Architecting Adaptable Security Infrastructures for Pervasive Networks through Components

Marc Lacoste

Orange Labs, France
marc.lacoste@orange-ftgroup.com

Abstract

Security management in pervasive networks should be fundamentally flexible. The dynamic and heterogeneous character of these environments requires a security infrastructure which can be tailored to different operating conditions, at variable levels of granularity, during phases of design, deployment, and execution. This is possible with component-based security architecture. We illustrate the benefits of this approach by presenting AMISEC, an integrated authentication and authorization middleware. Through the component paradigm, AMISEC supports different network topologies of TTPs, cryptographic algorithms, protocols, or trust management strategies, resulting in a fully à la carte security infrastructure.

1. Introduction

The promise of pervasive computing to be "*optimally connected, anywhere, anytime*'' implies an "optimal" management of security. What does this mean in practice? The large number and heterogeneity of devices, platforms, and networks, their complex, rich, and dynamic relationships □ including a high degree of distribution and mobility □ the absence of boundaries for systems which have not real inside nor outside amount from the security viewpoint to a collection of shifting, contradictory requirements. Protecting such systems thus becomes a real nightmare. This puzzle may only be tackled with a highly flexible security infrastructure, adaptable to changing conditions, to guarantee the most appropriate level of security. Three main issues remain unsolved: identity, privacy, and trust management [21].

Identity management has become a cornerstone of pervasive network security. Services are now accessed under a growing number of partial digital identities. They describe a subset of properties associated with a user, valid in a given context (e.g., car, home, office, etc.), and often linked with "real" identifying information. Many solutions have been proposed to federate identities across multiple domains [38], but are usually not well integrated with mechanisms for effective enforcement of privileges [36].

Privacy should also be addressed, since it is a key element to user acceptance of these new technologies. Several degrees of communications anonymity and unlinkability of interactions are desirable, depending on the service accessed. Yet, privacy-preserving infrastructures are still in their infancy [23], few frameworks being really available [27].

A realistic model of trust for an open environment is needed as well. This notion remains largely not understood, with little agreement on trust models, and mostly closed platforms for managing trust [8]. For all those dimensions, an integrated and flexible security solution is clearly missing to support several security objectives, policies, mechanisms, and protocols.

For instance, to capture different network topologies for the Trusted Third Parties (TTPs) involved in the infrastructure.

This objective is within reach by choosing component-based security architecture. The component paradigm allows to reason in terms of system approach for the design of the infrastructure, with several sub-frameworks dealing with authentication, authorization, privacy, and trust management. This choice makes the infrastructure highly customizable at different levels of granularity depending on how the components are connected and deployed. The infrastructure is also reconfigurable by simple replacement of components.

We illustrate the benefits of this design approach by presenting AMISEC (AMbient Intelligence SECurity), a lightweight *Authentication and Authorization Infrastructure (AAI)*. Thanks to its component-based security architecture, AMISEC provides full flexibility for managing authentication and authorization using certificates, allowing different deployment topologies of TTPs, use of several types of certificates, cryptographic algorithms, security protocols, or strategies for privacy and trust management. We validated our design by prototyping in Java a proof-of-concept implementation on embedded devices. We also evaluated the infrastructure on sample scenarios for the home environment such as seamless authentication, both in connected and disconnected modes. We finally assessed the feasibility of realizing an extension of AMISEC for privacy.

The paper is organized as follows. Sections 2 and 3 first review related work, and give some key requirements for a security architecture for pervasive networks. Section 4 then provides some background on component-based design, and introduces the security model chosen for authentication and authorization. Section 5 describes the design and implementation of AMISEC. Finally, Section 6 presents some evaluation results on the flexibility of the infrastructure.

2. Related work

To guarantee security of pervasive networks, many building blocks have been available for a long-time but in separate contexts. A great number of infrastructures have been proposed for authentication or privilege management, but with no real integration effort. These solutions generally present heterogeneity and scalability issues, with little possibilities for adaptation. The scheme which perhaps most reflects this situation is identity-based entity authentication through exchange of certificates managed by a PKI [29]. Many types of certificates [25][30][48] have been proposed, but infrastructure interoperability remains difficult. PKI architectures are usually quite expensive to deploy and manage. They are thus perceived as too monolithic for pervasive networks. For instance, they generally do not support both hierarchical and P2P topologies of TTPs, where devices may be both clients and certification authorities. Similarly, entity authentication is restricted to verification of identity, but not of other attributes. Some solutions to federate identities have been proposed, but with little support for authorization or privacy [1][2][38]. Some integration efforts have been undertaken [36][37] such as the application of PKIs to authorization through attribute certificates [26], and the development of Privilege Management Infrastructures (PMI) [18][39], but adaptation capabilities remain limited. The agent-based PKI proposed in [28] allows different deployment topologies for TTPs, certificate formats, and protocols, and is similar to our approach, but does not address privacy.

These solutions are generally not functional in disconnected mode. They assume a TTP such as a security server to be available on-line. Many new trust models have been proposed to handle disconnected modes situations [8], such as reputation-based trust management systems [6][7]. Yet, there is no real agreement on an adequate and realistic model of trust. Furthermore, those systems do not really allow tuning the authentication method depending on the connectivity to a TTP.

Privacy-preserving security infrastructures so far received very little attention [15]. Research mostly focused on languages such as [46] to negotiate a level of privacy, and on advanced cryptography such as anonymous credentials [17][19][15] and new types of signatures [20][33][43][45]. Pseudonymous certificates have also been investigated in the context of PKI/PMIs [12][13][22]. By and large, anonymity techniques for Internet [23][24] do not apply well to pervasive environments due to limited resources. A balance between transparent (profile management on a server) and user-controlled (user-driven release of attributes) solutions still remains to be found [27].

Existing solutions thus lack adaptation capabilities in terms of deployment, security services, and protocols. They also require too many resources to be directly usable on limited devices. A more flexible approach to security is therefore required.

3. Architectural requirements

A distinguishing feature of pervasive environments is the dynamic interweaving of a great diversity of networks and devices. The resulting multiplicity of shifting protection requirements calls for a highly flexible security infrastructure, addressing several major challenges:

- **Integrated authentication and authorization**. Identity and privilege management are usually handled separately. Instead, a single *Authentication and Authorization Infrastructure (AAI)* [36] is needed to avoid theft of identities and forgery of credentials. Viewing authentication as the verification of a single identity clearly is insufficient. Federation of multiple partial identities should be considered, to establish authenticity of *attributes* such as location, from which can be derived authorizations.

