New Security Services based on PKI

Antonio F. Gómez[†], Gregorio Martínez[†] and Óscar Cánovas[‡],
[†]Department of Information and Communications Engineering
[‡]Department of Computer Engineering
University of Murcia
30.071 Murcia (Spain)
skarmeta@dif.um.es, gregorio@dif.um.es, ocanovas@ditec.um.es

Abstract

The basic job of a Public Key Infrastructure, PKI for short, is to define the mechanisms used both to allow a recipient of a signed message to trust a digital signature and to allow a sender to find the encryption key for a recipient. It is comprised of those elements needed to manage and enable the effective use of public key cryptography technology, particularly on a medium and large scale. Nowadays, PKIs are being considered as a key element for providing security to new distributed communication networks and services. In this paper, we describe two of these innovative security services built over our own designed and implemented Java IPv6 PKI: a distributed security policy management architecture and a distributed credential management system.

Keywords: Public Key Infrastructure, Security Services, SPKI Credentials, Security Policy, Distributed Management

1 Introduction

Public key cryptography is widely recognized as being a fundamental technology on which several basic security services can be built, such as authentication, integrity, non-repudiation, and confidentiality. Crucial to the operation of a global public key cryptosystem on the Internet is a practical and reliable method for publishing the public keys, called a Public Key Infrastructure or PKI.

Public key cryptography is suitable for distributed and dynamic environments, with a medium or big number of communicating parties sending data through insecure channels. In fact, it provides a secure communication method for recipients not previously known each other. In order to get this kind of secure communications, a common element of trust is necessary. Public key values used in these communications must be certified (signed) by trusted entities (certification authorities), in order to assure the identity of the parties involved.

On this basis we have used our own designed and implemented Java IPv6 PKI [20], to create two new secure distributed services related to the management of next generation networks. In fact, the management of such large and dynamic systems presents serious challenges to traditional centralised approaches and thus, the use of certified distributed management entities and digitally signed management statements is gaining real acceptance as the most viable alternative.

The first of these security services is called DSPM, or Distributed Security Policy Management service. Generally speaking, policy-based systems [17] are used in a wide spectrum of applications ranging from enterprise modelling to quality of service or secure management within networks. In all these environments, the concept of policy refers to a set of rules governing choices in the behaviour of one specific system. The final motivation is to be

able to modify a policy in order to change the behaviour of one system without having to re-implement the whole system.

With this set of concepts, a new network paradigm is being defined. This new paradigm is a shift away from hardware-based, inflexible network that need to be upgraded manually, to a flexible, programmable network in which configuration changes can be automatically and securely –using a PKI– propagated throughout the network devices belonging to a set of network domains under the control of one, or various, certified network administrators.

The second security service that we have built over our own PKI, is called DCMS, or Distributed Credential Management System [2]. Nowadays, the words *certificate* and *identity certificate* are still used as synonyms. However, a certificate is a record stating some information about the entity the certificate was issued to, and this information may be a role membership statement, or an authorization. Authorization certificates bind a capability to a key, and this capability can be used to determine what the entities are allowed to do.

One of the most outstanding proposals related to this type of certificates has been the SPKI/SDSI infrastructure [9]. SPKI/SDSI provides three types of digital certificates (ID, attribute, and authorization) that can be used in several security scenarios. In fact, there are several proposals making use of SPKI certificates in order to provide authorization services to many different application environments. Most of these scenarios are based on delegation, where resource guards have an ACL with few entries granting keys belonging to some authorization or naming authorities, the right to delegate all access to the controlled resources. However, some of these proposals do not explain how certificates are issued by the authorities, and this is usually application-dependent. Although simple and not distributed approaches can constitute a good alternative for small scenarios, some scalability and interoperability problems might arise in more complex and distributed environments.

The remaining of this paper is organized as follows. In section 2, we outline the main features of the Java IPv6 PKI we have designed and implemented. Section 3 discusses the first advanced security service developed over this PKI, the Distributed Security Policy Management system. Then, section 4 describes the other security service we have designed and implemented, the Distributed Credential Management System. Finally, section 5 concludes the paper.

