

SHAD: A Human Centered Security Architecture for Partitionable, Dynamic and Heterogeneous Distributed Systems *

Enrique Soriano Salvador[†]
Laboratorio de Sistemas, ESCET, Universidad Rey Juan Carlos
C. Tulipán S/N, 28933
Móstoles (Madrid), Spain
esoriano@lsub.org

ABSTRACT

In an ubiquitous computing environment, principals can use at any time different devices (displays, input devices, workstations, laptops, mobile phones, PDAs etc.) interconnected by different kinds of networks (Ethernet, Wi-Fi, IrDA, Bluetooth etc.). Some time later, these devices can be off line or disconnected. Principals, as well as devices, come and go. In this scenario, a lightweight security scheme is necessary to add authentication, secrecy and integrity without depending on connections to centralized services. This scheme must support disconnection and delegation. Also, it has to be easy to use and to deploy. Neither classic security schemes that are used in common open network systems nor security schemes proposed for ubiquitous environments comply with some of these requirements. In this paper we propose SHAD, a human centered security scheme designed for a new operating system named Plan B. SHAD avoids the use of centralized entities and it is designed to be agile in a Peer-to-Peer environment.

Categories and Subject Descriptors

D.4.6 [Software]: Operating Systems—*Security and Protection*; C.2.0 [Computer Communication Networks]: Security and Protection

Keywords

SHAD, Ubiquitous, Pervasive, Security, Authentication, P2P.

*This work has been funded in part by MCYT TIC2001-1586-C03-01 and URJC PPR-2003-40.

[†]Author supported by the MCYT FPI grant with id. BES-2003-2942.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

1st International Middleware Doctoral Symposium Toronto, Canada
Copyright 2004 ACM 1-58113-948-9 ...\$5.00.

1. PURPOSE DESCRIPTION

Ubiquitous computing is becoming real. It implies environments close to dynamic, partitionable and heterogeneous distributed systems where security is a big problem to resolve. Security is critic for ubiquitous environments: users must be authenticated, confidentiality and integrity have to be assured (mainly in wireless communications) and the access to the resources must be controlled. These environments are closer to a Peer-to-Peer system than to a classic client/server open network system:

- The nodes can be mobile devices (mobile phones, handhelds, etc.) or non mobile devices (workstations, servers, panels, etc.), and they can be interconnected by different kinds of network technologies (ethernet, wi-fi, bluetooth, etc.).
- These devices belong to users, and users would eventually need or want to use devices that they do not own.

The system is *partitionable*, because the security capabilities have to be available at locations where two principals can communicate although there may be no connection with the rest of the system. The system is *dynamic* too, because devices, as well as humans, come and go. The system is *heterogeneous* owing to the range of different devices that can exist in an ubiquitous environment.

In a scenario like this one, a new security architecture is needed.

Traditional security schemes, like Kerberos [24] and Sesame [13, 3], are based on centralized entities like key distribution centers and authentication authorities. Other security schemes that have been proposed for ubiquitous environments [4, 19] are heavily based on them. In most ubiquitous schemes [6, 19, 11, 22, 23], it is necessary to reach centralized servers to query context and location information too.

For these systems to work, principals need to be online and they must have connectivity with the centralized server if they want to communicate. What would then happen when two users meet at a remote isolated place? Another problem is that most of these systems are hard to administrate because they need accounts for users that require management. Moreover, the centralized services are unique failure points for the ubiquitous environment.

Many approaches are based on middleware. These systems have problems with existing or native applications,

because in most cases these applications have to be modified or wrapped. An example is Cerberus [18], the security architecture for Gaia [21], that would not work for old applications not using Gaia.

Approaches based on libraries and frameworks that allow programmers to reach context information [9, 23] have the same problem: ancient software does not work and it has to be re-implemented.

Moreover, most of these architectures fail to work in Peer-to-Peer settings.

Therefore, a new architecture is needed for ubiquitous computing environments, because the current ones introduce complexity, obtrusiveness and centrality.

2. GOAL STATEMENT

The purpose of our work is to design and implement a new general non-obtrusive, lightweight, Peer-to-Peer and human centered security architecture for the environments described above. From now on, we will refer to it as SHAD: Secure Human-centered Architecture for Distributed-systems.

