Towards Security and Privacy for Pervasive Computing*

Roy Campbell[1], Jalal Al-Muhtadi[1], Prasad Naldurg[1], Geetanjali Sampemane[1], M. Dennis Mickunas[1]

[1]Department of Computer Science, University of Illinois at Urbana Champaign, 1304 W. Springfield Ave., Urbana, IL 61801.
{rhc, almuhtad, naldurg, geta, mickunas}@cs.uiuc.edu

Abstract. Pervasive computing environments with their interconnected devices and services promise seamless integration of digital infrastructure into our everyday lives. While the focus of current research is on how to connect new devices and build useful applications to improve functionality, the security and privacy issues in such environments have not been explored in any depth. While traditional distributed computing research attempts to abstract away physical location of users and resources, pervasive computing applications often exploit physical location and other context information about users and resources to enhance the user experience. The need to share resources and collaborate introduces new types of interaction among users as well as between the virtual and physical worlds. In this context, it becomes difficult to separate physical security from digital security. Existing policies and mechanisms may not provide adequate guarantees to deal with new exposures and vulnerabilities introduced by the pervasive computing paradigm. In this paper we explore the challenges for building security and privacy into pervasive computing environments, describe our prototype implementation that addresses some of these issues, and propose some directions for future work.

1 Introduction

We are witnessing the birth of a revolutionary computing paradigm that promises to have a profound effect on the way we interact with computers, devices, physical spaces, and other people. This new technology envisions a world where embedded processors, computers, sensors, and digital communications are inexpensive commodities that are available everywhere. This eliminates time and place barriers by making services available to users anytime and anywhere.

Pervasive computing will surround users with a comfortable and convenient information environment that merges physical and computational infrastructures into an integrated habitat. This habitat will feature a proliferation of hundreds or thousands of computing devices and sensors that will provide new functionality, offer specialized services, and boost productivity and interaction. Context-awareness will

* This research is supported by a grant from the National Science Foundation, NSF CCR 0086094 ITR and NSF 99-72884 EQ.

allow this habitat to take on the responsibility of serving users, by tailoring itself to their preferences as well as performing tasks and group activities according to the nature of the physical space. We term this dynamic, information-rich habitat an "active space." Within this space, individuals may interact with flexible applications that may follow the user, define and control the function of the space, or collaborate with remote users and applications.

The realization of this computing paradigm is not far fetched. An average person today already owns vast numbers of consumer devices, electronic gadgets, and gizmos that already have processors, microcontrollers, and memory chips embedded into them, like VCRs, TVs, washers and dryers. The vehicles we use on daily basis already have a large number of embedded computers handling different subsystems of the vehicle, like ABS (Anti-lock Braking System) and ESP (Electronic Stability Program). Technologies like Bluetooth [1] and Wi-Fi [2] make it possible to embed networking capabilities into any small devices without hassle. In effect, these technologies help make networking much more general and achievable even on elementary devices, like toasters and paperclips.

1.1 Pervasive Computing Abstractions

To have a better understanding of the challenges associated with securing pervasive computing environments, it is important to list the salient features of pervasive computing. These include the following.

Extending Computing Boundaries. While traditional computing encompassed hardware and software entities, pervasive computing extends the boundaries of computing to include physical spaces, building infrastructures, and the devices contained within. This aims to transform dull spaces into interactive, dynamic, and programmable spaces that are coordinated through a software infrastructure and populated with a large number of mobile users and devices.

Invisibility and non-intrusiveness. In current computing models, computers are still the focus of attention. In effect, people have to change some of their behavior and the way they perform tasks so that these tasks can be computerized. To boost productivity, it is important that computing machinery disappear and leave the spotlight. Computers should blend in the background allowing people to perform their duties without having machines at the center of their focus.

Creating smart and sentient spaces. A dust of invisible embedded devices and sensors are incorporated to turn physical spaces into active, smart surroundings that can sense, "see," and "hear," effectively, making the space sentient and adaptable. Ultimately, the space should become intelligent enough to understand users' intent and become an integral part of users' everyday life.

Context awareness. A pervasive computing environment should be able to capture the different context and situational information and integrate them with users and devices. This allows the active space to take on the responsibility of serving users and automatically tailoring itself to meet their expectations and preferences.

Mobility and adaptability. To be truly omnipresent, the pervasive computing environment should be as mobile as its users. It should be able to adapt itself to

environments with scarce resources, while being able to evolve and extend once more resources become available.