2 Our Public Key Infrastructure

Generally speaking, a PKI is a set of hardware, software, people, and procedures needed to create, manage, store, distribute and revoke public key certificates. With this, a PKI is able to provide trusted and efficient private key and public key certificate management, thus enabling the use of authentication, non-repudiation, and confidentiality basic security services. To get this objective, the basic components of one public key infrastructure are normally a certification authority, one -or several- registration authorities, and a directory server. Some other extra components, like smart cards, time stamp servers, OCSP servers, and so on, can be implemented depending on the level of services offered by a particular PKI.

The two advanced security services that we are presenting in this paper are based on the PKI designed and implemented by our research group [20]. Our PKI main objective is to establish a high security infrastructure for distributed systems. It is based on a previous design [4] but adding some new features that we highlight now:

- It allows certificates to be requested, renewed and revoked for every entity (final user

or process) belonging to one organisation

- Final users can carry out the certification operations from their own web browsers or through the Registration Authorities
- It allows the use of an LDAP-based directory to store the certificates, CRLs, and PKI policies
- Users can use file system, Java Cards and/or RSA smart cards to store their own cryptographic information –private key(s), certificate(s) and CA's certificate(s)–. This allows user mobility and increases the security of the whole system
- It supports the definition of a Certification Policy as we explain in section 3.1
- It is completely developed in Java, allowing the use of any operating system to run an implementation of this PKI
- It is based on those drafts and standards specified by the IETF inside its PKIX [16] working group
- It has support for the SCEP [14] protocol (Simple Certificate Enrolment Protocol), enabling router certification requests
- It has support for the OCSP [15] protocol
- The Time-Stamp Protocol [1] defined by the IETF is implemented in the system
- The whole PKI is IPv6 enabled, so any operation can be performed using this new network protocol

The basic architecture and set of communications of our Java IPv6 PKI are depicted in figure 1.

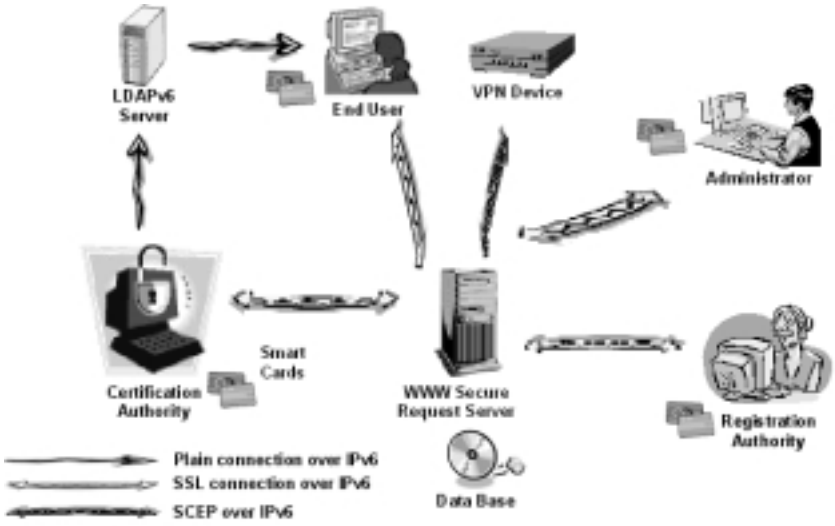

Figure 1: Java IPv6 PKI Architecture and Communications

3 Security Policies

Within the internet community there is a considerable interest in policy-based networking and communications, normally based either on the proposals coming from quite active research groups [17] or on the standards defined by the Internet Engineering Task Force (IETF) [11] and the Distributed Management Task Force (DMTF) [19]. But at the end, it seems that the number of complete systems and implementations able to support the specification and deployment of these policies is quite small, which is one of their main drawbacks.