- **Flexible topologies of authorities**. Attributes are certified by a set of *authorities* which may be organized in a combination of widely different topologies, leading to architectures ranging from centralized to completely decentralized. This complexity is due to the great number and heterogeneity of network nodes, which are also mobile, and scattered across traditional frontiers.

  Relationships between authorities are usually based on *certification*: one authority extends trust or delegates its powers to another. This leads to trust or delegation chains, typically organized in hierarchies as in traditional PKIs, where the root authority has control over its subordinates. Two hierarchies may be connected by a *bridge authority*, or by authorities cross-certifying one another. P2P links between authorities may also established and revoked, leading to more dynamic and decentralized organizations.

  A combination of these approaches where authorities cooperate is also possible. The functionalities of the authority can be distributed among a set of nodes [47]. Another option is to partition the network into clusters, with a single predefined authority node responsible for managing security inside each cluster [10]. For home networks, a hierarchy of device communities which may be split or merged yields a more dynamic structure [5]. In a community, the powers of the authority may be delegated, temporarily or permanently, to another node in case of failure or migration of a device away from home. The active authority may also be randomly shifted among nodes in the cluster for very short time spans [44]. Finally, backup authorities and alternative certification paths [35] make the system more resistant to DoS attacks, without the heavy protection requirements of a PKI root CA.

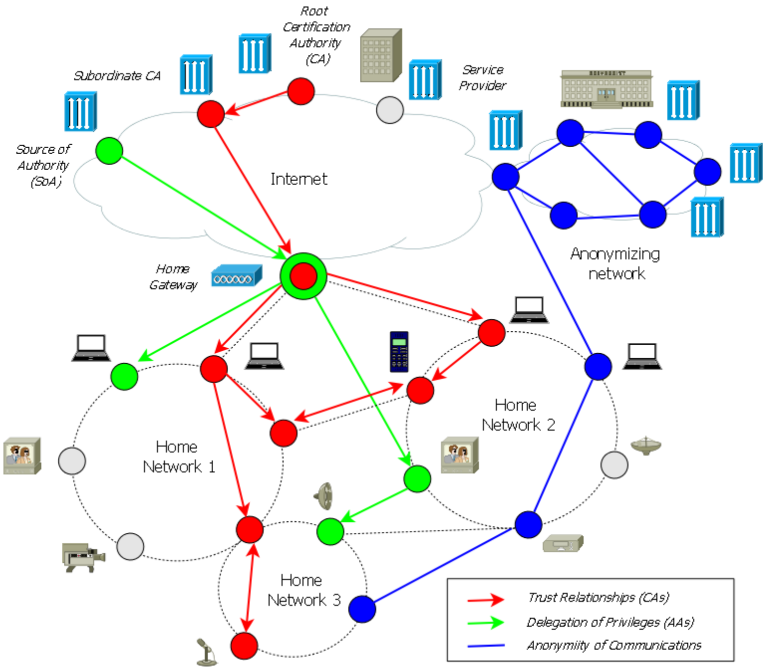

Root Certification Authority (CA)
Subordinate CA
Service Provider
Source of Authority (SoA)
Internet
Home Gateway
Anonymizing network
Home Network 1
Home Network 2
Home Network 3
Trust Relationships (CAs)
Delegation of Privileges (AAs)
Anonymity of Communications

Figure 1. Security authorities in a home network.

To make the problem harder, the topologies of authorities are usually not the same for different security dimensions such as trust management, delegation of privileges, and anonymization of communications. The result is a set of independent overlay networks for each dimension, where nodes in each network are organized into the structures described previously: hierarchical, meshed, P2P, etc. Figure 1 shows a typical pervasive network configuration with three different overlays for authorities.

The infrastructure must thus provide enough flexibility to support these deployment topologies and dynamic relationships between authorities, depending on security requirements. Additional tuning may be necessary to further control delegation, such as introducing path length constraints, or name space restrictions □ for instance, to limit delegation to a subgroup of authorities based on their attribute values.

- **Multiple certificate types and protocols.** The security infrastructure should be open to meet variable security objectives. For instance, it should handle several cryptographic protocols and formats of certificates. Flexibility is needed as well in management protocols [3][40][42] to tailor the security infrastructure to application requirements. This may mean customizing certificate life-cycle management such as enrollment procedures, finding the right trade-off between off-line [29] and on-line validation [40], adapting the security protocols to device and network capabilities, or interoperating with other security infrastructures. Finally, reconfiguration capabilities should also be available to match dynamic conditions of execution, e.g., to add new security mechanisms, download system patches, or personalize security settings.

- **Multiple trust management strategies.** Pervasive network nodes are usually highly decentralized, and follow P2P communication patterns. Those networks lack stable backbones, which results in intermittent connectivity. Therefore, centralized TTP-based trust management solutions may not be adequate. A realistic trust model is therefore required to handle disconnected mode situations. The infrastructure should enable choosing the right strategy for trust management depending on the availability on-line of a TTP, such as certificates validated by a chain of authorities in connected mode, and a reputation management system in disconnected mode.
- **Tunable privacy.** Flexibility in privacy management is also a major enabler of pervasive computing, often at odds with trust management: the user should control disclosure of his personal information, and yet let the infrastructure communicate transparently with TTPs to assess the validity of presented credentials. A minimal disclosure of information is also needed to establish trust relationships between entities. Customizable degrees of anonymity are thus required to select the right trade-offs, the willingness of the user to disclose personal data also depending on its perception of the service accessed.
- **Embedded constraints.** Last, but not least, the security infrastructure should comply with limited computation and communication resources, with lightweight protocols, and minimal footprint on devices. Only the key security services should be included in the infrastructure.

To meet those requirements, we propose to adopt a component-based architecture for the security infrastructure. This approach allows the infrastructure to be adaptable to several types of execution environments, and to be reconfigured according to security objectives, policies, and mechanisms □ either at a fine-grained level (e.g., certificate formats), or at a macroscopic level (e.g., topologies for authorities), depending on how the components are connected and deployed. This architecture naturally leads to a framework-oriented design, specific sub-frameworks dealing with each adaptability dimension (e.g., authentication and authorization, trust, and privacy management), making the infrastructure highly customizable.