1.2 The Problem

Current research in pervasive computing focuses on building infrastructures for managing active spaces, connecting new devices, or building useful applications to improve functionality. Security and privacy issues in such environments, however, have not been explored in depth. Indeed, several researchers and practitioners have admitted that security and privacy in this new computing paradigm are real problems. Langheinrich [3, 4] warns us about the possibility of an Orwellian nightmare in which current pervasive computing research continues on without considering privacy in the system. Stajano [5] notices that while researchers are busy thinking about the killer applications for pervasive computing, cyber-criminals and computer villains are already considering new, ingenious attacks that are not possible in traditional computing environments. Kagal et al. [6, 7] admit that securing pervasive computing environments presents challenges at many levels.

The very same features that make pervasive computing environments convenient and powerful make them vulnerable to new security and privacy threats. Traditional security mechanisms and policies may not provide adequate guarantees to deal with the new exposures and vulnerabilities.

In this paper we address some of these issues as follows. In Section 2 we explore the challenges for building security and privacy into pervasive computing environments. In Section 3 we describe our prototype implementation that addresses some of these issues. In Section 4 we propose some directions for future work and conclude.

2 Security Challenges and Requirements

In this section, we talk about the major challenges and requirements for securing pervasive computing environments.

2.1 Challenges

As mentioned before, the additional features and the extended functionality that pervasive computing offers make it prone to additional vulnerabilities and exposures. Below, we mention these features that add extra burden to the security subsystem.

2.1.1. The Extended Computing Boundary

Traditional computing is confined to the virtual computing world where data and programs reside. Current distributed computing research tends to abstract away physical locations of users and resources. Pervasive computing, however, extends its reach beyond the computational infrastructure and attempts to encompass the

surrounding physical spaces as well. Pervasive computing applications often exploit physical location and other context information about users and resources to enhance the user experience. Under such scenarios, information and physical security become interdependent. As a result, such environments become prone to more severe security threats that can threaten people and equipment in the physical world as much as they can threaten their data and programs in the virtual world. Therefore, traditional mechanisms that focus merely on digital security become inadequate.

2.1.2. Privacy Issues

The physical outreach of pervasive computing makes preserving users' privacy a much more difficult task. Augmenting active spaces with active sensors and actuators enables the construction of more intelligent spaces and computing capabilities that are truly omnipresent. Through various sensors and embedded devices, active spaces can automatically be tailored to users' preferences and can capture and utilize context information fully. Unfortunately, this very feature could threaten the privacy of users severely. For instance, this capability can be exploited by intruders, malicious insiders, or even curious system administrators to track or electronically stalk particular users. The entire system now becomes a distributed surveillance system that can capture too much information about users. In some environments, like homes and clinics, there is usually an abundance of sensitive and personal information that must be secured. Moreover, there are certain situations when people do not want to be tracked.

2.1.3. User Interaction Issues

One of the main characteristics of pervasive applications is a richer user-interface for interaction between users and the space. A variety of multimedia mechanisms are used for input and output, and to control the physical aspects of the space. At any point of time, the set of users in the space affects the security properties of the space. Because of the nature of these interactions, users in the space cannot easily be prevented from seeing and hearing things happening in it, so this has to be taken into account while designing access control mechanisms. We believe that the access control mechanisms should allow groups of users and devices to use the space in a manner that facilitates collaboration, while enforcing the appropriate access control policies and preventing unauthorized use. Thus the physical and "virtual" aspects of access control for such spaces have to be considered together.

2.1.4. Security Policies

It is important in pervasive computing to have a flexible and convenient method for defining and managing security policies in a dynamic and flexible fashion. Policy Management tools provide administrators the ability to specify, implement, and enforce rules to exercise greater control over the behavior of entities in their systems. Currently, most network policies are implemented by systems administrators using tools based on scripting applications [8, 9] that iterate through lists of low-level interfaces and change values of entity-specific system variables. The policy management software maintains an exhaustive database of corresponding device and resource interfaces. With the proliferation of heterogeneous device-specific and

vendor-specific interfaces, these tools may need to be updated frequently to accommodate new hardware or software, and the system typically becomes difficult to manage. As a result, general purpose low-level management tools are limited in their functionality, and are forced to implement only generic or coarse-grained policies [10].

Since most policy management tools deal with these low-level interfaces, administrators may not have a clear picture of the ramifications of their policy management actions. Dependencies among objects can lead to unexpected side effects and undesirable behavior [11]. Further, the disclosure of security policies may be a breach of security. For example, knowing whether the system is on the lookout for an intruder could actually be a secret. Thus, unauthorized personnel should not be able to know what the security policy might become under a certain circumstance.

