Wearable Security Services

Jalal Al-Muhtadi
Dept. of Computer Science
Univ. of Illinois at Urbana-Champaign
almuhtad@uiuc.edu

Dennis Mickunas
Dept. of Computer Science
Univ. of Illinois at Urbana-Champaign
mickunas@uiuc.edu

Roy Campbell
Dept. of Computer Science
Univ. of Illinois at Urbana-Champaign
rhc@uiuc.edu

Abstract

Active spaces and smart rooms are quickly gaining popularity in both the research arena, and in commercial projects. Active spaces are physical spaces populated with massive numbers of specialized embedded computing devices that increase human productivity by enabling users to interact seamlessly with the surrounding environment. Security for active spaces is necessary. In such settings, it is essential to have lightweight, mobile and convenient security services for users that enable them to prove their identities and interact securely with the smart devices that populate the surrounding environment. We propose embedding the security services in a basic wearable device that is already being worn and used by many people on daily basis, the wristwatch. We describe our implementation of these wearable security services, and show how the system is used in our active space test-bed.

1. Introduction

Active spaces are physical areas with specific geographical boundaries. They encompass different entities, properties, semantics, services as well as massive numbers of embedded computing devices. Active spaces increase human productivity by enabling users to interact seamlessly with the surrounding environment. All these components intercommunicate and cooperate with each other to create a "smart" and "active" physical space. An active space reconfigures automatically on the fly to match users' preferences, facilitate the exchange of information and provide convenient services. For example, upon entering an active living room, the lights, room temperature and background music are tailored to the user's preferences. Once the user looks toward the TV set, it should turn on. As soon as the telephone rings and the user answers, the TV is muted automatically. In an active classroom, once the class is over, the overhead projector is turned off automatically and lights are turned on. Upon the entry of the instructor for the next class, the lighting is adjusted, and the smart whiteboard is enabled. Any writings on the whiteboard are multicast automatically to the students' PDAs.

In our research in active spaces, we introduce Gaia [5] and [6], an operating system for active spaces that provides the infrastructure in the form of core services, over which smart spaces can be constructed and "dull" normal spaces can come to life and turn into active spaces. A detailed description of Gaia is beyond the scope of this short paper, however interested readers are referred to [5] and [6]. Various other research efforts have addressed possible infrastructures for managing the physical spaces and their contents [1], [2], [3], and [4]. However many either do not put adequate effort into securing their proposed environments or their security measures are customized for particular scenarios. Since active spaces with all their entities are perceived as computing systems with massively distributed components, any breach of security can easily cause serious disturbance and damage. Further, in any active space environment, there are numerous entities that interact with one another causing a huge flow of information. These entities may be mobile, roaming between different active spaces. An entity in an active space may refer to users, objects, computers, embedded devices, and sensors that populate an active space at a given instance of time. In such a dynamic and interactive setting, it is crucial to have security services provided at the core of the system. These services include the ability to authenticate users entering a space. Entities might also need to authenticate each other using some form of mutual authentication. In some situations it is desirable to enable secure exchange of information and enforce access control.

Adding security services to an active space may lead to additional complexity in the system design as well as inconveniences for users. For instance, in traditional systems, authentication of entities can be done through passwords, security badges, smart cards or biometrics. However, these techniques are either impractical or inconvenient in active space environments, where a user may be moving from one active space to another constantly. Password-based authentication is impractical every time a user enters a space. Security badges need to be worn at all times, their abilities are limited, and users tend to forget to put them on. Smart cards have limited functionality and they are still an additional piece of equipment to carry along and can be easily lost or stolen. Biometrics can only be used for authentication purposes. However, thanks to recent advances in mobility and miniaturization technologies, basic wearable devices with decent processing power and resources are now a reality that have already made it to the consumer market.

To enable security services without adding burden to the users, we propose embedding the security services in a basic wearable device that is already being worn and used by many people on daily basis; the wristwatch. Most people wear or carry watches to keep track of time and date. By providing lightweight security services in a "smart" wristwatch, a user need not worry about carrying or wearing additional equipment while roaming in an active space environment. We were able to build a prototype for wearable security services on a wristwatch using Matsucom's OnHand™ PC wristwatch [7], and deploy it in our active space test-bed. In this paper, we discuss our implementation of the security services and the applications that we built using this system.