In what follows, we recall the main elements of component-based design, and review the PKIX security model we use as basis to illustrate how component-based architectures meet the previous requirements.

3. Background

3.1. Component-based design

Components are usually defined as entities encapsulating code and data which appear in software systems as units of execution, configuration, deployment, or administration. Building a system according to a component model allows mastering the complexity of implementation of a software infrastructure, since components can be composed to form higher-level units of code. The resulting infrastructure is thus very modular. Component-based architectures also offer flexibility of configuration, since functionalities can be adapted or introduced by addition or replacement of components in the system; both in the large and in the small (see Figure 2). This approach is thus well adapted to the dynamic needs of pervasive networks.

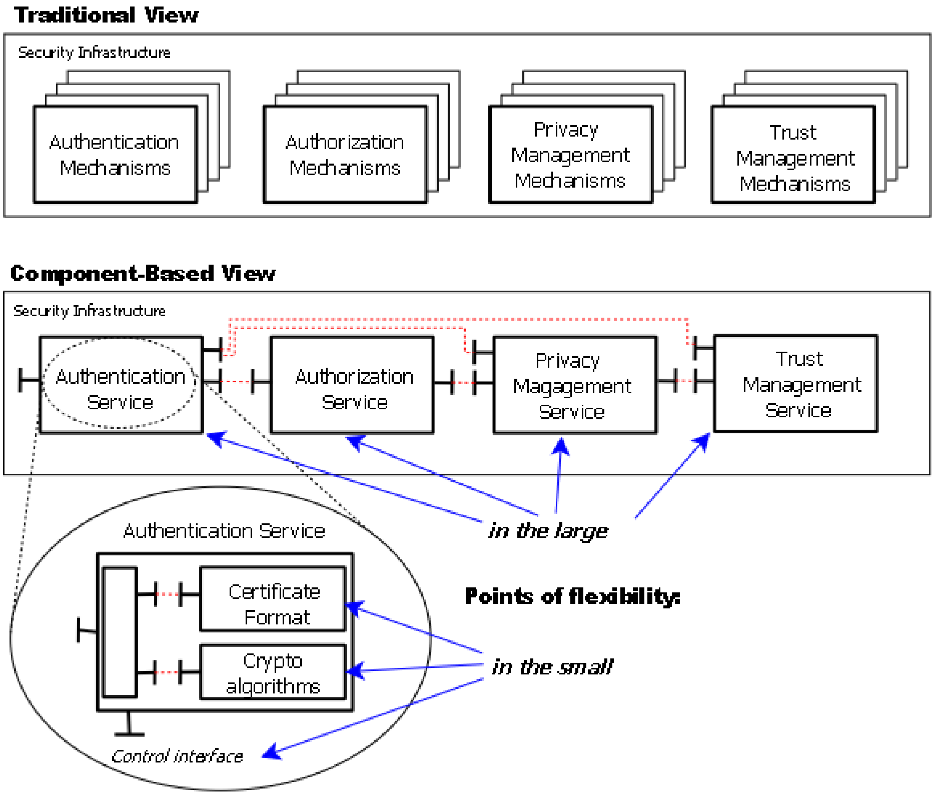


Figure 2. AMISEC approach to security flexibility.

We specify the architecture of the security infrastructure with Fractal [16], a generic component model capturing reconfiguration by flexible composition of components with a minimal number of concepts: a *component* is a run-time entity built from a *controller*, which supervises execution of a *content* possibly including other components (*sub-components*). A *composite component* offers a white-box perspective of its content by revealing its organization, while a *primitive component* is a black-box encapsulating legacy code. A component only interacts with its environment through well-defined access points called *interfaces*. A Fractal component provides and may require interfaces. Interaction between components is performed by establishment of *bindings* between their interfaces.

Fractal manages reconfiguration independently from component functionality by separating control interfaces from functional interfaces. The main control interfaces of the component framework cover: containment relationships and bindings between components (`BindingController`); introspection, e.g., to discover the structure of a component or configure its properties (`AttributeController`); dynamic reconfiguration, e.g., to add or remove sub-components (`ContentController`); and life-cycle management, e.g., to suspend or resume the execution of a component (`LifeCycleController`). A reference implementation of Fractal called Julia is provided to program applications according to the component model [41].

3.2. PKIX-compliant AAIs

The PKIX working group [29][30] proposed a unified reference model for the organization of a certificate-based AAI. Two sub-infrastructures are distinguished: the *PKI (Public Key Infrastructure)* for authentication, and the *PMI (Privilege Management Infrastructure)* for authorization and attribute management. The AAI manages trust and enforces privileges through the exchange of certificates digitally signed by authorities. *Public key (or identity) certificates (PKC)* signed by *Certification Authorities (CAs)* □ the root CA being the trust anchor □ guarantee the link between an identity and a public key. *Attribute certificates (ACs)* signed by *Attribute Authorities (AAs)* □ the root AA also being called the *Source of Authority (SoA)* □ establish the relationship between the identity and a number of attributes. Authorities may delegate issuance of certificates to subordinate or peer authorities.

The main elements of the PKI model are the following. *PKI clients* initiate *Certificate Signing Requests (CSRs)* to ask for a new certificate, check the validity of certificates, or request their revocation, for instance when a public key has been compromised. A *Registration Authority (RA)* is responsible for certificate enrollment and approves CSRs which are transmitted to a CA. It may also trigger revocation of PKCs. One or more CAs verify CSRs, and issue, sign, verify, or revoke certificates. Finally, the *Certificate Repository*, usually implemented by one or more databases, allows storing and retrieving PKCs and *Certificate Revocation Lists (CRLs)*. Additional primitives may be included for certificate renewal, loss or compromise.

The elements of the PMI model are quite similar. The AAs generate, sign, and revoke attribute certificates. They also publish these certificates, and the corresponding revocation lists (ACRLs) in certificate repositories. The AAs are organized in the same structures as the CAs. The AA acts on behalf of the SoA to deliver and manage attribute certificates. *PMI clients* ask for new attribute certificates, or request their verification or their revocation. A *Privilege Verifier* is responsible for verifying the validity of an AC, or revoking the corresponding privileges. Finally, one or more Certificate Repositories allows storing and retrieving ACs and ACRLs.