2.1.5. Info Ops

There is a great deal of concern over new types of threats, namely, Information Operations (info ops) and cyber-terrorism, which are natural consequences of the increasing importance of electronic information and the heavy reliance on digital communication networks in most civilian and military activities. Info ops, which can be defined as "actions taken that affect adversary information and information systems while defending one's own information and information systems," [12] is a serious concern in today's networks. In such a scenario, cyber-terrorists and other techno-villains can exploit computer networks, inject misleading information, steal electronic assets, or disrupt critical services. Pervasive computing gives extreme leverage and adds much more capabilities to the arsenal of info warriors, making info ops a much more severe threat.

2.2 Security Requirements

To deal with the new vulnerabilities introduced by pervasive computing, security and privacy guarantees in pervasive computing environments should be specified and drafted early into the design process rather than being considered as add-ons or afterthoughts. Previous efforts in retrofitting security and anonymity into existing systems have proved to be inefficient and ineffective. The Internet and Wi-Fi are two such examples both of which still suffer from inadequate security. In this section, we briefly mention the important requirements needed for a security subsystem for pervasive computing environments.

2.2.1. Transparency and unobtrusiveness

The focal point of pervasive computing is to transform users into first class entities, who no longer need to exert much of their attention to computing machinery. Therefore, even the security subsystem should be transparent to some level, blending into the background without distracting users too much.

2.2.2. Multilevel

When it comes to security, one size does not fit all. Hence, the security architecture deployed should be able to provide different levels of security services based on system policy, context information, environmental situations, temporal circumstances, available resources, etc. In some instances, this may go against the previous point. Scenarios which require a higher-level of assurance or greater security may require users to interact with the security subsystem explicitly by, say, authenticating themselves using a variety of means to boost system's confidence.

2.2.3. Context-Awareness

Often, traditional security is somewhat static and context insensitive. Pervasive computing integrates context and situational information, transforming the computing environment into a sentient space. The security aspects of it are no exceptions. Security services should make extensive use of context information available. For example, access control decisions may depend on time or special circumstances. Context data can provide valuable information for intrusion detection mechanisms. The principal of "need to know" should be applied on temporal and situational basis. For instance, security policies should be able to change dynamically to limit the permissions to the times or situations when they are needed. However, viewing what the security policy might become in a particular time or under a particular situation should not be possible. In addition, there is a need to verify the authenticity and integrity of the context information acquired. This is sometimes necessary in order to thwart false context information obtained from rogue or malfunctioning sensors.

2.2.4. Flexibility and customizability

The security subsystem should be flexible, adaptable, and customizable. It must be able to adapt to environments with extreme conditions and scarce resources, yet, it is able to evolve and provide additional functionality when more resources become available. Tools for defining and managing policies should be as dynamic as the environment itself.

2.2.5. Interoperability

With many different security technologies surfacing and being deployed, the assumption that a particular security mechanism will eventually prevail is flawed. For that reason, it is necessary to support multiple security mechanisms and negotiate security requirements.

2.2.6. Extended boundaries

While traditional security was restricted to the virtual world, security now should incorporate some aspects of the physical world, e.g. preventing intruders from accessing physical spaces. In essence, virtual and physical security become interdependent.

2.2.7. Scalability

Pervasive computing environments can host hundreds or thousands of diverse devices. The security services should be able to scale to the "dust" of mobile and embedded devices available at some particular instance of time. In addition, the security services need to be able to support huge numbers of users with different roles and privileges, under different situational information.

In the following section, we suggest solutions that address some of the issues mentioned above.

3 Case Study: Gaia OS Security

Gaia [13-15] is a component-based, middleware operating system that provides a generic infrastructure for constructing pervasive computing environments. Gaia provides the necessary core services to support and manage active spaces and the pervasive applications that run within these spaces. By using Gaia, it is possible to construct a multipurpose, prototype active space. This active space contains state-of-the-art equipment, including a surround audio system, four touch plasma panels with HDTV support, HDTV video wall, X10 devices, electronic white boards, IR beacons, Wi-Fi and Bluetooth access points, video cameras, and flat panel desktop displays. Currently, this active space is used for group meetings, seminars, presentations, demos, and for entertainment purposes (like listening to music and watching movies). The different uses of this active space translate into different contexts. In this active space, we deployed several security mechanisms that addresses some of the issues mentioned in this paper.