When considering the deployment of these policy frameworks to manage secure distributed systems, as it is our case, there are several aspects we have to be aware of. The most important one is, of course, security. In fact, and as stated during the introduction, these policies are normally used to manage distributed communication systems, and thus they need to be based on secure statements that need to be trusted along different administrative domains. Public Key Infrastructures are the best option to provide the right level of security requested by this kind of services.

Our interest is mainly based on policies for managing public key infrastructures and policies for managing secure virtual private networks. In the first case, policies generation is centralized and utilization is distributed. In the second one, both processes are distributed.

Next sections describe how both security services have been designed and implemented, based on the secure and trustworthy statements provided by our PKI.

3.1 Security Policies for PKIs

The first security service we have created using the security information provided by a PKI is the definition and management of those policies driving the way the PKI itself works. This is mainly characterized for a centralized creation process driven by the PKI administrator (or administrators), but it is utilized in a distributed way by different components in the system.

We consider a PKI policy as the digital implementation of some issues contained in a particular certification practice statement (CPS). It is a digitally-signed document specifying how some issues related to basic services of a PKI, such as certification, publication, renewal, or revocation, must be addressed. It is widely distributed amongst different components of the PKI, especially the registration authorities.

This document can be only generated by authorized users since it has a great impact on how some architectural elements perform their operations. This set of users (administrators) is commonly defined in an application-dependent way, although usually using a centralized approach (the list of valid administrators is stored in a central database).

A PKI policy is formed by a serial number, date of issuance, date of next issuance, and the set of policy elements. As we will see, this set specifies which rules must be applied to those certificates that are going to be defined or are already issued. The policy is digitally-signed in order to protect the integrity and to authenticate the originator. The signature can be generated using the administrator's private key or can be directly created by the certification authority. Administrator's signatures are tricky since verifiers must obtain the administrator's certificate using a trusted channel, and it can become difficult when there is a high number of verifiers or administrators. It is worth emphasizing that the X.509 certificates related to the administrators are only statements about identities (they do not contain information about authorization for issuing PKI policies), and therefore a verifier can only accept a security policy issued by an administrator if the related certificate was obtained through a trusted

channel, otherwise a malicious user might easily sign and distribute an alternative policy. On the other hand, CA-generated PKI policies are a better alternative for large, complex, and dynamic environments. The certification authority's certificate is the trusted root for every element in the system, and therefore it is widely distributed. A PKI policy signed by the CA can be easily verified by any entity.

Thus, we propose a system where different administrators can establish different policies, but those must be signed by a certification authority. Once the policy is signed, it is published using the X.500 entry related to the certification authority in order to be available. Although this scheme can be considered too rigid, the fact that these policies are issued by the same central authority is justified by the inherently centralized structure of a public key infrastructure, and the complexity of assigning permission to keys using the X.509 approach.

3.1.1 Rules expressed by a PKI policy

A PKI policy contains a set of rules or conditions which must be enforced or applied by an element of the PKI. The rules include a specification about which is the set of users controlled by them (this specification is usually expressed by means of relative distinguished names), and one or more values related to the parameter being controlled. We can group the rules into several categories:

- **Certification rules**. Those are used to control some fields included in a certification request, such as the validity period, key type, key length, certificate extensions or alternative names. These rules are usually enforced by the registration authorities in order to validate the certification requests presented by the users. Other rules related to certification are used by the certification authority in order to add some informational extensions to the certificates being issued. These extensions can contain information about the URL of additional services, like an OCSP [15] server, or the URL of an electronic version of the CPS.

- **Reissuance rules**. These rules are applied to certificates which are about to expire. They control whether the certificate can be reissued and the next validity period. Registration authorities are also responsible for enforcing these rules.

- **Revocation rules**. The rules about revocation specify what should be done when a a particular key is compromised, especially when the certification authority's certificate must be revoked. For example, a rule can force an automatic reissuance of all previous valid certificates when a CA certificate is revoked and the new certificate has been already issued.