4. The AMISEC infrastructure

4.1. From high-level security services...

As already shown, in an AAI security authorities (CAs and AAs) may be organized in different network topologies. Moreover, infrastructure services are the result composing several functional components (CAs, certificate repository, AAI clients...), well-described by the PKIX model. Finally, for each security service such as certificate validation, interactions between functional components can be specified with several protocols. As a result, the AAI design space is very large.

We now describe AMISEC, a PKIX-compliant AAI supporting these different types of design. The entirely component-based architecture of AMISEC provides full control over the deployment of functional components, their relationship with security services, and the interaction protocols between components.

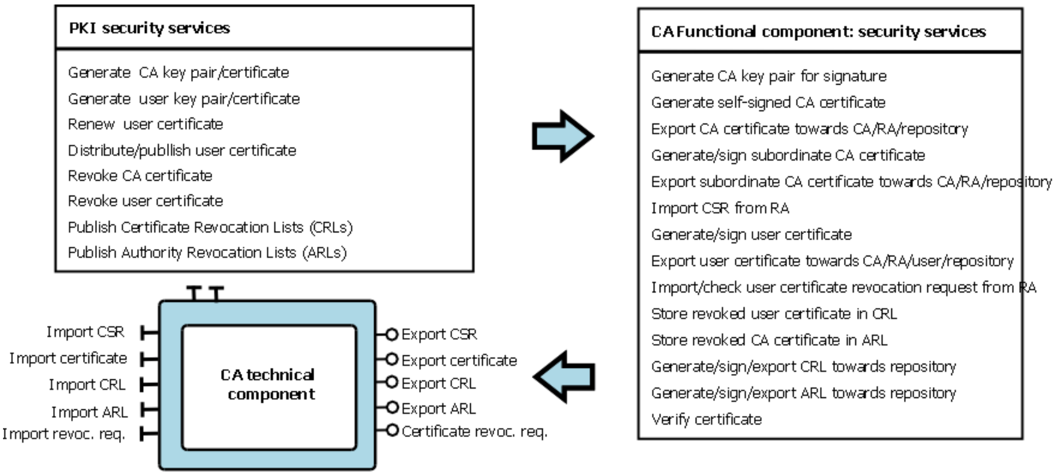

PKI security services
Generate CA key pair/certificate
Generate user key pair/certificate
Renew user certificate
Distribute/publlish user certificate
Revoke CA certificate
Revoke user certificate
Publish Certificate Revocation Lists (CRLs)
Publish Authority Revocation Lists (ARLs)
CA Functional component: security services
Generate CA key pair for signature
Generate self-signed CA certificate
Export CA certificate towards CA/RA/repository
Generate/sign subordinate CA certificate
Export subordinate CA certificate towards CA/RA/repository
Import CSR from RA
Generate/sign user certificate
Export user certificate towards CA/RA/user/repository
Import/check user certificate revocation request from RA
Store revoked user certificate in CRL
Store revoked CA certificate in ARL
Generate/sign/export CRL towards repository
Generate/sign/export ARL towards repository
Verify certificate
Import CSR
Import certificate
Import CRL
Import ARL
Import revoc. req.
CA technical component
Export CSR
Export certificate
Export CRL
Export ARL
Certificate revoc. req.

Figure 3. From security services to technical components.

The main security services provided by AMISEC are shown in Figure 3. Due to lack of space, we only present the PKI services. The PMI ones are similar, covering generation, distribution, validation, and revocation of attribute certificates. These services, independent from the type of design, can be mapped to the main functional PKIX components to reach a description of the services supported by each component, and of the expected interactions between these components. One then obtains a set of technical components which can be arranged very flexibly to realize several types of PKI/PMI architectures (see Figure 4). The interfaces of these components can be specified with an ADL (Architecture Description Language) such as the Fractal ADL [41]. The CA technical component is shown in Figure 3, the other components being similar.

Using the control interfaces provided by Fractal to manage component bindings and containment relationships, the AMISEC technical components may be distributed on the network nodes according to a specific topology for authorities. The topology may be reconfigured according to the context, e.g., by creating a new CA, closer to clients, to optimize communications. Security services can also easily be customized to the execution environment (security objectives, available resources), by adding/removing specific security components in the architecture. This design approach leads to a lightweight infrastructure, where only the minimal required set of security services are included. Finally, new interaction protocols (e.g., for more efficient certificate validation) can be introduced by implementing specific bindings between components. The AMISEC design thus provides adaptability in the deployment architecture, the provided security services, and the supporting security protocols.

5.2. ...To fine-grained tuning of protection

The AMISEC technical components are composite components: they share a number of finer-grained components which allow to tune several AMISEC functionalities. For instance: cryptographic algorithms (`Cryptography` component); format of certificates (`CertificateFormat` component); initialization procedures of entities; certificate life-cycle (creation, signing, publishing, validation, renewal, revocation); local storage of key pairs and certificates; or certificate validation protocols. Thus, the protocols governing interactions between the AMISEC components can be implemented and adapted very flexibly.