3.1 Gaia Authentication

Authentication mechanisms in pervasive computing environments should strike a balance between authentication strength and non-intrusiveness. A smart badge that broadcasts short range radio signals, for instance, is a good non-intrusive authentication mechanism; however, it only provides weak authentication. A challenge-response mechanism provides stronger authentication, but often at the expense of additional interactions on behalf of the user. We let context decide how strong the authentication needs to be. This allows the authentication process to enable principals to authenticate themselves to the system using a variety of means. These include the use of wearable devices, face recognition, smart badges, fingerprint identification, retinal scans, etc. To enable this, we differentiate between different strengths of authentication by associating *confidence values* to each authentication process. This confidence value represents how "confident" the authentication system is about the identity of the principal. We represent this by a number between 0 and 1. This confidence value is based on the authentication device and the authentication protocol used. Principals can employ multiple authentication methods in order to increase the confidence values associated with them. Access control decisions can now become more flexible by utilizing confidence information. Several reasoning techniques can be used to combine confidence values and calculate a net confidence

value for a particular principal. The techniques we have considered so far include simple probabilities, Bayesian probability, and fuzzy logic [16]. The authentication service is managed by authentication-related policies. These policies are expressed as rules in first order predicate logic. The logic used includes temporal and fuzzy operators that allow the policies to capture context or temporal information, like revocation of authentication credentials under certain circumstances and so on.

Since there are a large number of diverse devices that can be deployed for identification and authentication purposes, and as technology advances, we expect many new authentication devices to become available. This makes it necessary to have dynamic means for adding new authentication devices and associating them with different capabilities and protocols. Naturally, some means of authentication are more reliable and secure than others. For example, it is easy for smart badges to be misplaced or stolen. On the other hand, the use of biometrics, retinal scans for instance, is a fairly good means of authentication that is difficult to forge. Because of the various authentication methods and their different strengths, it is sensible to accommodate different levels of confidence and incorporate context and sensor information to infer more information or buildup additional confidence in a principal's identity. Further, the same techniques can assist in detecting intruders and unauthorized accesses and assessing their threat level.

The various means of authenticating principals and the notion of different confidence levels associated with authenticated principals constitute additional information that can enrich the context awareness of smart spaces. In a later section, we illustrate how such information is inferred and exchanged with other Gaia core services.

To meet the stated requirements we propose a federated authentication service that is based on distributed, pluggable authentication modules. Fig. 1 provides a sketch of the authentication architecture that incorporates the objectives mentioned above. PAM (Pluggable Authentication Module) [17] provides an authentication method that allows the separation of applications from the actual authentication mechanisms and devices. Dynamically pluggable modules allow the authentication subsystem to incorporate additional authentication mechanisms on the fly as they become available. The Gaia PAM (GPAM) is wrapped by two APIs. One interface is made available for applications, services, and other Gaia components, to request authentication of entities or inquire about authenticated principals. Since the authentication service can be running anywhere in the space (possibly federated) we use CORBA facilities to allow the discovery and remote invocation of the authentication services that serve a particular smart space. The authentication modules themselves are divided into two types: Gaia Authentication Mechanisms Modules (AMM), which implement general authentication mechanisms or protocols that are independent of the actual device being used for authentication. These modules include a Kerberos authentication module, a SESAME [18] authentication module, the traditional username/password-based module, a challenge-response through a shared secret module, etc. The other type of modules is the Authentication Device Modules (ADM). These modules are independent of the actual authentication protocol; instead, they are dependent on the particular authentication device.

This decoupling enables greater flexibility. When a new authentication protocol is devised, an AMM module can be written and plugged in to support that particular

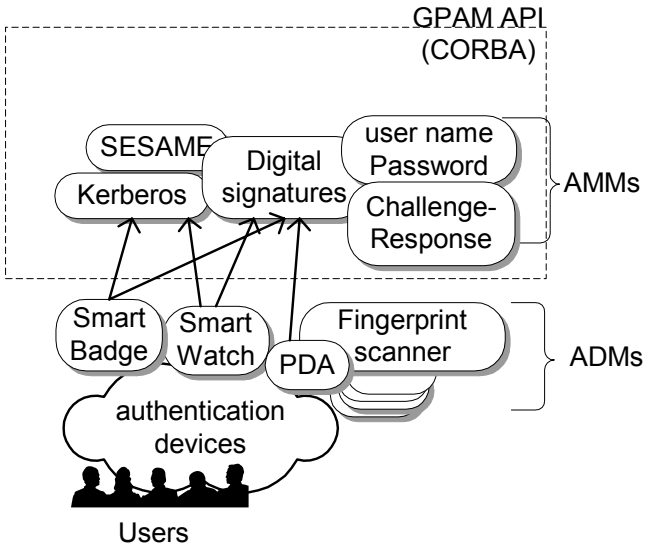

Fig. 1: Gaia Authentication Service

protocol. Devices that can capture the information required for completing the protocol can use the new authentication module with minimal changes to their device drivers. When a new authentication device is incorporated to the system, a new ADM module is implemented in order to incorporate the device into the active space. However, the device can use existing security mechanisms by using CORBA facilities to discover and invoke authentication mechanisms that are compatible with its capabilities. In effect, this creates an architecture similar to PAM and is also federated through the use of CORBA. Many CORBA implementations are heavyweight and require significant resources. To overcome this hurdle, we used the Universally Interoperable Core (UIC), which provides a lightweight, high-performance implementation of basic CORBA services [19].

