Proceedings of the 2nd ACM Annual Workshop on Principles of Mobile Computing (POMC'02), Toulouse, France, Oct. 2002

Towards Flexible Credential Verification in Mobile Ad-hoc Networks

Sye Loong Keoh
Department of Computing
Imperial College, 180, Queen's Gate
London SW7 2BZ, U.K.
+44 20 7594 8449
slk@doc.ic.ac.uk

Emil Lupu
Department of Computing
Imperial College, 180, Queen's Gate
London SW7 2BZ, U.K.
+44 20 7594 8249
e.c.lupu@doc.ic.ac.uk

ABSTRACT
Ad-hoc networks facilitate interconnectivity between mobile devices without the support of a network infrastructure. In this paper we propose a flexible credential verification mechanism, which improves the likelihood that participants in an ad-hoc network can verify each other's credentials despite the lack of access to certification and attribute authorities. Users maintain Credential Assertion Statements (CASs), which are formed through extraction of X.509 and attribute certificates into an interoperable XML form. Trusted entities that can verify the credentials listed in the CAS can then issue signed Assertion Signature Statements (ASSs) to other participants in the ad-hoc network. In addition, each user maintains a key ring, which comprises the list of public-keys trusted to sign credential assertion statements. All public-keys in the ring are assigned a trustworthiness level. When a user presents his/her CAS together with matching ASSs to a verifier, the verifier checks the signatures in the ASSs against its key ring to determine whether credentials in the CAS are authentic and acceptable. Transitivity of trust is generally not allowed, but there are exceptional cases in which it is permitted.

Categories and Subject Descriptors
D.4.6 [**Operating Systems**]: Security and Protection - *Authentication, Access Control*; K.6.5 [**Management of Computing and Information System**]: Security and Protection – *Authentication, Unauthorized Access.*

General Terms
Security, Management

Keywords
Security, Authentication, Credential Verification, Trust

POMC'02, October 30-31, 2002, Toulouse, France.

1. INTRODUCTION
Several recent projects have focussed on the design and implementation of ubiquitous, pervasive, wearable and invisible computing environments [4, 12, 15]. However, addressing the security issues arising in such environments where dynamic ad-hoc communities of devices may be formed remains a significant research and engineering challenge. Addressing user concerns on their security and privacy implications will play a significant role for their acceptance with the general public.

Ad-hoc networks that comprise several mobile computing devices connected through wireless links (i.e. Bluetooth, Infrared, Wave LAN etc) are vulnerable to security attacks ranging from passive eavesdropping on the wireless transmissions to active attacks such as message modification, spoofing, masquerading and impersonation. Communication in an ad-hoc network can be established everywhere without the aid of a central infrastructure. The vision of ad-hoc networking is to enable users who carry portable devices such as PDAs, handphones and laptops to establish ad-hoc communities in which they can communicate with each other, share resources and provide services to each other (e.g., ad-hoc routing). Usually, ad-hoc networking scenarios assume that none of the entities that are part of the ad-hoc network has a connection to a fixed network infrastructure, even intermittently, and that the entities that are part of the ad-hoc network have no *a priori* knowledge of each other. This makes the task of verifying credentials (including authentication credentials) presented by one entity to another very difficult. The absence of any network connectivity implies that certification authorities are unreachable. The absence of any trust relationships between the entities implies that security relevant information that is shared between them may not be relied upon. In many "real-life" scenarios the assumptions are however not so stringent. As wireless networks are deployed in homes, airports and public places, at least one of the entities may have some form of intermittent access to the network. This may be even a low bandwidth connection through a cellular phone. Furthermore, mobile entities usually form communities when there is some purpose to the collaboration. This may be ad-hoc business meetings, meetings between colleagues, friends etc. In all these cases, some *a priori* trust relationship between the entities exists or can be established *out of band*. This paper aims to provide a flexible means of verifying credentials in such situations.

2. MOTIVATION AND AIMS

X.509 has been used to provide authentication across networks. An X.509 certificate, which binds a user's public-key to the user's distinguished name (DN) is signed by a Certification Authority (CA) under provision of its certificate policies and Certificate Practice Statement (CPS). Therefore, in a typical wired network environment, the validity of a certificate can be verified by obtaining an authentic copy of the CA's public-key. In order to verify the latter, an authentic copy of its issuer's public-key must be obtained. This gives rise to a certificate chain, which must be traceable to some unique root authority with a public key that is known *a priori*. In practice, there are many different 'root' authorities even if one considers authentication credentials alone. However, authorisations decisions may not be based on the identity of the accessing subject alone but also on its attributes such as its role in a given company, certification by a professional body, etc. This would require being able to verify the authenticity of additional entities. Thus, to verify the credentials received, an entity must either have a large database of *known* public keys or be able to access the information on-line. On-line access is also needed in order to be able to verify the revocation status of a credential.

However, mobile entities in an ad-hoc network which want to form a community have neither on-line access nor sufficient memory resources to maintain a large database of keys. In the worst case scenario, when a user (Client) presents a chain of certificates with its root CA unknown to another user (Verifier) and the verifier does not have a connection to the base station to obtain the public-key of the root CA, the verifier is not able to verify the certificate chain, thereby not being able to grant permissions according to the credentials presented. The client will be denied entry to the ad-hoc network and its resources although he/she may have possessed valid credentials.