Our vision of an ubiquitous environment is realistic and is reachable on present days. A human owns a set of devices (laptops, PDAs, workstations, servers, phones, printers, panels, projectors, etc.), maybe tens but not hundreds. The main goal is to allow the human to share his resources with trusted people at locations he often frequents (i.e. work place, residence blocks, etc.).

Our idea is to center security on humans. Ubiquitous environments are physical environments where humans interact sharing their own resources and using others resources.

In our Peer-to-Peer approach, the human is the Peer.

SHAD will meet the following requirements:

1. **Independence of centralized services or authentication servers.**
2. **Ease of use and the non-obtrusiveness.**
3. **Supporting of disconnections and delegation.**
4. **Minimizing of power consumption and the processing limitations of mobile devices.**
5. **Ease of deployment.**

In next subsections we describe how we will face these requirements in SHAD.

2.1 The Personal Command Module

SHAD is being designed to be agile in Peer-to-Peer environments. It is centered on the human and is based on a personal device called PCM (*Personal Command Module*) [15].

The PCM is a mobile device (currently we are using a Symbian [17] enabled mobile phone) that represents the user in the system. The PCM allows the user to control his activities and to use the pervasive elements nearby. In addition, it is the perfect candidate to be a Peer-to-Peer (or Human-to-Human) authentication server.

The PCM stores and manages the user's secrets and it is the entity in charge of granting or refusing service requests. Using the PCM to authenticate users, we can comply with the necessity of independence of centralized services: if two

principals can communicate, although there may be no way to connect to the rest of the system, they can mutually authenticate and exchange tickets.

The use of personal mobile devices to authenticate users has been proposed in several related works [19, 6, 7, 9, 8]. The main problem of this approach is that these devices could be stolen or lost because they are physical objects. We think that people can bear these risks, because in real life all of us depend on physical objects like credit cards, personal documents and keys. Consequently, we are accepting this trade-off every day.

2.2 Human Centered Approach

The human centered approach satisfies the non obtrusiveness and the supporting of delegation and disconnection. In addition, it is the natural way to represent the concept of trust in an ubiquitous environment. Some related works [11, 22] use estimations made by the machine to fix the trust level. Alternatively, Langheinrich proposes in [14] that trust, as a subjective concept, must be fixed by the human and not by the machine. Our approach complies with this proposal while keeping the simplicity:

- A device is always owned by a human. For example, the printer located in my office belongs to me.
- Humans place trust in humans. For example, I trust Paco, therefore Paco is allowed to use my printer.

The user does not have to be introducing passwords or to be using authentication devices (i.e. fingerprint reading devices) all the time, because the PCMs are able to negotiate keys without human supervision. The PCM stores the secrets used by its owner to use resources that don't belong to him. A principal can use a resource (that doesn't belong to him) when the owner is on line. Later, the principal will be able to keep using the resource although the resource owner disconnects, because the PCM may handle disconnections by transferring session keys to devices.

The resources belonging to a set of humans in real life will be owned by virtual users in the ubiquitous environment. For example, a touch panel located in a common room of the LS department is owned by a virtual user called "LS". The PCM belonging to "LS" is a common workstation located on the servers room.

Consider the scenario depicted in Figure 1. Katia and Gorka are users that work in the ubiquitous environment and meet often. One day, Gorka and Katia decided that they trusted each other and then they paired their PCMs using an IRdA communication to avoid sniffer attacks¹. Note that the pairing process is only made once, the day that the users decide to trust each other. Another day, Katia needs to use Gorka's laptop to play an mp3 audio file stored in Katia's PDA:

1. Katia's PDA makes contact with Katia's PCM, and then Katia's PCM acts like an authentication server for Katia's PDA.
2. Katia's PCM negotiates a ticket with Gorka's PCM to use the audio capabilities of Gorka's laptop.

¹It is not the only way to pair PCMs, we can use a PIN based manual pairing (as in Bluetooth [12]), e-mail ciphered with PGP[26], SSH[25], SSL[10] based chat, etc.