3.2 Mist – Privacy Communication

To address the privacy problems in pervasive computing, we introduce Mist [20, 21] a general communication infrastructure that preserves privacy in pervasive computing environments. Mist facilitates the separation of location from identity. This allows authorized entities to access services while protecting their location privacy. Here, we just give a brief overview of how Mist works. Mist consists of a privacy-preserving hierarchy of *Mist Routers* that form an overlay network, as illustrated in Fig. 2. This overlay network facilitates private communication by routing packets using a *hop-by-hop*, *handle-based routing* protocol. We employ public key cryptography in the initial setup of these handles. These techniques make communication infeasible to trace by eavesdroppers and untrusted third parties.

A handle is an identifier that is unique per Mist Router. Every incoming packet has an "incoming handle" that is used by the Mist Router to identify the next hop to which to forward the packet. The incoming handle is replaced by an outgoing handle before the packet is transmitted to the next hop. This hop-by-hop routing protocol allows a Mist Router to forward the packet to the next hop, while hiding the original source and final destination. In effect, this process creates "virtual circuits" over which data can flow securely and privately.

Mist introduces *Portals* that are installed at various locations in the pervasive computing environment. These Portals are devices capable of detecting the presence of people and objects through the use of base stations or sensors. However, they are incapable of positively identifying the users. To effectively hide a user's location, we introduce a special Mist Router referred to as a *Lighthouse*. A user registers with this Lighthouse, which allows packets to be routed to and from the user. The Lighthouse

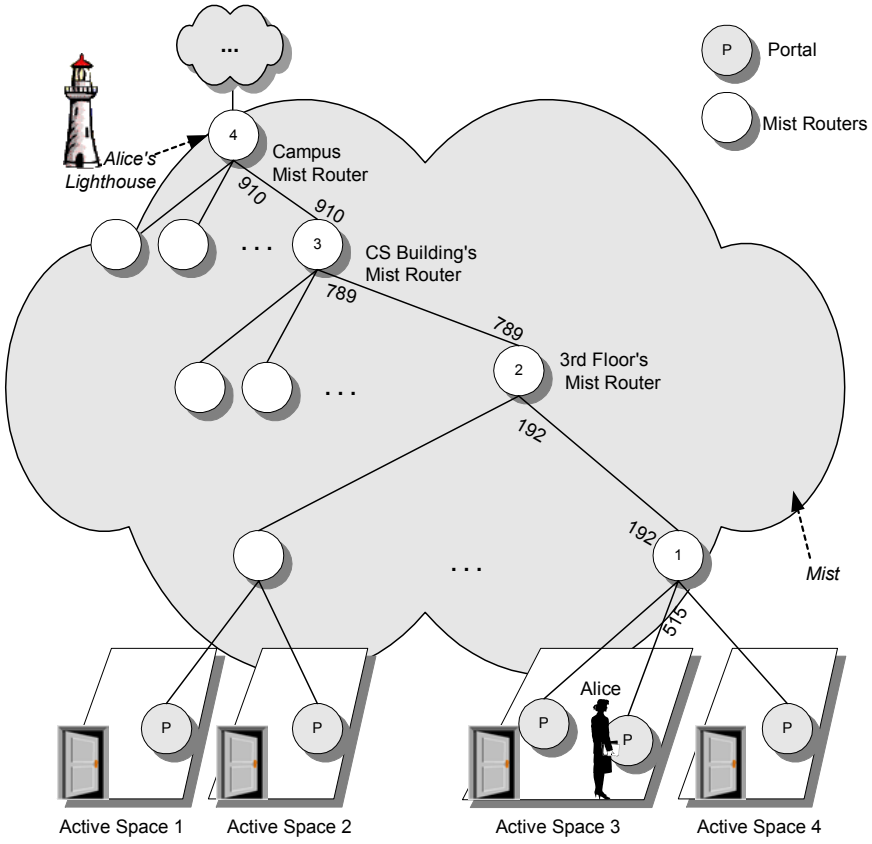

Fig. 2. The Mist communication protocol.

of a user exists at a "higher level" in the hierarchy, high enough not to be able to deduce the actual physical location of the user. However, the Lighthouse is kept in the dark about the actual physical location of the user (thanks to the hop-by-hop routing protocol).

