

Trust and the Establishment of Ad-hoc Communities

Sye Loong Keoh and Emil Lupu

Department of Computing, Imperial College
180, Queen's Gate, London SW7 2BZ

{slk, ecl1}@imperial.ac.uk

1. INTRODUCTION

The proliferation of computing devices, which are being progressively embedded in the objects of everyday life, gives rise to numerous interactions and “collaborations” between these devices and as many security concerns. Applications increasingly rely upon the services provided by other computing devices and this dependence is exacerbated when devices are mobile, autonomous and interconnected through inherently unreliable wireless links. Computational devices which interact frequently form *communities* that follow well defined rules governing their behaviour. For example, implanted or wearable medical devices on a patient collaborate in order to monitor the healthcare of the individual, PDAs and laptop computers of different persons can form a community in an ad-hoc business meeting and Virtual Organisations are frequently formed between businesses to share resources and participate in joint ventures. The security issues which need to be addressed in such communities include authentication, communications security, community membership and access control to the resources shared in the community. The rationale for clearly specifying the rules which govern the behaviour of the participants in the community is to build *trust* between the community participants. Trust in this case represents the expectation that the participants will enforce the rules defined in the community specification (or *doctrine*) and that the membership of the community will be governed by clearly defined constraints. This does not preclude that one, several or all of the community participants contribute to the enforcement of the rules specified in the *doctrine* or that participants monitor each other's behaviour. The amount of redundancy and verification that occurs in the enforcement of the *doctrine* depends on a variety of factors including the computational capabilities of the devices involved. Thus, the security issues in the establishment of ad-hoc communities of mobile devices (e.g., PDAs) which communicate over wireless links are particularly interesting as the limited computational capabilities of the devices forces them to rely upon one another for the enforcement of the security policies. This paper aims to provide an overview of the issues involved, present some of the solutions that we are currently working on, and discuss some of the “trust” assumptions underpinning the framework.

2. ESTABLISHMENT OF AD-HOC COMMUNITIES

In this paper we consider communities at the application level ignoring aspects relating to the confidentiality of the information while in transit or issues relating to ad-hoc routing. Instead we focus on the access control and authentication aspects linked to the establishment and evolution of ad-hoc communities. In essence, the community *doctrine* is a specification which clearly defines: the roles of the participants in the community, the rules (or policies) governing their behaviour in terms of authorisations, obligations, etc. and constraints which external entities have to satisfy in order to join the community. In the enforcement of the community it is necessary to ensure that: only the eligible users are allowed to participate in the community, and each of them is allocated a set of access privileges to use the resources and services provided by the other members of the community. From an implementation perspective the concept of a community *doctrine* integrates the use of access control policies, with the assignment of roles to external entities based on their certifiable attributes rather than identities [3] and with role-based access control [2],[6]. In addition, a set of security protocols on the formation and evolution of the community is required and forms an integral part of the management of the community.

Typically the *doctrines* are specified as templates parameterised by the participants in the community and the running context and are made publicly available to their potential users. The *doctrine* template can then be used to instantiate several communities with different participants. However it is necessary for all users to have a copy of the *doctrine* in use by the community they wish to join. As mentioned above a *doctrine* specification contains the role type specifications, the user-role assignment policies (or constraints), policies governing the behaviour of the entities assigned to the roles and constraints on the permissible community instances. Furthermore, the *doctrine* can specify an initial set of trusted key specifications in order to facilitate the establishment of the community.

2.1 Role Type Specifications and User-role Assignment Policies

A *doctrine* defines its name, purpose, and the time of its creation (timestamp). In addition, it contains role type specifications to assign users into various roles. For example, roles can be defined as *Instructor*, *Student*,

Printer, and *Projector* for a lecture community. Each role type specification defines a set of user-role assignment (URA) policies and is expressed as a list of credential requirements $\{r_1, r_2, \dots, r_n\}$. A mobile user who wishes to join the community must ensure that at least one of the credential requirements is fulfilled prior to the role assignment. In addition, the proposed specification can also cater for the role assignment which requires the user to satisfy more than one credential requirement. A credential requirement, r designates the credential or certification that a user or a computing service must exhibit in order to demonstrate his/her/its possession of certain identity, characteristics or attribute information such as the position in an organisation, professional membership etc.

2.2 Role-permission Assignment Policies

Role-permissions assignment policies are specified in the *doctrine* to grant users access to services and permissions to use resources. These policies are expressed using XML-based Ponder policies [1] and they are grouped according to the role type specification. An authorisation policy defines what a subject role is permitted or prohibited to do to a target role. It protects the target roles from unauthorised actions and this policy is enforced at the target. In a mobile ad-hoc environment, users who provide services are required to enforce these access control policies and all service requests are then granted based on the policies. This implies that all the service providers must be able to interpret the policies and subsequently enforce them. This is similar to the Law-Governed Interactions (LGI) [5] that requires a trusted controller at each site to enforce the law of the group.