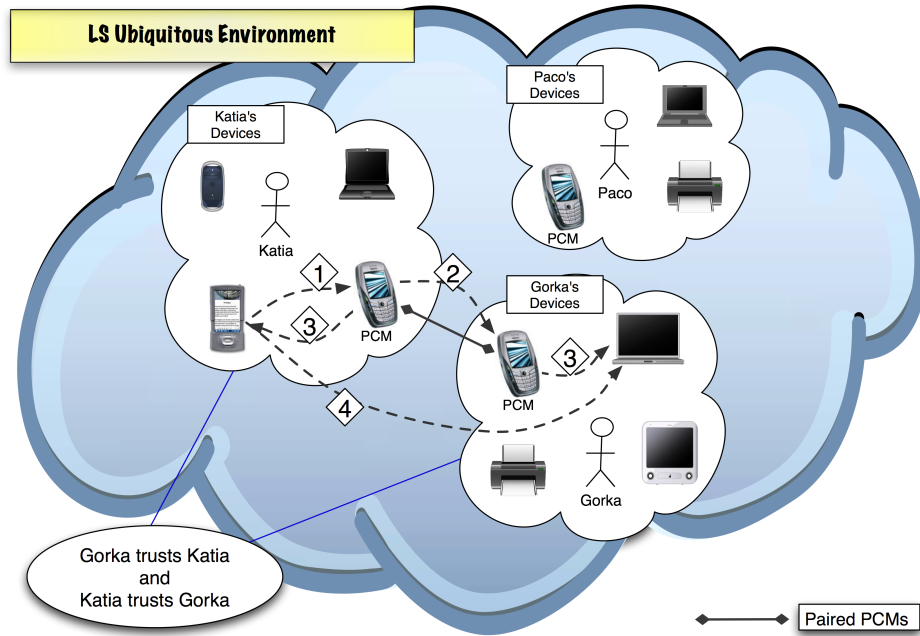


Figure 1: An example scenario of our human centered approach.

3. Gorka's PCM sends the ticket to Gorka's laptop. Concurrently, Katia's PCM sends the ticket to Katia's PDA.
4. Katia's PDA makes contact with Gorka's laptop and the access is granted by the ticket negotiated before.

Two days later, Katia needs to play another mp3 file in Gorka's laptop, but Gorka and his PCM are away. There is no problem, because Katia's PDA has the ticket used two days ago and it is still valid.

Now suppose another scenario. Nemo and Eva meet at an isolated location (i.e. a subterranean parking) where there is no way to make contact with the rest of the system or the Internet. Nemo's Pocket PC and Eva's laptop are equipped with both Bluetooth and Wi-Fi technologies. Nemo needs to use the 17" display of Eva's laptop to show her a high resolution video stored in his Pocket PC. A security architecture that depends on centralized entities could not face this situation. In contrast, SHAD fits well into this scenario: Nemo's PCM and Eva's PCM can negotiate tickets to allow Nemo's Pocket PC to make contact with Eva's laptop. We represent this scenario in Figure 2.

An access control policy must be adopted for SHAD. We have chosen a role based access control (RBAC) policy: the human assigns roles to other humans in his own PCM. In our scheme, trust is reciprocal, but there may be different access privileges in each side depending on the level of trust. For example, in the scenario depicted in Figure 1, Gorka had given to Katia an *output devices user* role in his PCM, so Katia can use the audio output of Gorka's laptop but she cannot use input devices (i.e. the Gorka's laptop keyboard). On the other hand, Katia had given to Gorka an *all devices user* role in her PCM, because Katia-to-Gorka trust level was higher than Gorka-to-Katia trust level.

2.3 Integration with the Operating System

The ease of deployment is addressed by operating at the file system level², using the Plan B operating system [5] approach. Therefore, old applications will not need to be modified or to be aware of the security. Applications that run in other operating systems will only need to access remote files to be part of the ubiquitous environment.

In Plan B, all resources are represented by files. Operations over files (both local and remote) are performed by a small set of self-contained RPCs following a protocol called *Bp*. Consequently, all the machinery related with security can be integrated with the operating system. To use remote files, the kernel has to be authenticated in the remote device and it has to share the keys to crypt and decrypt data.

As a result, ancient applications do not need to be modified, because they use the same system calls to manipulate files.

On the other hand, we are working with symmetric cryptography, because the use of asymmetric cryptography can be inappropriate for mobile devices: this kind of algorithms are more expensive in processing time and therefore in power consumption. In addition, SHAD is being designed to be cheap in messages to reduce the power consumption too, so it complies with the minimizing of power consumption necessity.