To illustrate, in Fig. 2 Alice, who is in active space 3, is detected by the Portal in that space. The Portal only detects Alice's badge ID (or other information embedded into other devices that Alice is carrying or wearing) however, this information alone is insufficient to indicate that this is actually Alice. The campus Mist Router is designated as Alice's Lighthouse. A secure channel between Alice's devices and her Lighthouse is established, going through the Portal, node 1, node 2, node 3, and finally node 4. To prevent private information from leaking, encryption is employed. The numbers over the links shown in the figure represents the handles. As depicted, the handles are valid only over a single hop. The intermediate nodes translate an incoming handle to an outgoing one (e.g. Mist Router 1 in the figure translates the incoming handle 515 to outgoing handle 192). Thus, intermediate Mist Routers can only route to the next hop correctly, but do not know the actual destination or source. Mist distributes the trust to better preserve the privacy. Only if enough intermediate Mist Routers collude, can the true location of Alice be found. Note that in the example, Alice's Lighthouse can only infer that Alice is located somewhere within the campus. Mist provides a customizable level of privacy. A user can enjoy better location privacy if he or she chooses a Lighthouse that is higher in the hierarchy, e.g. choosing the campus Lighthouse as opposed to the CS building Lighthouse. Since users can register anonymously with Portals, their anonymity is preserved.

3.3 Dynamic Security Policies

To address the new challenges in defining and managing security policies in pervasive computing environments, we propose a new class of policies called dynamic policies [22-24] that are designed with explicit knowledge of system behavior, focusing on the interactions between various system objects. We develop behavioral descriptions of programs that can be sent across networks to change a system's software state, while preserving certain security and privacy properties. These program modules correspond to the dynamic policy implementation, and can be enforced by executing them in a suitable software context.

We explore the nature of security guarantees that can be made about the system state before, during, and after the execution of dynamic policies. We believe that security concerns need to be integrated into models of system behavior, and security policies have to form an integral part of system specifications. This research is crucial in the context of active spaces and in other dynamic environments where operational parameters are constantly changing. Dynamic policies enable the creation of customizable programs that can be deployed on-the-fly, to enforce and implement strong security policies that can adapt to a changing software environment. In [24] we present a powerful set of formal methods and mechanisms that can be used to create policies with strong security guarantees, eliminating guesswork in the design and deployment of reactive security systems.

To illustrate this procedure, we present the policy development life-cycle for dynamic access control policies. We construct dynamic access control policies by building a formal behavioral model of an access control system. The state of the system in our formal specification includes all entities that participate in an access control decision. This includes sets of all subjects, objects, and corresponding permissions. The specification of the system encompasses the dynamic behavior of the state variables. This is specified by state transitions that correspond to all methods in our implementation that change the list of subjects, objects and their corresponding permissions.

The set of access rights triples (subject, object, permissions) forms a conceptual access control matrix. Given the behavioral specification of our access control system, we define what security properties we want to preserve when the system state changes dynamically. An access control model is secure if and only if, starting from an empty (or safe) access control matrix, the state transitions in the model add only authorized access rights to the matrix. In other words, an access control model is secure if all access rights are authorized. This property has to be preserved by the access control system even when the state of the system changes dynamically, due to users and devices entering and leaving an active space. The definition of authorized access right depends on the type of access control policy. For example in MAC (Mandatory access control) system, only an administrator is authorized to add a new access right to the system. In DAC (Discretionary Access Control) object owners can add access rights.

In order to enforce the access control safety property, we annotate the specification we built earlier with authorization proofs. These proofs are subroutines that use credentials to attest the ownership of an object by a subject (for DAC) or the type of subject (for MAC). The credentials are cryptographically protected by digital signatures and/or encryption. All state transitions in the access control specification are rewritten as guarded commands. The guards verify access control safety condition, by validating authorization proofs. The commands (that correspond to methods that change state variables) are executed only if the guard can be validated. This annotation (the guard) to each dynamic state transition automatically guarantees the safety properties even when the access matrix is allowed to change dynamically, preserving the security of the system at all times.