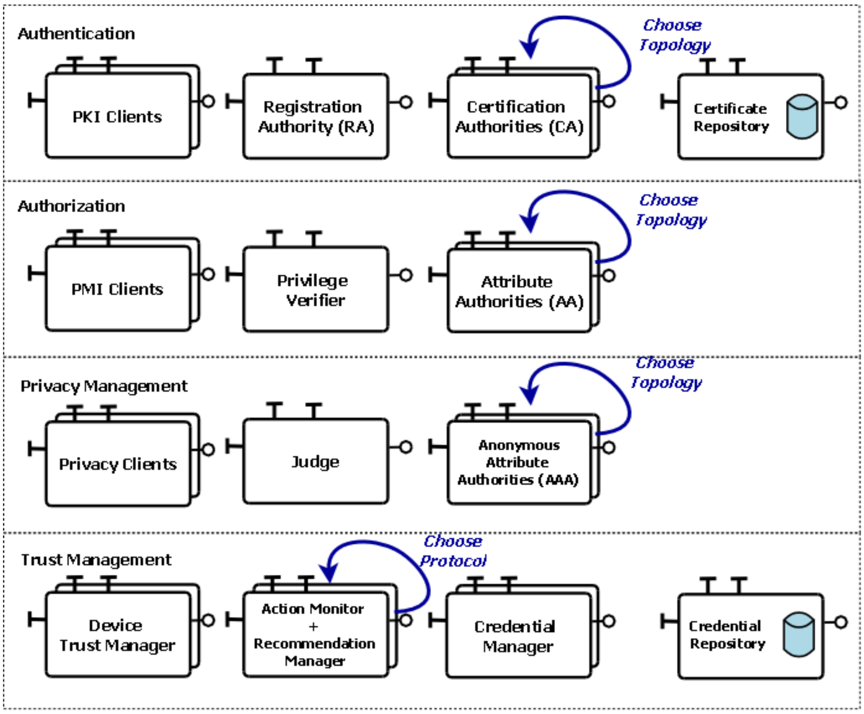


Figure 4. AMISEC technical components

5.3. Implementation

The AMISEC architecture was specified using the Fractal ADL which describes for each component its interfaces (provided, and required), its sub-components and its bindings to other components. The design was then validated by prototyping a distributed Java implementation on the Julia platform. To manage distribution, we used Java RMI as communication protocol between technical components. Exchanged certificates are X.509v3 compliant, ACs having the format described in [26]. For all standard cryptographic operations, we relied on the BouncyCastle Java library. Additional librairies developped internally for group and fair-blind signatures were also used in the privacy extension of AMISEC. The certificate repository was based on an Open LDAP server, encapsulated as a component.

A supervision console was also developed to deploy and manage dynamically each element of the PKIX architecture. We could test simply several configurations of the infrastructure for different topologies of authorities, and control certificate management procedures. The Fractal model was quite helpful to capture dynamic aspects of communication: components could be bound/unbound very easily between entities joining/leaving the network, without needing to know them in advance.

The AMISEC infrastructure is also currently being ported on an OSGi platform, using Apache Felix running on Nokia Internet tablets N800 and N810. We plan to evaluate how the infrastructure can evolve from a purely component-based towards a *Service-Oriented*

Architecture (SOA) where each device may dynamically register and lookup the security services it provides and requires respectively. The objective is study possible integration of this middleware with other SOA architectures for the home environment such as [14].

6. Evaluation

6.1. Extension for privacy

The flexibility of the AMISEC security architecture was validated by specifying and implementing a privacy-enhancing extension to support tunable degrees of anonymity of communications. This feature is highly desirable for user acceptance of digital home networking technologies to prevent observable behaviors from being linkable to user identities, and to selectively control disclosure of user attributes, which may be variably bound with personal identifying information.

Many cryptographic primitives such as anonymous credentials [15][17], fair-blind signatures [45], group signatures [20], traceable signatures [33], or ring signatures [43] have been proposed to build privacy-preserving mechanisms □ e.g., see [11] for a comparison of their anonymity guarantees.

A trade-off must be found between full disclosure of attributes and uncontrolled anonymity, which may lead to abuse by malicious users. We focus on pseudonymity mechanisms which guarantee privacy while preserving accountability. The degree of anonymity can be tuned depending on the type of cryptographic mechanism chosen. The component-based security architecture allows to implement simply this extension in the infrastructure, transparently to the cryptographic mechanism.

An elegant solution to enhance the PMI with anonymity services is proposed in [12]. X.509 ACs are extended into *anonymous attribute certificates (AAC)*, the link between identities and attributes being based on pseudonyms. A TTP guarantees the secrecy of the link between real identities and pseudonyms, but may disclose the identity of the certificate holder under some particular conditions. The scheme described in [12] is based on fair-blind signatures. AACs may also be based on other types of signature schemes [11] such as group signatures [13].

A special type of AA, an *Anonymous Attribute Authority (AAA)*, is introduced to issue AACs. Once users have applied for an AAC to the AAA, the AAC can be used to enforce their privileges in the same manner as a regular AC, except that the process is performed anonymously. This approach thus makes anonymity completely transparent to the authorization process. The AAA may rely on a regular AA for all primitives which are not related to anonymity in the management of certificates.

A simple manner to implement AAAs in the AMISEC architecture is to introduce a new component containing as sub-component a regular AA, as well as other sub-components specific to the chosen anonymity scheme, e.g., a group manager for group signatures-based AACs, a pseudonym manager for fair-blind signature-based AACs, implementations of the anonymity protocols, etc. TTPs may also need to be added in the infrastructure, for instance, to manage the disclosure of identities of the anonymous users.

A richer cryptographic interface is also needed. To support anonymity, the AMISEC `Cryptography` component interface was extended with new methods implementing advanced signature schemes, such as primitives for blinding messages, and generating group signatures. Other primitives such as commitments and zero-knowledge proofs could also be added, for instance following the cryptographic framework [9] which describes the primitives needed to implement privacy-enhancing mechanisms for certificate management infrastructures. The `CertificateFormat` component was also extended to take into account the format of AACs, including fields such as pseudonyms, TTP identities, and special conditions under which the TTP and the AA may collude to reveal user identities.

This design facilitates the support of different anonymity policies. For example, as shown in Figure 5, a user may ask several AACs to different AAs: either to a regular AA configured with the standard cryptographic sub-component if he chooses to fully disclose his identity; or to an AA issuing AACs based on group signatures (AA_1) to be moderately anonymous; or again to an AA issuing AACs based on fair-blind signatures (AA_2) to be strongly anonymous, assuming in this example that the chosen group signature scheme provides weaker anonymity than the fair-blind signature scheme. AA_1 and AA_2 both contain the standard AA as a sub-component, but configured respectively with the cryptographic component for group and fair-blind signatures. The client device then may access transparently the service by presenting the correct AC, depending on the degree of anonymity desired.