The remainder of this paper is divided as follows. In Section 2, we give a description of the hardware used. In Section 3, we briefly mention the security services that are needed on this wearable device. Sections 4 to 7 give more details about our implementation. In Section 8, we give some performance measures. Next, we discuss potential future work in Section 9. Finally, we conclude in Section 10.

2. Hardware

We are using the Matsucom's OnHand™ PC wristwatch [7] (Figure 1). This is an enhanced version of "Ruputer™" wristwatch, which was introduced by Seiko Instruments Inc. in Japan in mid-1998. The OnHand™ PC features:

- A 16-bit microcontroller (MN10200 by Matsushita Electronics Corporation) running at 3.64 MHz, with three-stage pipeline architecture.
- 2 MB of flash memory,
- 128 KB RAM,
- 102x64 pixels LCD.
- To enable communication with the outside world, the device features infrared (IR) and serial ports with a transfer rate of 38400 bps.

An SDK that is based on GNU C is provided for the OnHand™ PC. We have used it to implement the wearable security services.

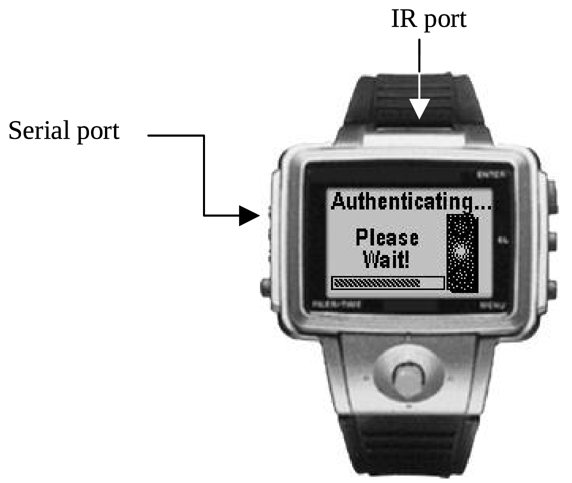

Figure 1: The OnHand™ PC wristwatch.

3. Security Requirements

In this paper, we are concerned about constructing lightweight, wearable, security services without adding additional burden to users. The security services that we need include the following.

3.1 Authentication

While some active spaces can be made "public", i.e. accessible by anyone, other active spaces may need to be restricted to authorized users only. For example, in a closed meeting or in an exam it is desirable to authenticate the users entering the restricted space.

3.2 Access Control

In some situations, it is necessary to enforce some access control, particularly in restricted spaces. Additionally, some resources within an active space can be restricted to specific users.

3.3 Location Service

It is very convenient to provide users with the ability to register their locations within an active space environment, making it possible for others to locate them if necessary. This is also useful for locating resources spread throughout the active environment.

3.4 Secure Communication

The security services should be able to establish secure communication channels between parties to prevent eavesdropping.