2.3 Trusted Key Specifications

When verifying credentials, e.g. a public-key certificate, the public-key of the issuer is required to check the authenticity of the certificate. Since it is unlikely that the user maintains the public-key information of all the Certification Authorities (CAs) and Attribute Authorities (AAs), and the lack of a continuous online connection to the fixed network infrastructure, it is difficult to verify the user's credentials without having the public-key information of the credential issuer. Hence, required public-key information of the CAs and AAs can be included in the *doctrine* as trusted key specification in order to ease the verification process.

2.4 Constraint Specifications

Constraints are used to express security requirements of a community by imposing conditions on the cardinality of membership and guaranteeing separation-of-duty among the participating users. These constraints are evaluated when the community is first established and subsequent changes to the membership of the community trigger the re-evaluation of these constraints. Three types of constraints can be defined, namely *mutual exclusion*, *cardinality* and *community establishment*. The *mutual exclusion* constraint guarantees separation-of-duty among conflicting user roles, while the *cardinality* constraint limits the number of users that can be assigned to a role. Finally, the *community establishment* constraint defines the conditions on the pre-establishment of a community. It ensures amongst others that the user roles and services that are indispensable for the community to function properly are available prior to the establishment of the community.

3. TRUSTING THE PEERS IN THE COMMUNITY

Mobile ad-hoc environments are characterised by wireless transmissions which are unreliable or altogether absent. Therefore, devices in a community have to rely on both knowledge and services provided by the other members of the community. Knowledge includes certificates that they possess or that they have verified while services include routing, certificate verification, access decision services or purely computational services. While in fixed networked environments a public-key infrastructure and attribute certificates are used for authenticating external entities, these are not easy to use in ad-hoc mobile communities. In particular devices sometimes do not have the computational power to verify large numbers of attribute certificates, they lack trusted knowledge of the public-keys of certification and attribute authorities and do not have the on-line connection which would enable them to verify the Certificate Revocation Lists (CRLs). In a scenario where the community is based on peer-to-peer communication and entirely disconnected from any fixed network infrastructure the entities have to rely solely on the information they already have and can exchange between them. However, real-life situations are rarely as extreme and often at least one device will have some form of intermittent connection.

In order to increase the chances that certificates presented by external entities can be successfully verified, we allow entities in the community to exchange information regarding the certificates they know or they have verified. In essence, we advocate the use of credential assertions [4], obtained from peers, in order to ascertain a user's identity, role, membership in groups or other attributes. These assertions (a.k.a. Assertion Statement, ASS) permit authorisation decisions to be made based on credential information that has been attested to by peers. This implies that peer participants in the community are trusted not only to abide by the policies of the community but also in the truthfulness of the assertions they make. Note, however that even for assertion statements one could distinguish different degrees of trust. For example, it is possible to trust only assertions regarding credentials that have been *partially verified*. In this case an entity will only trust assertions for

certificates where the signature has been verified but where revocation status could not be ascertained. In addition, each user maintains a key ring of his/her own that contains a list of trusted public-keys and similar to PGP [7], all public-keys in the ring are assigned a trustworthiness level (Refer to [4] for more details).

In principle each community member would need to ascertain for itself that each of the other member satisfies the URA policies specified in the *doctrine*. This is not practical as it leads to a significant amount of redundant calculations. Thus a particular node (usually the one with most computational resources) will be assigned the task of *coordinating node* (see Figure 1) and will assume the responsibility of checking these policies and responding to admissions requests. When a user A requests to join the community, the request is forwarded to the CN. There are then two possible scenarios:

- **User A presents his/her credentials only.**
The CN broadcasts user A's credentials to all members of the community. If any of the members has an intermittent connection to the fixed network or possesses *out of band* knowledge about user A's credentials, he/she can verify the credentials. If successful, he/she issues an ASS to the CN. The CN then admits user A into the community if the presented credentials conforms to the URA policies.
- **User A presents his/her credentials together with an Assertion Statement (ASS).**
The CN broadcasts user A's credentials and ASS to all members. All the members check whether or not they have connections to the fixed network or possesses *out of band* knowledge. If not, they proceed to checking the signature in the ASS against its own key ring to determine whether the assertions can be accepted based on the configurable policy. If yes, it issues a new Assertions Statement to the CN attesting to the validity of the user A's credentials. User A is admitted into the community if the presented credentials satisfy the URA policies.