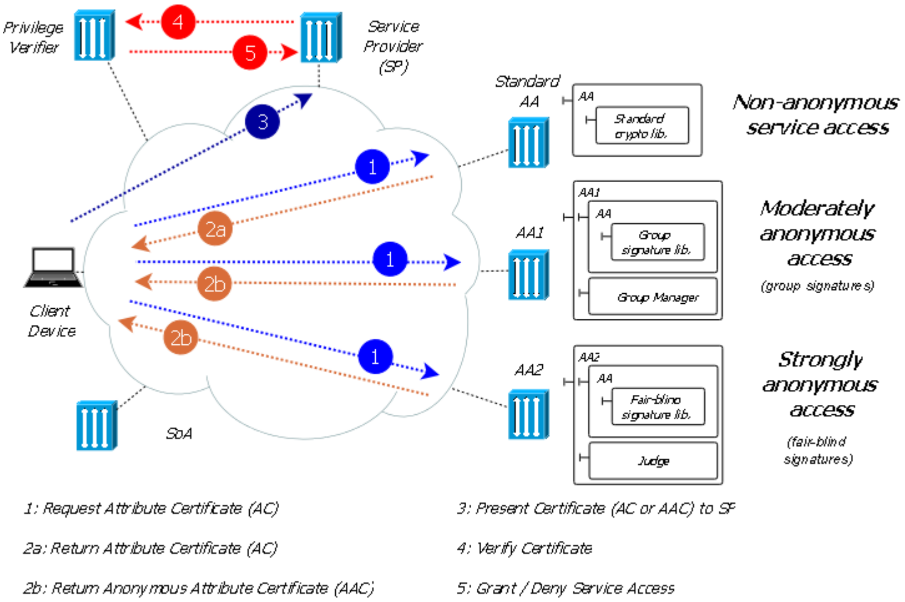

Privilege Verifier
4
5
Service Provider (SP)
3
Standard AA
AA
Standard crypto lib.
Non-anonymous service access
1
2a
1
2b
2b
1
Client Device
SoA
AA1
AA1
AA
Group signature lib.
Group Manager
Moderately anonymous access
(group signatures)
AA2
AA2
AA
Fair-blind signature lib.
Judge
Strongly anonymous access
(fair-blind signatures)
1: Request Attribute Certificate (AC)
2a: Return Attribute Certificate (AC)
2b: Return Anonymous Attribute Certificate (AAC)
3: Present Certificate (AC or AAC) to SP
4: Verify Certificate
5: Grant / Deny Service Access

Figure 5. Tunable anonymity in AMISEC

6.2. Flexible trust management

We also explored the ability of AMISEC to support several strategies for trust management. Indeed, the method of authentication may be adapted depending on the connectivity status to a TTP. We thus consider two modes of operation.

In *connected mode*, a TTP is available on-line to validate credentials. A traditional certificate management infrastructure such as a PKI is therefore applicable. In *disconnected mode*, due to missing information, validation cannot be performed so simply. A reputation-based system might then be preferred to manage trust between devices without central servers. These alternatives may be unified into a single abstract authentication component on each device, the interface of which may be bound to sub-components, either of the PKI, or of the reputation system. The same approach can be followed to adapt the authorization strategy depending on connectivity, using the PMI and a trust-based access control scheme [7].

To assess the feasibility of this adaptation, we included additional Fractal components into AMISEC based on the design of the PTM reputation management system [6]. These components are shown in Figure 4. An *Action Monitor* keeps track of behaviors (normal or malicious) of other devices. A *Trust Manager* combines this information with recommendations received from other devices serve to compute the reputation of each device according to the chosen trust model. A *Recommendation Manager* implements the recommendation protocol between devices. Reputation values are then converted by the *Credential Manager* into credentials for authentication, stored in a repository.

Some of these components are shared with, or can be seen as extensions of the components of the AMISEC PKI. For instance, the Credential Repository and Manager are enhancements of the Certificate Repository and Authority components respectively to manipulate trust information. Similarly, the Recommendation Manager is a connector-type of component binding two Trust Manager components on each device, implementing a specific infrastructure management protocol.

The component-based architecture of the infrastructure also allows to fine-tune some parameters such as the strength of authentication by selecting the threshold T for trust values above which user entities are authenticated in P2P mode, with T=1 for Boolean authentication using the PKI. One could also change the trust model, action monitoring policy, recommendation protocol, or type of exchanged credentials by replacing the corresponding components of the reputation system.

These different adaptability dimensions of AMISEC were evaluated on device authentication scenarios in the home environment, both in connected and disconnected modes. Different trust management schemes were used for each mode, AMISEC enabling smooth transition between schemes depending on the on-line availability of a TTP. A brief overview of this evaluation is given in appendix.

6.3. Some remaining challenges

We showed how the AMISEC architecture offered maximum flexibility in the choice and combination of security mechanisms. Assessing the impact of this flexibility on overall security remains an unsolved issue. Other dimensions may also be involved such as performance or QoS.

The problem may be viewed as selecting the most adequate component assembly, given current security objectives. One solution is to capture security at the component-level by a set of security properties assumed to be composable □ in the form of discrete, continuous values, or even predicates [31]. Overall guaranteed system security can then be derived by aggregating and reasoning on properties of individual components, and compared to targeted security objectives. Properties may be advertised by component-level security contracts [32], expressing adequation of guaranteed security properties with respect to security requirements. This approach may be generalized to richer notions of properties, to express trade-offs between different security or non-security dimensions such as QoS [4], or confidence in security context information [34].

Yet, this approach is based on the (strong) assumption that security properties are composable across components, which is seldom the case. The general case remains yet an unsolved issue, well beyond the scope of this paper.

7. Conclusion

We presented the design and implementation of AMISEC, a certificate-based privacy-enhanced AAI for pervasive networks. Adopting the component paradigm for the security architecture yields a highly flexible infrastructure, which may be adapted both to changing conditions and to shifting security requirements. AMISEC offers an integrated framework for authentication, authorization, and privacy, adaptable both in the large (topologies of security authorities, management protocols, trust strategies...), and in the small (cryptographic algorithms, certificate formats...), while remaining lightweight.

In future work, we plan to formalize better the privacy and trust management frameworks. First, by supporting multiple types of anonymous credentials and defining an abstract interface for anonymity, going beyond a solution purely based on group signatures. Second, by making the trust model and trust information manipulated fully customizable. We also plan to study how a service-oriented evolution of AMISEC might serve as a basis for a reference end-to-end security infrastructure for digital home and multi-homed environments.

References

[1] OpenID, http://openid.net/.

[2] The Liberty Alliance Project, http://www.projectliberty.org/.