3. METHODOLOGY

First we studied the schemes proposed for both traditional distributed systems and ubiquitous computing. When we were sure that no one of these schemes could satisfy our

²Please note that this approach is quite different to SFS [16]. SFS uses a PKI infrastructure. In addition, it is resource centered because public keys are embedded in file names.

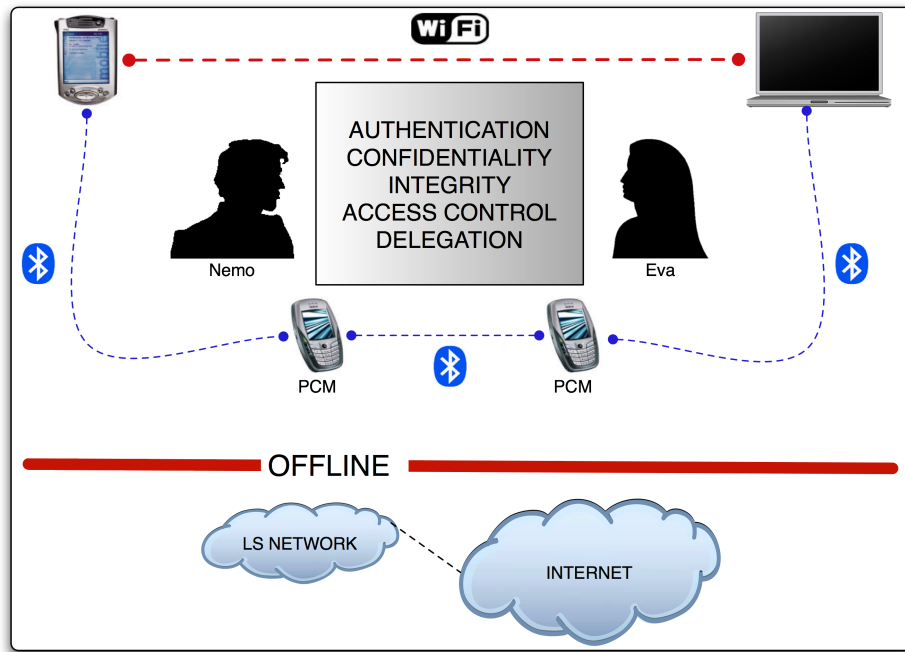


Figure 2: An example of communication in an isolated location.

needs, we started by designing SHAD. Then, we studied how to integrate the architecture with our distributed operating system, Plan B. Currently, we have this work done.

Now, we are working on a prototype for the ideas of SHAD using the Plan 9 [20] operating system. We will integrate the ideas with the context infrastructures that we are using daily in our department: a X10 service for controlling the electrical installations and for locating things and persons, and a vision system for controlling the snail mail of the department staff.

Next, we will rework the Plan 9 security scheme to use one authentication server for each user in the system, just like the PCM would do in the final architecture. Later, we will change the boot process to make the Plan 9 login process less obtrusive: when a user has her principal terminal booted (that will act like her own authentication server), she will not introduce the password again (although she boots more terminals). Finally, we will port it to Plan B and we will implement a complete PCM. Before, we will be using it in our own ubiquitous environment [1] (a demo video can be downloaded from [2]) and we will test the architecture to check that it works indeed.

4. EVALUATION

The goal can be easily evaluated using the ubiquitous environment and proving that the architecture does its job and meets all the usability requirements (non obtrusiveness, easy configuration, etc.).

First, we will try to attack the system using well known techniques (reply attacks, lost messages, man in the middle, etc.) to check that the security architecture works and is reasonably secure for ubiquitous environments.

Second, we will be using the ubiquitous environment daily in our department. Also, we will build a model for de secu-

urity system and we will try to prove it correct.