Similar to access control safety, we have developed dynamic policies for some information flow and availability properties. These security properties are a combination of safety and liveness properties and include temporal quantifiers. At a more fundamental level, we argue that dynamic environments require dynamic security solutions. Dynamic policies enable administrators to react to vulnerabilities detected by IDS and risk analyzers with greater confidence. By including temporal properties in our design of security policies, we can change our system implementations in a controlled manner, and turn on restrictive attack resilient policies at will, without sacrificing security guarantees. This dynamism also allows us to change back to default policies after the attack has been mitigated, allowing us to implement minimal security solutions on a need to protect basis, and amortize performance penalties.

3.4 Access Control

Smart rooms are typically shared by different groups of users for different activities at different points in time. Each activity-specific incarnation of an active space (such as a "classroom" or a "meeting") is called a "Virtual Space". Access control policies and mechanisms are necessary to ensure that users only use the resources (both hardware and software) in an active space in authorized ways, and to allow shared use of the space. These different virtual spaces have varying access control requirements, and the access control policies and mechanisms should allow seamless transitions between virtual spaces, without sacrificing security properties. In addition, the policies should be easy to configure, enforce, and administer.

We are in the process of developing an access control system [25] for Gaia. Our access control system supports customizable access control policies, and integrates physical and virtual aspects of the environment into the access control decision mechanisms. It can reconfigure an active space dynamically to support different access control policies for different virtual spaces, depending on the set of users and the activity being undertaken in the space. We also provide dynamic support for explicit cooperation between different users, groups of users, and devices.

Our system uses the RBAC model [26] for policy configuration, and implements both discretionary and mandatory access controls, providing flexibility and expressiveness. We define three types of roles viz., system, space, and application roles. Each role can be managed by a different administrator, thus allowing for decentralized administration.

Within each active space, access control policies are expressed in terms of *space roles*. The space administrator sets access control policies for resources within a particular space. These policies are in the form of access lists for resources within the space, expressed in terms of space roles and permissions. When users enter a space, their system role is mapped into an appropriate space role automatically.

We also build an explicit notion of cooperation into our access control model using the concept of the *space mode*. We define four distinct modes of collaboration in our model: individual, shared, supervised-use and collaborative, corresponding to different levels of cooperation between users in the space who do not necessarily trust each other. The mode of the space depends on the users within the space and the activity being performed.

This model of access control is useful in developing access control policies that are appropriate for collaborative applications that are common in such environments.

4 Conclusion and Future Directions

The shift to the pervasive computing paradigm brings forth new challenges to security and privacy, which cannot be addressed by mere adaptation of existing security and privacy mechanisms. Unless security concerns are accommodated early in the design phase, pervasive computing environments will be rife with vulnerabilities and exposures. In this paper we talked about some of these challenges. We also presented some solutions in our prototype implementation.

The construction of complete, integrated pervasive environments and their real-life deployment are still things of the future. Security in pervasive computing is expected to be an integral part of the whole system, which is not realized yet. It should be noted, however, that there is no single "magical" protocol or mechanism that can address all the security issues and meet the requirements and expectations of secure pervasive computing. Moreover, security itself consists of a variety of different and broad aspects each of which requires detailed research and customized solutions. For these reasons, our prototype implementations are not meant to be a solution for all problems. Instead, they represent milestones towards the construction of a full-fledged security subsystem.

Promising future directions include the development of formal specifications of desirable behavior in the form of security and privacy properties in pervasive computing. Access control, information flow, availability, and secure protocols for authentication, non-repudiation, confidentiality and integrity can be specified in terms of system properties such as safety and liveness. It is also promising to incorporate intelligence and automated reasoning into the security architecture. This "intelligent" security system would be able to make judgments and give assistance in securing the environment without too much intervention by users or administrators. Therefore, we are exploring the possibility of incorporating automated reasoning and learning into the active spaces security architecture, enabling it to perform intelligent inferences under different contexts despite the uncertainties that arise as a result of bridging the physical and virtual worlds.

We are also looking into the development of several middleware reflective object-oriented patterns that can support the different aspects of security, including authentication, access control, anonymity, and policy management, as well as how to instantiate them with diverse mechanisms. Finally, because it is difficult to develop security models that involve people and physical spaces, more studies on integrating virtual and physical security need to be considered.

5 References

[1] "Bluetooth." http://www.bluetooth.com/.

[2] "Reference number ISO/IEC 8802-11:1999(E) IEEE Std 802.11, 1999 edition. International Standard [for] Information Technology-Telecommunications and information exchange between systems-Local and metropolitan area networks-Specific Requirements-Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications," *IEEE*, 1999.

[3] M. Langheinrich, "Privacy by Design - Principles of Privacy-Aware Ubiquitous Systems," presented at ACM UbiComp 2001, Atlanta, GA, 2001.