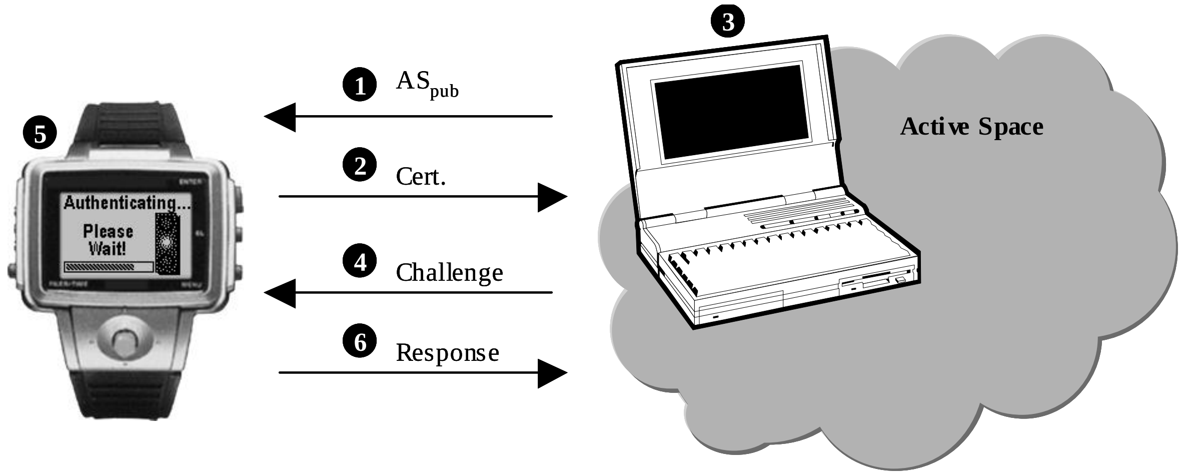

Figure 2: Authentication

4. Authentication

To provide a flexible and convenient authentication scheme for active spaces, we need a system that is capable of scaling to massively distributed systems, while supporting the dynamism and flexibility that active spaces promote. Our approach is to build lightweight, simple certificates. These certificates are issued and digitally signed by certificate authorities (CAs) that can be scattered throughout the active space environment. Initially, the certificate of the user is stored in his or her wristwatch. Although X.509 digital certificates [8] are becoming a popular standard, however, they are too complicated and sometimes too large to transfer over the limited bandwidth that most wearable devices have. Therefore, the lightweight certificates stored in the wristwatches only contain the fields shown in Table 1. These fields include the unique ID of the user, his or her public key, the issuing CA, and the digital signature of the issuing CA. The "extensions" field can be used for storing additional security attributes, or storing a "handle" for associating this certificate with additional information stored at some certificate store. The wristwatch also stores the private key of the user.

Table 1: Lightweight Certificate Format

User ID
Public Key
Issuing CA
Extensions
CA's digital signature

Current public key technologies, such as RSA, have too much overhead for porting into such a small wearable device. Instead, our current implementation uses symmetric encryption along with the Diffie-Hellman (DH) key exchange protocol [9]. DH is a well-known algorithm that allows two parties who have no prior knowledge about each other to agree on a shared secret.

The authentication protocol is illustrated in Figure 2. It involves the following steps:

1. Upon entering an active space that requires authentication, the user can use the IR port on the wristwatch (or mount the wristwatch on a cradle with a serial connection) to communicate with one of the authentication devices of the active space. The device will transmit the active space's public key (AS_{pub}).
2. The watch sends the user's certificate.
3. The active space validates the certificate by validating the CA's signature or consulting the issuing CA. If the certificate is not valid, the authentication fails. Otherwise, using DH key exchange protocol, the authentication device derives a session key (K) from the active space's private key and the user's public key.
4. The active space generates a random challenge, and sends it to the watch.
5. Upon receiving the random challenge, the watch derives the session key K from the user's private key and AS_{pub} that was obtained from step 1.
6. The response is computed using K and sent back to the active space. If the response is correct, then the authentication is successful.

If mutual authentication is required, then two additional steps are added to the above protocol:

7. The wristwatch generates a random challenge, and sends it to the device.
8. The device computes the reply using the derived key K and sends it back to the wristwatch.

These two additional steps prove to the wristwatch that the authentication device actually possesses the private key pair of AS $_{pub}$.

5. Access Control

There are different models for access control. Gaia employs a role-based access control model (RBAC) [10]. RBAC is a form of non-discretionary access control, which defines a hierarchy of roles. The roles are set up to mirror the organizational structure of people. Different permissions are granted to different roles and users are assigned to one or more roles depending on their title or job description. RBAC is convenient as it closely mimics the structure of organizations and allows access control policy administrators to define well-structured security policies.

In some settings, there might be a large number of users roaming in active spaces. Rather than keeping a long list of all the authorized users, it is much more efficient to group users into different roles and grant access to whoever possesses a particular role. Alternatively, the decision to perform an operation or access a resource must be decided by the access policy manager.