[3] C. Adams and S. Farrell. “Internet X.509 Public Key Infrastructure Certificate Management Protocols”, RFC 2510, March 1999, http://www.ietf.org/rfc/rfc2510.txt.

[4] M. Alia and M. Lacoste, “A QoS and Security Adaptation Model for Autonomic Pervasive Systems”, *International Workshop on Security of Software Engineering (IWSSE)*, 2008.

[5] M. Aljnidi and J. Leneutre, “Towards an Autonomic Security System for Mobile Ad Hoc Networks”, *Third International Symposium on Information Assurance and Security (IAS),* 2007.

[6] F. Almenárez, A. Marín, C. Campo, and C. García, “PTM: A Pervasive Trust Management Model for Dynamic Open Environments”, *Workshop on Pervasive Security, Privacy and Trust (PSPT),* 2004.

[7] F. Almenárez, A. Marín, C. Campo, and C. García, “TrustAC: Trust-Based Access Control for Pervasive Devices”, *International Conference on Security in Pervasive Computing (SPC)*, 2005.

[8] D. Artz and Y. Gil, “A Survey of Trust in Computer Science and the Semantic Web”, *Web Semantics: Science, Services and Agents on the World Wide Web*, 5(2), 2007, pp. 58–71.

[9] E. Bangerter, J. Camenisch, and A. Lysyanskaya, “A Cryptographic Framework for the Controlled Release of Certified Data”, *Twelfth International Workshop on Security Protocols*, 2004.

[10] M. Bechler, H.-J. Hof, D. Kraft, F. Rahlke, and L.Wolf, “A Cluster-Based Security Architecture for Ad Hoc Networks”, *Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, 2004.

[11] V. Benjumea, S. G. Choi, J. Lopez, and M. Yung, “Anonymity 2.0 □ X.509 Extensions Supporting Privacy-Friendly Authentication”, *International Workshop on Cryptology and Network Security (CANS),* 2007.

[12] V. Benjumea, J. Lopez, J. Montenegro, and J. Troya, “A First Approach to Provide Anonymity in Attribute Certificates*”, International Workshop on Practice and Theory in Public Key Cryptography (PKC),* 2004.

[13] V. Benjumea, J. Lopez, and J. Troya, “Anonymous Attribute Certificates based on Traceable Signatures”, *Internet Research*, 16(2), 2006, pp. 120–139.

[14] A. Bottaro and A. Gerodolle, "Home SOA - Facing Protocol Heterogeneity in Pervasive Applications", *International Conference on Pervasive Services (ICPS)*, 2008.

[15] S. Brands, *Rethinking Public Key Infrastructures and Digital Certificates: Building in Privacy*, MIT Press, 2000.

[16] E. Bruneton, T. Coupaye, M. Leclercq, V. Quéma, and J. B. Stéfani, "The Fractal Component Model and its Support in Java", *Software – Practice and Experience, special issue on Experiences with Auto-adaptive and Reconfigurable Systems*, 36(11–12), 2006, pp. 1257–1284.

[17] J. Camenisch and A. Lysyanskaya, "Efficient Non-Transferable Anonymous Multi-Show Credential System with Optional Anonymity Revocation", *Advances in Cryptology (EUROCRYPT)*,2001.

[18] D. Chadwick and A. Otenko, "The PERMIS X.509 Role-Based Privilege Management Infrastructure", *ACM Symposium on Access control Models and Technologies (SACMAT),* 2002.

[19] D. Chaum, "Untraceable Electronic E-Mail, Return Addresses, and Digital Pseudonyms*", Communications of the ACM*, 4(2), 1981, pp. 84–88.

[20] D. Chaum and E. van Heyst, "Group Signatures", *Advances in Cryptology (EUROCRYPT)*, 1991.

[21] D. Cook and S. Das, *Smart Environments: Technologies, Protocols, and Applications*, Wiley, 2005.

[22] D. Critchlow and N. Zhang, "Security-Enhanced Accountable Anonymous PKI Certificates for Mobile E-Commerce", *Computer Networks*, 45(4), 2004, pp. 483–503.

[23] G. Danezis and C. Diaz, "A Survey of Anonymous Communication Channels", Technical Report MSR-TR-2008-35, Microsoft Research, 2008.

[24] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The Second-Generation Onion Router", *USENIX Security Symposium,* 2004.

[25] C. Ellison, B. Frantz, B. Lampson, R. Rivest, B. Thomas, and T. Ylonen, "SPKI Certificate Theory", RFC 2693, September 1999, ftp://ftp.isi.edu/in-notes/rfc2693.txt.

[26] S. Farrell and R. Housley, "An Internet Attribute Certificate Profile for Authorization", RFC 3281, April 2002, http://www.ietf.org/rfc/rfc3281.txt.

[27] M. Hansen and H. Krasemann, " Privacy and Identity Management for Europe", IST PRIME Project White Paper, 2005.

[28] Q. He, K. Sycara, and Z. Su, "A Solution to Open Standard of PKI", *Australasian Conference on Information Security and Privacy (ACISP),* 1998.

[29] R. Housley, W. Polk, W. Ford, and D. Solo, " Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002, http://www.ietf.org/rfc/rfc3280.txt.

[30] IETF, PKIX Working Group, http://www.ietf.org/html.charters/pkix-charter.html.

[31] K. Khan and J. Han, "A Security Characterisation Framework for Trustworthy Component-Based Software Systems, International *Computer Software and Applications Conference (COMPSAC)*, 2003.

[32] K. Khan and J. Han, "Deriving Systems Level Security Properties of Component-Based Composite Systems", *Australian Software Engineering Conference (ASWEC)*, 2005.

[33] A. Kiayias, Y. Tsiounis, and M. Yung, "Traceable Signatures", *Advances in Cryptology (EUROCRYPT)*, 2004.

[34] M. Lacoste, G. Privat, and F. Ramparany, "Evaluating Confidence in Context for Context-Aware Security", *European Conference on Ambient Intelligence (AmI),* 2007.

[35] J. Lee, M. Lee, J. Gu, S. Lee, S. Park, and J. Song, "New Adaptive Trust Models against DDoS: Back-Up CA and Mesh PKI", *Second International Conference on Human.Society@Internet (HSI)*, 2003.