[4] M. Langheinrich, "A Privacy Awareness System for Ubiquitous Computing Environments," presented at 4th International Conference on Ubiquitous Computing, 2002.

[5] F. Stajano, *Security for Ubiquitous Computing*: Halsted Press, 2002.

[6] L. Kagal, T. Finin, and A. Joshi, "Trust-Based Security in Pervasive Computing Environments," *IEEE Computer*, 2001.

[7] L. Kagal, J. Undercoffer, F. Perich, A. Joshi, and T. Finin, "Vigil: Enforcing Security in Ubiquitous Environments," in *Grace Hopper Celebration of Women in Computing 2002*, 2002.
[8] J. Boyle and e. al, "The COPS Protocol." Internet Draft, Feb. 24, 1999.
[9] R. Mundy, D. Partain, and B. Stewart, "Introduction to SNMPv3." RFC 2570, April 1999.
[10] M. Stevens and e. al, "Policy Framework." IETF draft, September 1999.
[11] P. Loscocco and S. Smalley, "Integrating Flexible Support for Security Policies into the Linux Operating System," presented at Proceedings of the FREENIX Track of the 2001 USENIX, 2001.
[12] E. A. M. Luiijf, "Information Assurance and the Information Society," presented at EICAR Best Paper Proceedings, 1999.
[13] M. Román, C. K. Hess, R. Cerqueira, A. Ranganat, R. H. Campbell, and K. Nahrstedt, "Gaia: A Middleware Infrastructure to Enable Active Spaces," *IEEE Pervasive Computing (accepted)*, 2002.
[14] M. Roman and R. Campbell, "GAIA: Enabling Active Spaces," presented at 9th ACM SIGOPS European Workshop,, Kolding, Denmark, 2000.
[15] M. Roman, C. Hess, A. Ranganathan, P. Madhavarapu, B. Borthakur, P. Viswanathan, R. Cerqueira, R. Campbell, and M. D. Mickunas, "GaiaOS: An Infrastructure for Active Spaces," University of Illinois at Urbana-Champaign Technical Report UIUCDCS-R-2001-2224 UILU-ENG-2001-1731, 2001.
[16] L. Zadeh, "Fuzzy sets as basis for a theory of possibility," *Fuzzy Sets and Systems*, vol. 1, pp. 3-28, 1978.
[17] V. Samar and R. Schemers, "Unified Login with Pluggable Authentication Modules (PAM)," RFC 86.0, 1995.
[18] P. Kaijser, T. Parker, and D. Pinkas, "SESAME: The Solution to Security for Open Distributed Systems," *Computer Communications*, vol. 17, pp. 501-518, 1994.
[19] M. Roman, F. Kon, and R. H. Campbell, "Reflective Middleware: From Your Desk to Your Hand," *IEEE Distributed Systems Online Journal, Special Issue on Reflective Middleware*, 2001.
[20] J. Al-Muhtadi, R. Campbell, A. Kapadia, D. Mickunas, and S. Yi, "Routing Through the Mist: Privacy Preserving Communication in Ubiquitous Computing Environments," presented at International Conference of Distributed Computing Systems (ICDCS 2002), Vienna, Austria, 2002.
[21] J. Al-Muhtadi, R. Campbell, A. Kapadia, D. Mickunas, and S. Yi, "Routing through the Mist: Design and Implementation," UIUCDCS-R-2002-2267, March 2002.
[22] Z. Liu, P. Naldurg, S. Yi, R. H. Campbell, and M. D. Mickunas, "An Agent Based Architecture for Supporting Application Level Security," presented at DARPA Information Survivability Conference (DISCEX 2000), Hilton Head Island, South Carolina, 2000.
[23] P. Naldurg and R. Campbell, "Dynamic Access Control Policies in Seraphim," Department of Computer Science, University of Illinois at Urbana-Champaign UIUCDCS-R-2002-2260, 2002.
[24] P. Naldurg, R. Campbell, and M. D. Mickunas, "Developing Dynamic Security Policies," presented at Proceedings of the 2002 DARPA Active Networks Conference and Exposition (DANCE 2002), San Francisco, CA, USA, 2002.
[25] G. Sampemane, P. Naldurg, and R. Campbell, "Access Control for Active Spaces," presented at the Annual Computer Security Applications Conference (ACSAC), Las Vegas, NV, 2002.
[26] R. Sandhu, E. Coyne, H. Fienstein, and C. Youman, "Role Based Access Control Models," in *IEEE Computer*, vol. 29, 1996.