5. REFERENCES

- [1] Laboratorio de sistemas. <http://lsub.org>.
- [2] Ls: papers and documentation. <http://lsub.org/#docs>.
- [3] Sesame: A secure european system for applications in a multi-vendor environment. <https://www.cosic.esat.kuleuven.ac.be/sesame/>.
- [4] J. Al-Muhtadi, M. Anand, D. Mickunas, and R. Campbell. Secure smart homes using jini and sesame. In *Proceedings of the 16th ACSA/ACM Annual Computer Security Applications Conference, 2000*, 2000.
- [5] F. J. Ballesteros, G. Guardiola, K. Leal, E. Soriano, P. Heras, E. M. Castro, and S. Arévalo. Plan B: Boxes for networked resources. *Submitted for publication, also in <http://lsub.org/ls/export/box.html>*, March 2004.
- [6] Jakob E. Bardram, Rasmus E. Kjær, and Michael Ø. Pedersen. Context-aware user authentication – supporting proximity-based login in pervasive computing. *UbiComp 2003*, 2003.
- [7] Allan Beaufour and Philippe Bonnet. Personal servers as digital keys. *Second IEEE International Conference on Pervasive Computing and Communications*, 2004.
- [8] L. Bussard and Y. Roudier. Authentication in ubiquitous computing. *Workshop on Security in ubiquitous computing, in Proceedings of the UbiComp 2002*, 2002.
- [9] M. Corner and B. Noble. Protecting applications with transient authentication. In *First ACM/USENIX International Conference on Mobile Systems, Applications and Services (MobiSys'03)*, San

- Francisco, CA., 2003.
- [10] Netscape Communications Corp. *The SSL (Secure Sockets Layer) 3.0 Protocol*. Mountain View, CA, 1995.
- [11] C. English, P. Nixon, and S. terzis. Dynamic trust models for ubiquitous computing environment. *Workshop on Security in ubiquitous computing, in Proceedings of the Ubicomp 2002.*, 2002.
- [12] J. Haartsen, M. Naghshineh, J. Inouye, O. Joeressen, and W. Allen. Bluetooth: Vision, goals, and architecture, 1998.
- [13] P. Kaijser, T. Parker, and D. Pinkas. Sesame: The solution to security for open distributed systems. *Computer Communications, vol. 17, pp. 501-518*, 1994.
- [14] M. Langheinrich. When trust does not compute - the role of trust in ubiquitous computing. *Workshop on Privacy at the 5th International Conference on Ubiquitous Computing, Ubicomp 2003.*, 2003.
- [15] K. Leal, F.J. Ballesteros, G.Guardiola, and E. Soriano. Plan B's personal command module. commanding user activities in ubiquitous environments. *Submitted for publication, also in <http://lsub.org/lsub/>*, 2004.
- [16] David Mazieres, Michael Kaminsky, M. Frans Kaashoek, and Emmett Witchel. Separating key management from file system security. *Symposium on Operating Systems Principles*, 1999.
- [17] D. Mery. *Symbian OSversion 7.0 functional description. White paper*. Symbian Ltd., 2003. <http://www.symbian.com/technology/whitepapers.html>.
- [18] Al Muhtadi, Anand Ranganathan, Roy Campbell, and N. Dennis Mickuna. Cerberus: A context-aware security scheme for smart spaces. *IEEE International Conference on Pervasive Computing and Communications*, 2003.
- [19] Al Muhtadi, Anand Ranganathan, Roy Campbell, and N. Dennis Mickunas. A flexible, privacy-preserving authentication framework for ubiquitous computing environments. *In Proceedings of IWSAEC 2002*, 2002.
- [20] R. Pike, D. Presotto, K. Thompson, and H. Trickey. Plan 9 from bell labs. In *Proceedings of the Summer 1990 UKUUG Conference*, pages 1-9, London, July 1990.
- [21] M. Roman, C. K. Hess, R. Cerqueira, A. Ranganat, R. H. Campbell, and K. Nahrstedt. Gaia: A middleware infrastructure to enable active spaces. *IEEE Pervasive Computing, vol. 1, pp. 74-82*, 2002, 2002.
- [22] N. Shankar and W. Arbaugh. On trust for ubiquitous computing. *Workshop on Security in ubiquitous computing, in Proceedings of the Ubicomp 2002*, 2002.
- [23] N. Shankar and D. Balfanz. Enabling secure ad-hoc communication using contextaware security services. *Workshop on Security in Ubiquitous Computing, in Proceedings of the Ubicomp 2002*, 2002.
- [24] J. G. Steiner, B. C. Neuman, and J. I. Schiller. Kerberos: An authentication service for open network systems. *In Proceedings of the Winter 1988*, 1988.
- [25] T. Ylonen. SSH - secure login connections over the internet. *Proceedings of the 6th Security Symposium* (USENIX Association: Berkeley, CA):37, 1996.
- [26] Phil Zimmermann. *Pretty Good Privacy (PGP), PGP User's Guide*. Massachusetts Institute of Technology, 1994.