As an example, in a closed meeting for all project leaders, rather than compiling a list of all project leaders, the role name "project leader" can be added to the certificates stored in the project leaders' watches. In this case, the watch merely stores the roles that the owner may possess. The role hierarchy, access permissions and policies are stored in an RBAC component, which may be a central-

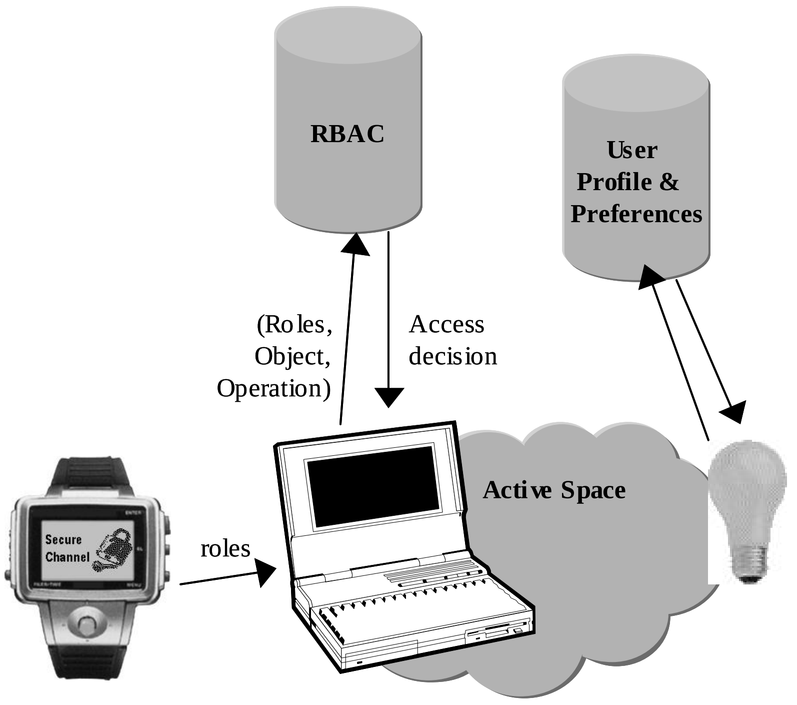

Figure 3: Access control & user preferences

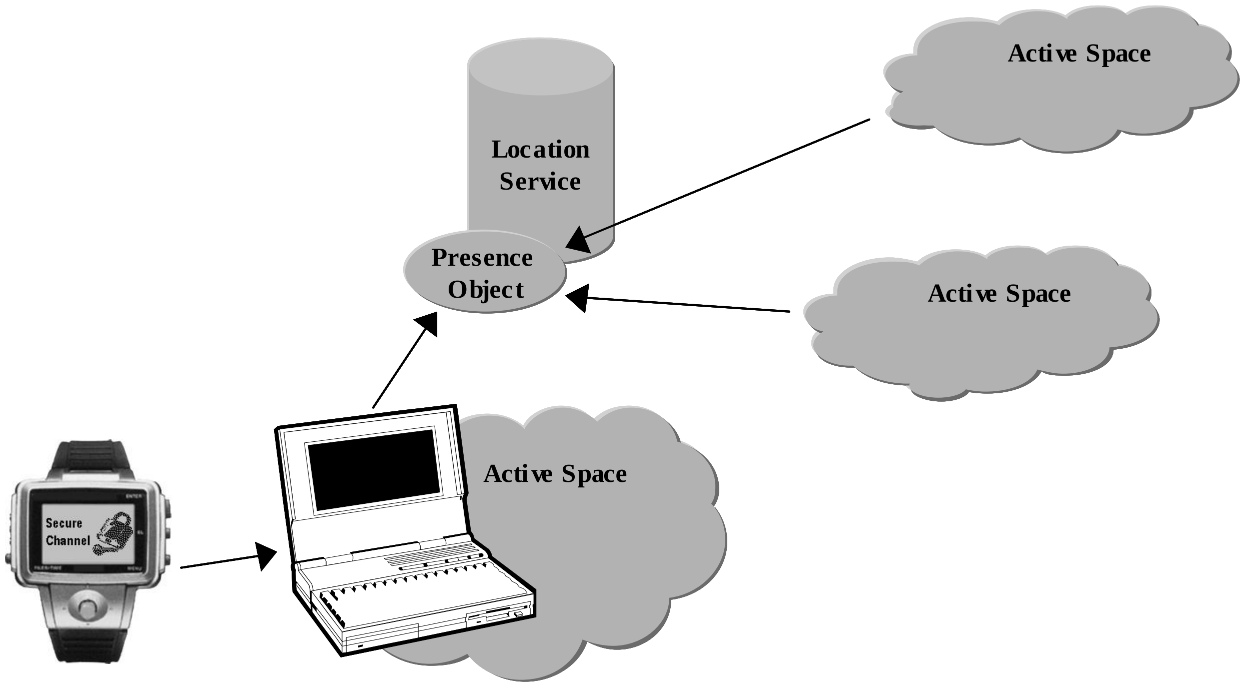

Figure 4: Location Service.

