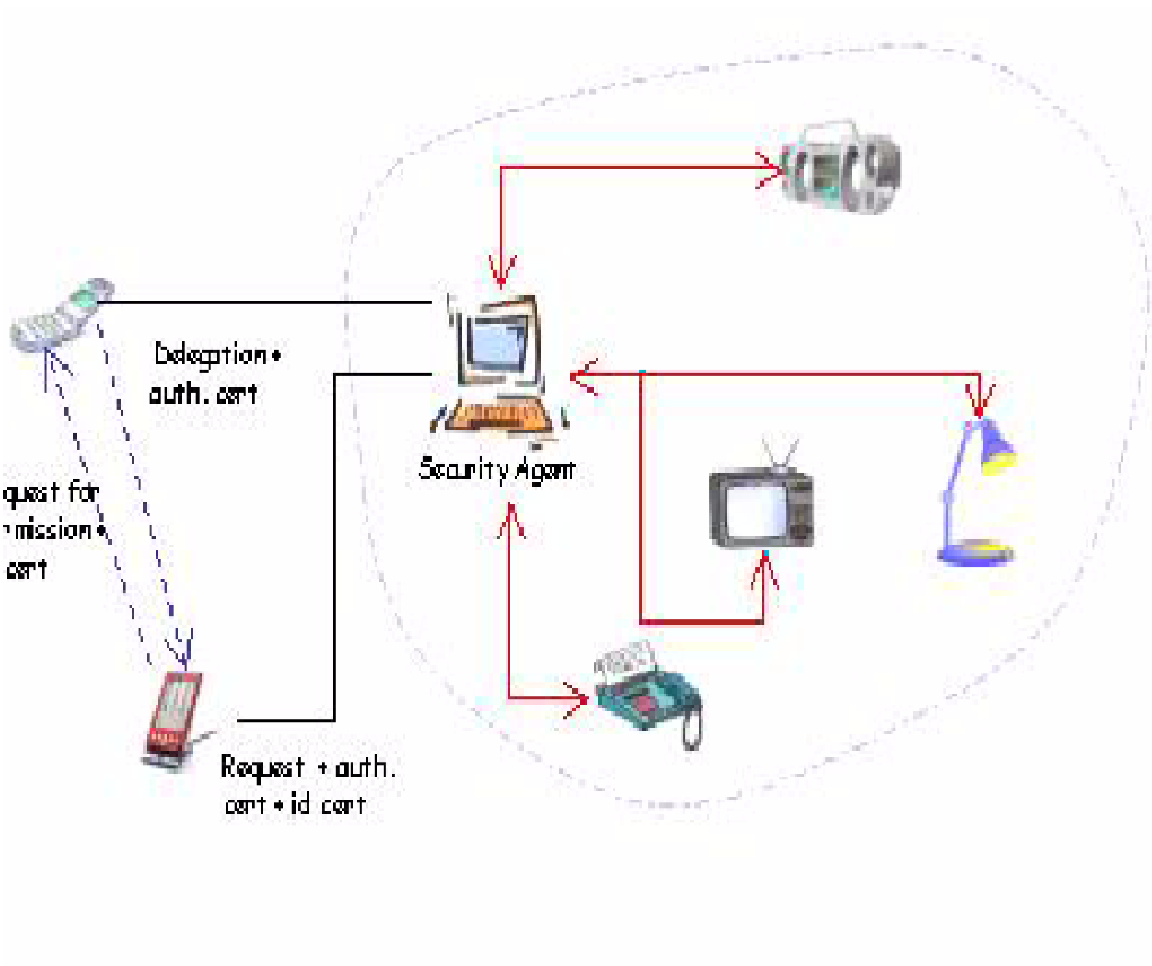

Figure 2. Trust in Pervasive Computing Environments

back a signed statement containing the delegation. The security agent is responsible for honoring the delegation, based on the delegator's and delegatee's credentials and the policies.

The security policy could also contain information about roles of some agents and the abilities associated with certain roles. Our work is related to *Role Based Access Control* [13] in that, an agents access rights are computed from its properties. Our approach is more general, however, because one can use ontologies which include not just role-hierarchies but any properties and constraints which can be defined by declarative horn clauses.

Consider the previous example of John entering a *SmartRoom*. John is an employee of one of the office's partners, but the service manager is unable to understand his role in the organization, so he is denied access to the services. John approaches one of the managers, Susan, and asks for permission to use the services in the *SmartRoom*. According to the policy, Susan has the right to delegate those rights to anyone she trusts. Susan delegates to John, the right to use the lights, the coffee maker and the printer but not the fax machine, for a short period of time. Susan's laptop sends a short lived signed delegation to John's hand-held device. When John enters the room, the client on his hand-held device sends his identity certificate and the delegation to the service manager. As Susan is trusted and has the ability to delegate, the delegation conforms to the policy and John now has access to the lights, the coffee maker and the printer in the room. Once the delegation expires, John is denied access to any service in the room and must ask Susan for another delegation. In this way, a foreign user, John, is allowed access to certain services without creating a new identity for him in the system or assigning a temporary role to him or insecurely opening up the system in anyway. This scenario demonstrates the importance of *trust* over security.