3.2 Security Policies for VPNs

IPsec [12] is receiving widespread deployment to restrict access or enforce security operations in VPN scenarios [3]. IPsec is a typical policy-enabled networking service, where security functions will be executed properly only if policies are correctly specified and configured. But, current practices imply that IPsec policy databases are manually configured, which is quite inefficient and error-prone for large distributed networking systems. Besides this, the growing number of secure Internet applications and services is doing IPsec policy deployment more and more complex.

Therefore, a policy management system is clearly demanded to automatically configure and manage several IPsec policies, within the same administrative domain or along several domains.

This system, usually called PBNM –policy based network management– is divided into two different components: one distributed, automated, and coordinated management of network services (in our case, IPsec security services) aimed to define policies ruling these services, and one flexible service for discovering, accessing and processing these policies. Both subsystems have just one network element in common: a distributed set of policy stores.

3.2.1 Policy Definition Process

Our own designed and implemented policy management system is trying to provide a common means of specifying vendor and platform independent communication security policies that map onto several heterogeneous freeware or commercial IPsec and IKE stacks, thus enabling the coordinated control of IP-level security services in the context of a network, or security domain, as a whole.

To get this objective, we work on a language designed to express security policies, security domains, and the entities managing these policies and domains. It is currently supporting policies for packet filtering, IP Security (IPsec), and IKE exchanges. However, it has been designed and implemented to be easily extended to express other types of policies.

In this way, policy-based network management automates the control of the network infrastructure by defining security policies and storing then in a common repository, that can be a general policy server, a LDAP Server (as it is our case), a web server, etc. The policies described by this language normally use an IF-THEN approach: "IF certain conditions are present, THEN specific actions are taken", which for the IPsec protocol can be something like "IF conditions include a type of traffic, IP address, and/or TCP/UDP port, THEN actions should include setting certain level of authentication and encryption of traffic".

As we can see in figure 2, these policies are defined using CIM [19] (the short for Common Information Model). This DMTF-defined model provides a common data schema for describing management information for whatever kind of networks, and also the details for integration with other already existent and well-known management models, as the one defined by SNMP. In our specific case, it is used to define all the IPsec-related policies in XML format and store them into a common LDAP repository.

With this we are changing the paradigm from the current hardware-based, manually configured networks to programmable, automated, and policy-based networks. This new paradigm is enabled by new software entities, called Policy Decision Points and Policy Enforcement Points, running on programmable and dynamically configurable network devices, as described in next section.

3.2.2 Policy Recovery Process

In this new network model we have just presented, VPN changes can happen automatically and in real time, based simply on a set of security policies. But to apply these policies in a medium/big set of configurable network devices requires of new communication standard methods.

In this line, the IETF community has defined an integrated framework of standards around one protocol describing the automated interaction and dynamic configuration of services on

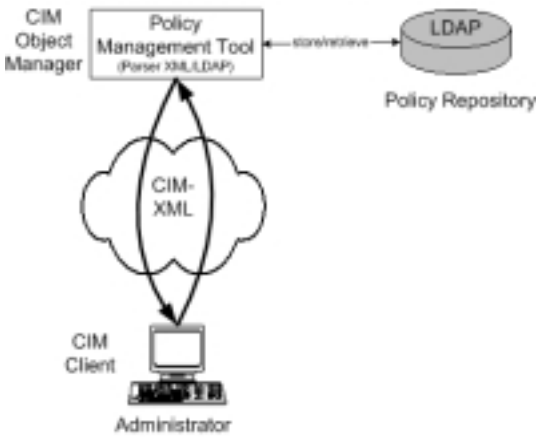

Figure 2: Policy Creation Process

programmable devices. This new framework is called Common Open Policy Service [6], or COPS for short. COPS is the interface between the policy manager (policy decision point, or PDP) and the network device, that usually comprises or interact with (depending on the scenario, as we can see in figure 3) a Policy Enforcement Point, or PEP. This interface [5] is, in our specific case, used between these two elements for exchanging security policy information.