[36] J. Lopez, R. Oppliger, and G. Pernul, "Authentication and Authorization Infrastructures (AAIs): A Comparative Survey", *Computers & Security*, 23(7), 2004, pp. 578–590.

[37] F. Mendoza, M. Carbonell, J. Forné, F. Hinarejos, M. Lacoste, A. M. López, and J. Montenegro, "Design of an Enhanced PKI for Ubiquitous Networks", *International Workshop on Secure Ubiquitous Networks (SUN)*, 2005.

[38] T. Miyata, Y. Koga, P. Madsen, S. Adachi, Y. Tsuchiya, Y. Sakamoto, and K. Takahashi, "A Survey on Identity Management Protocols and Standards", *IEICE - Transactions on Information and Systems*, E89-D(1), 2006, pp. 112–123.

[39] J. Montenegro and F. Moya, "A Practical Approach of X.509 Attribute Certificate Framework as Support to Obtain Privilege Delegation", *European PKI Workshop: Research and Applications (EUROPKI),* 2004.

[40] M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. Adams, "Internet X.509 Public Key Infrastructure: Online Certificate Status Protocol – OCSP", RFC 2560, June 1999, http://www.ietf.org/rfc/rfc2560.txt.

[41] ObjectWeb Consortium, The Fractal Component Framework, http://fractal.objectweb.org/.

[42] D. Pinkas and R. Housley, "Delegated Path Validation and Delegated Path Discovery Protocol Requirements", RFC 3379, September 2002, http://www.ietf.org/rfc/rfc3379.txt.

[43] R. Rivest, A. Shamir, and Y. Tauman, "How to Leak a Secret", *Advances in Cryptology – International Conference on the Theory and Applications of Cryptology (ASIACRYPT),* 2001.

[44] G. Safdar and M. McLoone, "Randomly Shifted Certification Authority Authentication Protocol for MANETs", *Mobile and Wireless Communications Summit*, 2007.

[45] M. Stadler, J. Piveteau, and J. Camenisch, "Fair Blind Signatures", *Advances in Cryptology (EUROCRYPT)*, 1995.

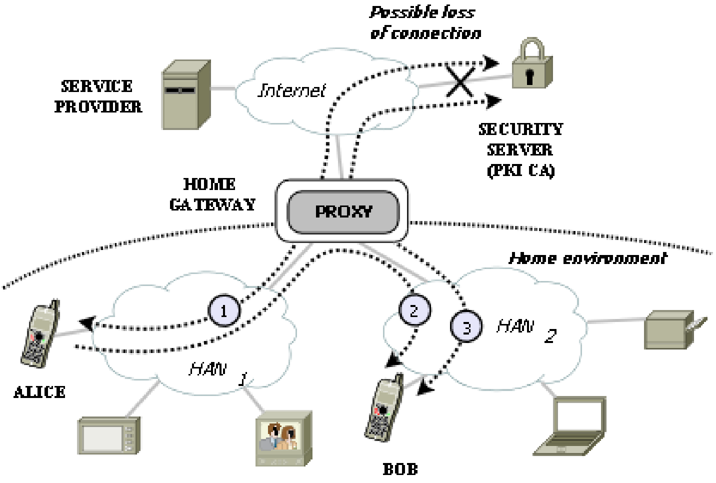
[46] W3C, The Platform for Privacy Preferences (P3P) Project, http://www.w3.org/P3P/.

[47] L. Zhou and Z. Haas, "Securing Ad Hoc Networks", *IEEE Network*, 13(6), 1999, pp. 24–30.

[48] P. Zimmermann, *The Official PGP User's Guide*, MIT Press, 1995.

A. Flexible authentication in home area networks

The AMISEC infrastructure was tested in the environment shown in Figure 6, where devices connected to Home Area Networks (HANs) access Internet services through a residential gateway. To determine the users to trust and protect HAN resources, device authentication is required, but difficult to achieve when a security server is not available on-line.



1: certificate request/reply; 2: certificate transmission; 3: certificate verification

Figure 6. An authentication scenario.

With AMISEC, authentication is possible both in connected and disconnected modes by using the gateway as a proxy for the CA. A proxy is installed on the gateway to cache and forward certificates when the CA is on-line. Authentication is then based on the latest cached version of the certificate. The cache is updated periodically by synchronization with the certificate repository contained in the PKI security server. Most components of AMISEC for connected mode (CA, RA, and certificate repository) are installed on the PKI security server, the client components being deployed on the devices. The proxy is a surrogate for the full PKI, and thus shares with it many interfaces (e.g., credential validation or revocation status checking). AMISEC components for disconnected mode are deployed both on the gateway and the devices.

When Alice asks the proxy for a certificate, the request is forwarded to the on-line CA. A valid certificate is returned to Alice, the proxy caching the certificate and its revocation status. When Alice's certificate is presented to Bob, he can verify her identity by querying the proxy, which in turn will ask the CA to check the validity of the certificate and return the response to Bob. When the connection to the CA is lost, the validation process is similar, but the response of the proxy is based on locally cached information, e.g., the certificate revocation status, dating back to the last synchronization with the PKI certificate repository.

Thus, authentication is possible both when the CA is on-line and off-line. Performance is also increased by placing the authentication data closer to the devices to authenticate, since the connection to an external security server may be costly. Authentication is also possible P2P between devices (e.g., to exchange directly multimedia content between Alice and Bob) through recommendations from other devices or from the gateway, by using the components of AMISEC for disconnected mode. The strength of authentication and type of credentials used (certificates vs. trust values) may thus be freely chosen.

One can also configure parameters such as the frequency of synchronizations between the proxy and the PKI server, which directly impacts the freshness of the security data used for authentication. Thus, the degree of trust granted to the authentication process can be tuned and weighed up against the estimated risk and performance requirements.

Author

Dr. **Marc Lacoste** graduated from Ecole Polytechnique and Télécom Paris, and received a PhD in Computer Science from the University of Grenoble, France. In Orange Labs, he is a senior research scientist on security architectures. His research spans several areas of security such as flexible protection in embedded systems, lightweight public-key infrastructures, and privacy-enhancing technologies. Member of the ACM and IEEE, he also published numerous security research papers in international conferences, journals, or book chapters, and holds several patents.