5 Ongoing Work and Conclusion

We are working on integrating trust into the security infrastructure for *Centaurus*. We believe that trust will add a new dimension to pervasive computing, allowing greater flexibility in designing policies and more control over accessing of services and information. At the same time, we are improving our trust architecture. The system is being extended to include entitlements, prohibitions and obligations and the ability to delegate them.

Another important issue with distributed networks is that of privacy. Users do not want their names and actions to be logged, so we are trying to do away with with X.509 certificates and replace them with XML signatures [11] from a *trusted authority* and does not include the identity of the bearer, but only a role or designation.

Our past work on distributed trust represented actions, privileges, delegations and security policy as horn clauses encoded in Prolog. In order to develop a approach that is better suited to sharing information in an open environment, we are recasting this work in DAML [4], the DARPA Agents Markup Language. DAML is built on XML and RDF and provides a description logic language for defining and using ontologies on the web. In applying our framework, one must extend the initial ontology (http://daml.umbc.edu/ontologies/trust-ont.daml) by defining domain-specific classes for actions, roles, privileges, etc. and creating appropriate instances.

In pervasive computing environments, security plays a very important role. But simple security itself is insufficient because the users are generally unknown and there is no central authority. To make the vision of ubiquitous computing a reality, we firmly believe that *distributed trust* needs to be added to the security infrastructure.

References

[1] Public-key infrastructure (x.509), http://www.ietf.org/html.charters/pkix-charter.html.

[2] Tim Berners-Lee, James Hendler, and Ora Lassila. The semantic web. In *Scientific America*, May 2001.

[3] Matt Blaze, Joan Feigenbaum, and Angelos D. Keromytis. The role of trust management in distributed systems security. In *Secure Internet Programming*, pages 185–210, 1999.

[4] DAML. Daml specification, 2000.

[5] Carl M. Ellison, Bill Frantz, and Brian M. Thomas. Simple public key certificate. Internet document, 1996.

[6] Lalana Kagal, Tim Finin, and Yun Peng. A framework for distributed trust management. In *To appear in proceedings of IJCAI-01 Workshop on Autonomy, Delegation and Control*, 2001.

[7] Lalana Kagal, Vlad Korolev, Harry Chen, Anupam Joshi, and Tim Finin. Centaurus: A framework for intelligent services in a mobile environment. In *Proceedings of International Workshop on Smart Appliances and Wearable Computing (IWSAWC), in the The 21st International Conference on Distributed Computing Systems (ICDCS-21) April 16-19, 2001*.

[8] M.Blaze, J.Feigenbaum, and J.Lacy. Decentralized trust management. *IEEE Proceedings of the 17th Symposium*, 1996.

[9] J. Steiner, C. Neuman, and J. Schiller. An authentication service for open network systems. *In Proceedings of the USENIX Winter Conference, pages 191–202, February 1988.*, 1988.

[10] Blue tooth website. The official bluetooth website http://www.bluetooth.com/, 2001.

[11] W3C. Xml signature http://www.w3.org/signature/.

[12] Mark Weiser. The computer for the 21 st century. *Scientific American*, 265(3):94–104, 1991.

[13] Longhua Zhang, Gail-Joon Ahn, and Bei-Tseng Chu. A rule-based framework for role-based delegation. *6th ACM Symposium on Access Control Models and Technologies (SACMAT), Chantilly, VA, May 3-4*, 2001.

[14] Philip R. Zimmermann. *The Official PGP User's Guide*. MIT Press, Cambridge, MA, USA, 1995.

6 Appendix

We briefly describe some of the terms that are used in the paper in following table.

Term	*Explanation*
Authentication	involves verifying the identity of a person or process. Popular authentication techniques in open environments include Kerberos tickets [9] and digital certificates (e.g., X.509 [1]).
Access	is the ability to do something with a computer resource (e.g., use, change, or view) and *Access Control* is the means by which the ability is explicitly enabled or restricted in some way (usually through physical and system-based controls).
Role-Based Access Control	is an approach in which access decisions are based on the roles that individual users have as part of an organization, such as doctor, nurse, teller, manager, student etc.
Distributed trust	can be viewed as developing of security policies, the assignment of credentials to entities, verifying if the credentials fulfill the policy and the delegation of trust to third parties [8, 3].
Deontic logic	is a branch of logic that deals with reasoning pertaining to normative matters like permissions, obligations, entitlements, prohibitions.
Pervasive computing	is the set of technologies for developing highly interactive environments that allow mobile users to access information and integrated services via hand-held devices. The *pervasive computing environments* of the near future [12] will involve the interactions, coordination and cooperation of numerous, casually accessible, and often invisible computing devices. These devices, whether carried on our person or embedded in our homes, businesses and classrooms, will connect via wireless and wired links to one another and to the global networking infrastructure.
Bluetooth	is a specification for short range radio links between portable devices [10].
Semantic Web	is an approach for expressing information available on the *Internet* in a machine readable form [2].
DAML	is an extension to XML and the Resource Description Framework (RDF), which is being developed to markup information in machine readable form [4].

Table 1. Appendix