Thus, using COPS protocol we are able to describe quite easily new capabilities for network devices. Under this framework, network devices can be automatically reconfigured by remote processes to implement new security services, enforce updated administrative policies, or even handle end user request for secure network services on the fly.

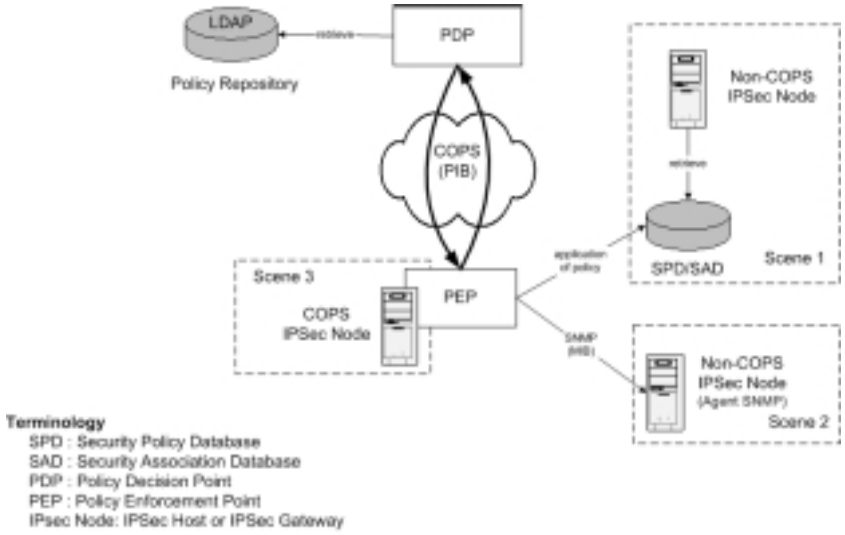

Figure 3: Policy Recovery Process

4 Use of digital certificates as an authorization mechanism

We have designed and implemented a system which addresses the problems related to scalability, certificate distribution, and interoperability in delegation-based systems. DCMS (Distributed Credential Management System) [2] defines how certification requests should be expressed, how different security policies can be enforced using this system, which are the entities involved in a certification scenario, and how these entities can exchange authorization-related information. This system is divided into the naming management system (NMS), which manages the issues related to SPKI ID certificates, and the authorization management system (AMS), which is responsible for those procedures related to SPKI attribute and authorization certificates. We believe that this system can lead up to the definition of an application-independent system which can be used in order to provide authorization services to many different scenarios based on delegation. DCMS also complements some proposed mechanisms for revocation and validation of SPKI certificates [13], and can make use of public repositories for certificate storage purposes [10].

4.1 Naming Management System (NMS)

In this section we are going to present the naming management system, which is responsible for the certification operations related to SPKI ID certificates. This type of certificates can be used to link a name to a particular principal (public key), and also to define group membership. NMS is very useful when authorization is based on group membership.

Naming is not a requirement of distributed systems, but it is worth noting that large-scale SPKI-based delegation systems can be simplified using this mechanism. Naming is an optional tool for group management which can be useful to address scalability of complex systems.

Figure 4 shows the three types of entities involved in NMS: requestors, service access points, and naming authorities. In this section we are going to give a brief description about these core entities, we introduce why they are necessary and how they interoperate.

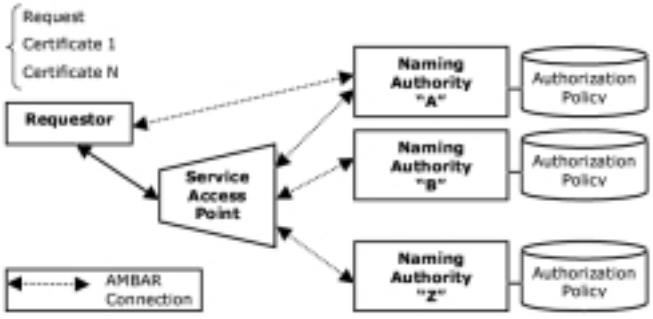

Figure 4: NMS entities

- **Requestor**. A requestor is a principal demanding the generation of a new ID certificate. This entity must create a certification request and must send it to a particular naming authority (NA) in order to obtain the demanded certificate. This submission can be accomplished directly or making use of a service access point. The request can