ized database that active spaces can query. In such scenarios, the active space authenticates the user as described in Section 4. The authorized roles of the user are extracted from his certificate. At this stage the active space queries the RBAC component to find out whether it is permissible for that user to perform a specific operation on a certain object or resource (Figure 3). Similarly, the user's certificate may contain certain attributes or a pointer to a list of preferences. Based on these preferences, an active space can reconfigure itself, for example, if a person enters his office the level of illumination is adjusted according to his preferences.

6. Location Service

The location service enables users to locate other users and resources within the active space environment. This is also useful for identifying occupants of an active space. For instance, knowing who is attending a particular meeting, or querying about the location of Alice, or finding the nearest color printer, and so on. The process of registering a user in an active space can occur after authentication, or it can be a separate process where the user communicates with one of the user-registration devices in a particular active space. Once a user registers with the designated device, it invokes the presence object (Figure 4) that handles the addition and removal of users to and from the location service. A location service can be shared among several active spaces; alternatively, each space may have its own location service. To inquire about a specific user or resource, the watch (or other devices which the inquirer may possess) can contact the user-registration device, which in turn returns the information from nearby location services.

7. Secure Communications

Wristwatches and other portable devices carried by the user should be able to communicate securely with other entities within an active space. Every smart device in an active space should have a public/private key pair. When a user decides to communicate with some device and employs the DH key exchange protocol, a unique session key (K) can be established between the wristwatch and the device. This key can be used to encrypt the data transmission. However, we would like to implement a protocol that offers *forward* and *backward secrecy*. Forward secrecy means that the compromise of the current key does not compromise future communication. Backward secrecy is similar, but relates to the compromise of past communication. We achieve this by introducing the concept of establishing secure channels for communication. A secure channel in this context refers to a virtual grouping of data packets that use a certain session key. Every secure channel uses a different session key. The protocol is illustrated in Figure 5. When any two devices require secure communication, they use DH to establish K. Next the initiator sends a special "establish channel" message and appends a random number (Rand) to it. Both devices employ a one-way cryptographic hash function for deriving the channel's session key (CK) from K and Rand: *h*(K, Rand) = CK. This way, K is not used as often, thus significantly lowering the chance of it being compromised. Upon completion of communication, the initiator can send a special message to teardown the channel.

8. Performance

Our implementation employs RC5 [11] for symmetric encryption as RC5 relies on primitive computational operations commonly found on microprocessors. We chose a key size of 128 bits and 12 rounds. For the one-way