A configuration policy (see Figure 1) can be used in addition to the *doctrine* in order to ensure a uniform configuration of the trust policies (and in particular trust in the CN) across all the entities in the community. Finally, some entities may have insufficient computational capabilities to evaluate complex constraints for access control decisions. In these cases the evaluation of the authorisation policies can be devolved to a particular entity in the community in a similar way. This implies that the entity is trusted to enforce the policies specified in the *doctrine*.

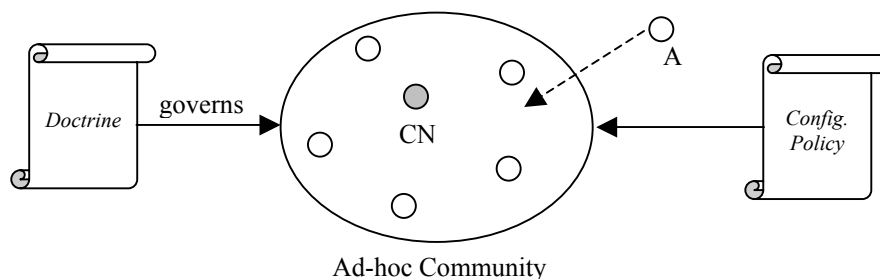


Figure 1: The *doctrine* governs the interactions of users in the ad-hoc community, while the configurable policy is used to specify the rules for accepting Assertion Statements (ASSs).

Although there is not any continuous connectivity to the fixed network infrastructure, the enforcement of the URA policies to check the user's credentials can still be performed when a simple trust framework is incorporated. In addition to this, it helps to enforce the role-permission assignment policies since the enforcement of access control policies is very much depending on the user-role assignment.

4. CONCLUSION

We have presented an approach towards providing policy-based management and coordination of ad-hoc communities based on the *doctrine*. This approach was chosen because in order to facilitate interactions among autonomous mobile devices, users have to be aware of who they are communicating with, what access privileges they have, what services are available, what the community's existing context is and how they verify other peers' credentials. All this security information can be expressed in the *doctrine*.

We have also briefly described an initial approach towards providing some level of trust for the establishment of ad-hoc communities based on assertions issued by trusted peers. In the absence of on-line connectivity to a network infrastructure, and in particular to the issuers of certificate and attribute authorities, the enforcement of the URA policies and verification of user credentials must rely upon the information provided by the peers. This approach is particularly suited when some degree of intermittent connectivity exists and draws significant advantages from *out of band* knowledge between peers.

As a conclusion, the approach taken towards *trust* is rather simplistic and based on PGP's web-of-trust concept. Although some researchers argue that PGP's web-of-trust is only suitable within a circle of close friends, ad-hoc networks in mobile computing environments have strong restrictions in terms of access to a supporting networking infrastructure. Thus, it is inevitable that nodes have to rely on security relevant information held by the peers. A more sophisticated policy-based approach towards trust is however desirable in order to limit the trust placed in assertions made by peers to specific types of assertions and only issued under certain conditions. This is necessary because it provides the ability to determine the authenticity of a user's credentials more precisely and accurately. Moreover, trust can be complemented by additional specifications of recommendations and past experience of users.

5. REFERENCES

- [1] Damianou, N., Dulay, N., Lupu, E., and Sloman, M. The Ponder Policy Specification Language. In *the 2nd International Workshop on Policies in Distributed Systems and Networks (POLICY 2001)*, Bristol, UK, 2001.
- [2] Ferraiolo, D., and Kuhn, R., Role-based Access Controls. In *15th National Computer Security Conference*. NIST, October 1992, 554-563.
- [3] Herzberg, A., Mass, Y., Mihaeli, J., Noaor, D., and Ravid, Y. Access Control Meets Public Key Infrastructure, Or: Assigning Roles to Strangers. In *IEEE Symposium on Security and Privacy*, IEEE Computer Society, Berkeley, CA, 17-14 May 2000, pp. 2-14.
- [4] Keoh, S.L., and Lupu, E. Towards Flexible Credential Verification in Mobile Ad-hoc Networks. In *the 2nd Annual International Workshop on Principles of Mobile Computing (POMC'02)*, Toulouse, France, October 2002, pp. 58 – 67.
- [5] Minsky, N.H., Ungureanu, V., Law-Governed Interaction: A Coordination and Control Mechanism for Heterogeneous Distributed Systems. In *ACM Transactions on Software Engineering and Methodology 9(3)*, 2000, pp. 273 – 305.
- [6] Sandhu, R.S., Coyne, E.J. Role-Based Access Control Models. In *IEEE Computer 29(8)*, 1996, pp. 38 – 47.
- [7] Zimmermann, P. *The Official PGP User's Guide*. MIT Press, Cambridge, MA, 1995.