be accompanied by other certificates in order to demonstrate that the principal has permission to obtain the demanded certificate. There are two types of requestors: first, the principal demanding an ID certificate for a particular public key; second, the principal demanding an ID certificate for a particular name (e.g. a certificate stating that group B is a subgroup of group A).

- **Service access point**. Requestors can make use of access points in order to submit their certification requests to the appropriate naming authorities. Access points are optional, but they are very useful since they provide several additional services to requestors. First, naming authorities can be hidden from users. Moreover, in some scenarios with many authorities, it might be complicated to know which are the appropriate naming authorities for a particular ID certificate (especially with group membership certificates). SAPs can learn that location information from digitally-signed statements containing information about the system structure and properties. It is simpler to distribute this type of information to few SAPs than to all the principals. Communication between requestors and access points is system-dependent, and it ranges from secure connections to public terminals placed at buildings or departments.

- **Naming authority**. Naming authorities are the certificate issuers. They create ID certificates upon the requests received through the access points or directly from the requestors. NAs are controlled by a particular authorization policy, which can be implemented using SPKI ACLs or other mechanisms. Whenever a NA receives a request and its related certificates, it executes a certificate chain discovery algorithm [7] in order to determine whether the certification request must be granted or denied. Inputs to this algorithm are the request, the additional certificates, and ACL entries. If a certificate chain is discovered, the algorithm returns the information that will be used to generate the new certificate.

Certification requests for ID certificates must contain information about the issuer defining the name, the name itself, the intended subject, and validity dates. Encoding can be based on s-expressions [18] since there is no need for making use of new syntax, and this can simplify the authorization process. Thus, requests might be encoded according to the representation form recommended by SPKI for the *authorization tag* field [8]. However, it is worth noting that the data elements contained in a request are also contained in a SPKI ID certificate, and therefore the structure for this type of certificates can be used. It is not necessary to define a completely new structure in order to express certification requests. Moreover, as we will explain, the same structure can be used by ACLs in order to encode authorization policies. S-expressions that we have used for certification requests and ACL entries have the following format:

```
(cert-request
  (issuer (name NA_i N_i^j))
  (subject P)
  (valid ..)
)
```

- *cert-request.* This identifies the s-expression as a certification request.

- NA_i. This is the public key of the naming authority. This authority is responsible for issuing the ID certificates related to the name N_i^j.
- N_i^j. N^j is one of the names defined in the namespace of the authority NA_i.
- P. This is the principal (or principals) requesting the ID certificate. P might be:
  - A public key.
  - A set of entities. There are two possibilities in order to express a set of entities. On the one hand, we can use a group name, i.e., `(name NA N)`. On the other hand, we can use the *-operator *set*, such as for instance `(* set Q R)`, where Q and R must be public keys or names.
- *valid*. This specifies the requested validity period. The structure of this field is the one included in the SPKI standard.

If this s-expression is used as a certification request, P can only be a public key or a name, and it means that a new ID certificate is being demanded, whose issuer will be NA_i, P will be the subject, N_i^j will be the name linked to P, and will be valid during, at most, the specified validity interval. However, if this s-expression is included in the *tag* field of a SPKI-like ACL entry, it means that the principal (or principals) P are authorized to obtain an ID certificate from NA_i, where the name N_i^j will be linked to P (or each of the principals contained in P) during the specified validity period.

Certification requests are encoded as sequences of two elements. The first element is the s-expression specifying the request, and the second one is a digital signature of that sequence. Signatures are encoded using the *signature* structure defined in [8], and they are generated using the requestor's private key. Requests have similar structure to certificates, but certificates are signed by issuers and requests are signed by requestors.