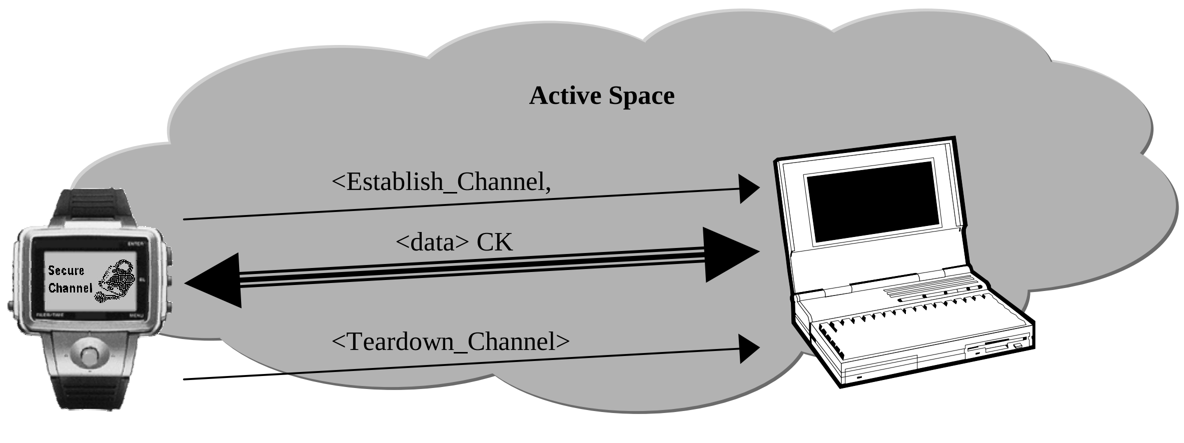

Figure 5: Secure Channels

hash function we use MD5 [12]. Table 2 shows the average performance numbers for the cryptographic algorithms that we used, running on the wristwatch's 16-bit 3.64-MHz processor. Our implementation of the DH key exchange protocol is straightforward and has room for optimization. Table 2 indicates that generating a 128-bit key through DH key exchange protocol consumes too much time. Hence, in our system we are generating a 64-bit key using DH then applying some transformations to derive a 128-bit key.

Table 2: Performance numbers

Operation	Average Time
RC5 encryption for 1024 bytes of data, using a 128-bit key, 12 rounds	2.62 seconds
RC5 decryption for 1024 bytes of data, using a 128-bit key, 12 rounds	2.58 seconds
RC5 encryption for 8192 bytes of data, using a 128-bit key, 12 rounds	18.61 seconds
RC5 decryption for 8192 bytes of data, using a 128-bit key, 12 rounds	17.58 seconds
MD5 hash for 1024 bytes of data	1.02 seconds
MD5 hash for 8192 bytes of data	7.26 seconds
DH generating a 64-bit key	38.74 seconds
DH generating a 128-bit key	200.06 seconds

9. Future Work

Elliptical curve cryptography is promising, particularly its ability to provide security equal to RSA for a far smaller bit size, thereby reducing the processing overhead. We plan to experiment with elliptical curve cryptography on the wristwatch and measure its performance. Moreover, we are considering offering a way for the wearable security services to interoperate with the Kerberos authentication system [13], which is widely used to provide authentication and security in today's distributed systems.

10. Conclusion

Securing active environments that are populated with various entities and massively distributed devices is essential. Like wearing armor in a battlefield to reduce harm, it makes sense to have security services embedded in a device that is naturally worn by people most of the time. In this paper, we discussed our implementation of wearable security services that provide authentication, access control and secure communication in active spaces.

Due to limited bandwidth and processing power, wearable devices provide less security and quality of protection in comparison with stationary devices. Steady advances in miniaturization techniques and transistor technology along with increasing memory and processing speeds promise higher quality of wearable security. Nevertheless, wearable devices can provide a good balance between mobility and security.

11. References

[1] CoolTown, HP laboratories CoolTown Appliance Computing, http://cooltown.hp.com

[2] Abowd, G.D., *Classroom2000: An experiment with the instrumentation of a living educational environment,* IBM Systems Journal. 38(4 - Pervasive Computing).

[3] Shafer, S., et al., *The new EasyLiving Project at Microsoft Research,* Microsoft Research, 1999.

[4] Gribble, S., et al., *The Ninja Architecture for Robust Internet-Scale Systems and Services*, to appear in a Special Issue of Computer Networks on Pervasive Computing.

[5] Roman, M. and R. Campbell, *GAIA: Enabling Active Spaces,* 9th ACM SIGOPS European Workshop. September 17th-20th, 2000, Kolding, Denmark.

[6] The Gaia Homepage, http://choices.cs.uiuc.edu/ActiveSpaces/index.html

[7] On Hand PC Homepage, http://www.onhandpc.com/

[8] ITU-T Recommendation X.509 (revised), *The Directory – Authentication Framework,* 1993, International Telecommunication Union, Geneva, Switzerland.

[9] Diffie, W., and M. Hellman, *New directions in cryptography,* IEEE transactions on Information Theory, 22(6):644-654. November 1976.

[10] Sandhu, R., E. Coyne, H. Fienstein, and C. Youman, *Role Based Access Control Models,* IEEE Computer, 29(2), February 1996.

[11] Baldwin, R. and R. Rivest, *The RC5, RC5-CBC, RC5-CBC-Pad, and RC5-CTS Algorithms*, RFC 2040, October 1996.

[12] Rivest, R., *The MD5 Message-Digest Algorithm*, RFC 1321, April 1992.

[13] Neumann, B. and T. Ts'o, "*Kerberos: An Authentication Service for Computer Networks*", IEEE Communications Magazine, 32(9): 33-38, September 1994.