4.2 Authorization Management System (AMS)

In this section we are going to present the authorization management system, which is responsible for certification operations related to SPKI authorization and attribute certificates.

NMS and AMS are based on similar architectural elements. Requestors and access points are also part of AMS. Naming authorities are replaced by authorization authorities (AA), but they share some basic functionality. AAs create attribute and authorization certificates upon the requests received through the access points or directly from the requestors.

An AMS requestor is a principal demanding the generation of a new attribute or authorization certificate. This entity must create a certification request containing information about the authorization tag (the tag is completely application-dependent). Like in NMS, there also are two types of requestors: first, the principal requesting an authorization certificate; second, the principal requesting an attribute certificate for a particular name.

S-expressions used in AMS to specify certification requests are also based on the structure defined by SPKI for attribute and authorization certificates. The main difference between NMS and AMS s-expressions is the *tag* field. This field contains information about the particular authorization being requested (when it is contained in a certification request) or granted (when it is part of an ACL entry).

Certification requests are also encoded as sequences composed by the request itself, and its signature.

5 Conclusions

In this paper we have presented two innovative services that can be built over a Public Key Infrastructure. The first service is related to a new active network management paradigm based on the concept of policy. These policies are distributed along different administrative domains and thus they just can be defined by authenticated network administrators and need to be digitally signed to verify its integrity before being applied to a network service or device.

In this way, Policy-based Network Management systems are providing network administrators the ability to proactively, not just reactively, address the dynamically changing needs of users and networks services and applications, as it is the case of either the VPN scenarios we have presented or our own Java IPv6 PKI.

The second service we have designed and implemented over our PKI is a system which address the problems related to the scalability, certificate distribution, and interoperability in delegation-based systems. DCMS, Distributed Credential Management System, as it is called, defines how certification requests should be expressed, how different security policies can be enforced using this system, which are the entities involved in a certification scenario, and how these entities can exchange authorization-related information.

6 Acknowledgements

This work has been partially supported by the ISAIAS project, code TIC2000-0198-P4-04.

References

[1] C. Adams, P. Cain, D. Pinkas, and R. Zuccherato. *Time-Stamp Protocol (TSP).* IETF, August 2001. Request For Comments (RFC) 3161.

[2] O. Canovas and A. F. Gomez. A Distributed Credential Management System for SPKI-Based Delegation Systems. In *Proceedings of 1st Annual PKI Research Workshop*, Gaithersburg MD, USA, April 2002. Accepted.

[3] O. Canovas, A. F. Gomez, G. Lopez, and G. Martinez. Dynamic virtual private networks. In *2000 SCS Euromedia Conference*, Antwerp, Belgium, May 2000.

[4] O. Canovas, A. F. Gomez, G. Martinez, and et al. Providing security to university environment communications. In *TERENA-NORDUnet Networking Conference 1999*, Lund, Sweden, June 1999.

[5] K. Chan, D. Durham, S. Gai, S. Herzog, K. McCloghrie, F. Reichmeyer, J. Seligson, A. Smith, and R. Yavatkar. *COPS Usage for Policy Provisioning.* IETF, March 2001. Request For Comments (RFC) 3084.

[6] D. Durham, J. Boyle, R. Cohen, S. Herzog, R. Rajan, and A. Sastry. *The COPS (Common Open Policy Service) Protocol.* IETF, January 2000. Request For Comments (RFC) 2748.

[7] J.E. Elien. Certificate discovery using SPKI/SDSI 2.0 certificates. Master's thesis, Massachusetts Institue of Technology, May 1998.

[8] C. Ellison, B. Frantz, B. Lampson, R. Rivest, B. Thomas, and T. Ylonen. *Simple Public Key Certificate*. IETF Internet Draft, draft-ietf-spki-cert-structure-06.txt edition, July 1999.

[9] C. Ellison, B. Frantz, B. Lampson, R. Rivest, B. Thomas, and T. Ylonen. *SPKI certificate theory*, September 1999. Request For Comments (RFC) 2693.

[10] T. Hasu and Y. Kortesniemi. *Implementing an SPKI Certificate Repository within the DNS*, Poster Paper Collection of the Theory and Practice in Public Key Cryptography (PKC 200) edition, January 2000.

[11] IP Security Policy (ipsp) Working Group. Ietf. *Available online at http://www.ietf.org/html.charters/ipsp-charter.html*.

[12] S. Kent and R. Atkinson. *Security Architecture for the Internet Protocol*. IETF, November 1998. Request For Comments (RFC) 2401.

[13] Y. Kortesniemi, T. Hasu, and J. Sars. A Revocation, Validation and Authentication Protocol for SPKI Based Delegation Systems. In *Proceedings of Network and Distributed System Security Symposium (NDSS 2000)*, February 2000.

[14] X. Liu, C. Madson, D. McGrew, and A. Nourse. *Simple Certificate Enrollment Protocol(SCEP)*. IETF Internet Draft, September 2001. Available online at http://www.ietf.org/internet-drafts/draft-nourse-scep-05.txt.

[15] M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. Adams. *OCSP: Online Certificate Status Protocol*, June 1999. Request For Comments (RFC) 2560.

[16] Public-Key Infrastructure (X.509) (pkix) Working Group. Ietf. *Available online at http://www.ietf.org/html.charters/pkix-charter.html*.

[17] O. Prnjat, G. Martinez, and et al. Policy-based Management for ALAN-Enabled Networks. In *IEEE Policy 2002*, Monterey, California, USA, June 2002. Accepted.

[18] R. Rivest and B. Lampson. *SDSI: A simple distributed security infrastructure*.

[19] Common Information Model (CIM) Standards. Dmtf. *Available online at http://www.dmtf.org/standards/standard-cim.php*.

[20] UMU-DIIC. Java ipv6 pki. *Available online at https://pki.ipv6.um.es –IPv6 only–*.

7 Vitae

Antonio F. Gómez received the M.S. degree in Computer Science from the University of Granada and B.S.(Hons.) and the Ph.D. degrees in Computer Science from the University of Murcia, Spain. Since 1993 he is Assistant Professor at the same Department and University. He has worked on different research projects, mainly in the national environment either in the distributed artificial intelligence field (project M2D2), or in the tele-learning and computer support for collaborative work, or the new telematics services in broadband networks (SABA). He is also coordinator of a Socrates CDA (European Master on Soft Computing) and of a Leonardo project for Distance and Open Learning. He is currently collaborating in two IST

projects related to tele-teaching and distance learning, called COLAB and ITCOLE. He is also collaborating with the Euro6IX IST project, aiming to set-up a whole IPv6 network in Europe. He has published over 20 international papers.

Gregorio Martínez received the M.S. degree in Computer Engineering by the University of Murcia (Spain). In 1997 he started to work in the Computer Service of the same University in different projects related with security in communications. In 1999 he started as research staff in the Department of Information and Communications Engineering of the University of Murcia. In 2001, he got a position as lecturer in the same department. His scientific activity is mainly devoted to security infrastructures, smart cards, access control systems, and to the new version of the Internet Protocol (IPv6). He has been working on different national research projects related to these topics. In the international side, he is collaborating with UCL-CS department from May, 2000, working on security and mobility in Internet, distributed systems, and IPv6/IPsec architectures. He is also collaborating in the Euro6IX IST project. He has several papers in national and international conferences and journals.

Óscar Cánovas received the M.S. degree in Computer Engineering by the University of Murcia (Spain). In 1998 he started to work in the Computer Service of the same University in different projects related with security in communications. In the same year, he started as a research staff in the Computer Science Department. From 1999, he is a lecturer in the Department of Computer Engineering of the same University. His scientific activity is mainly devoted to public key infrastructures, authorization management infrastructures and micropayments schemas. He has been working on different national research projects related to these topics. He has several papers in national and international conferences and journals.