Mobile devices such as PDAs or mobile phones also have limited computational resources. Verifying cryptographic signatures is computationally intensive, thus using precious CPU and battery resources. It is therefore desirable to reduce the number of verifications needed without having to rely on a large number of Certification Authorities (CAs) and Attribute Authorities (AAs).

In this paper, we propose a flexible verification mechanism to ascertain a user's identity and credentials based on assertions from peers. The approach proposed, which is similar to PGP's web-of-trust concept, also aims to lessen the number of signature verification needed and to provide interoperability between different certificate formats and paradigms including X.509, PGP and SPKI, providing X.509 and SPKI authentication if there is a connection to the wired network and PGP style verification when there is none.

This paper is set out as follows: In section 3, we briefly discuss the related work and in section 4 we outline the system requirements. Section 5 describes the architecture whilst section 6 discusses the limitations of the proposed framework and outlines possible extension for future work. The conclusions are given in section 7.

3. RELATED WORK

Hybrid trust models [10] permit cross-certification between independent CAs and are structured as follows: first, there are multiple root CAs; second, all non-root CAs are certified within a root CA's hierarchy; third, root CAs cross-certify each other in order to establish a link between one hierarchy to another via a single cross-certification at the root level; lastly, selected cross-certification between non-root CAs is permitted. Such hybrid models facilitate the establishment of separate subordinate hierarchies between independent hierarchies. In addition, they permit relaying parties to use their root CA as an anchor point to discover certification paths to other foreign entities within the public key infrastructure (PKI).

PGP [16] is a PKI implementation, which is based on referral certification. It allows multiple users to sign a public-key, generating multiple signatures to prove the association between a public-key and the real world person. Each user has an authentication key ring that keeps the public keys of those whom the user trusts as introducers. Certificates signed by these public-keys are considered legitimate. Trust in the introducer's signature may be partial in which case a weighted formula with configurable parameters is used to determine whether the key received is believed to be legitimate.

In 1996, R. L. Rivest and B. Lampson proposed the Simple Distributed Security Infrastructure (SDSI) [11]. Later the Simple Public Key Infrastructure workgroup joined force with SDSI to simplify X.509-based public key infrastructures. SPKI/SDSI defines credential-based certificates instead of identity-based certificates in order to authorise a permission requests, grant a capabilities, authorise access etc. SPKI/SDSI certificates bind meaningful attributes of an entity to a public-key. Attributes such as role information, groups and special privileges can be bound to the certificate, therefore providing flexibility to base access control decisions on additional information. In addition, SPKI/SDSI provides the ability to delegate authority by using groups and delegation certificates. A delegation certificate gives the delegatee the rights to sign certain types of statements on behalf of the delegator [11].

The IBM's trust establishment [6] engine provides a useful approach to address access control issues for entities external to the administration domain being accessed. A trust engine has been developed to interpret policy, which is written in the Trust Policy Language (TPL) and specifies to which role an external entity should be assigned to, based on the credentials it presents. A role-based access control [3] model is then used to associate the permissions with the roles. Although a web-based prototype has been developed successfully, TPL is aimed at a networked environment and does not seek to address issues arising from the lack of network connectivity or lack of computational resources. By and large, its approach is complementary to the work presented in this paper. Thompson et al. [14] present an approach similar to TPL which aims to provide certificate-based access control in a widely distributed systems. Instead of using TPL, all

stakeholders generate use-conditions[1] certificates as the criteria that need to be fulfilled by a client in order to use their resources. Clients must show their credentials by presenting sufficient attribute certificates to fulfil all the use-conditions defined in the use-conditions certificates. As in the case of TPL, this approach makes no provisions for lack of network access or computational resources.

The Security Assertion Markup Language (SAML) [5] enables the exchange of authentication, authorisation and profile information between different entities in order to provide interoperability between different security services. SAML defines an XML format for signed assertions, which can be either: attribute assertions, authentication assertions, subject assertions or authorisation assertions. Assertions are issued by SAML authorities which can be security service providers or business organisations such as AOL, AMEX and VISA, etc. For example, an assertion could specify that an individual was authenticated by a particular method at a specific time, or that an application has been granted certain access permissions to a resource under certain conditions. Assertions provide the means of avoiding redundant authentication and access control checks, thereby providing single sign-on functionality across multiple target environments. SAML also defines a request/reply protocol for obtaining assertions from authentication, authorisation or attribute authorities. SAML is in particular aimed at web-services and defines a mapping to SOAP messages exchanged via HTTP [13]. However, as stated in [13], although SAML introduces the concept of exchanging assertions, it does not define any new approaches towards authentication or authorisation.

The Terminodes [9] project, an initiative of the Swiss Federal Institute of Technology aims to build a self-organised mobile ad-hoc network platform. Authentication aspects are mainly built on Pretty Good Privacy (PGP) [16]. A fundamental premise in their work is that all nodes in the ad-hoc network have identical functionality and play an equal role in order to support the self-organised network i.e., it is not desirable to have a subset of the nodes playing specific roles such as monitoring and authentication authority and providing that service to the others. Each node issues its own public-key certificate, stores it locally and subsequently distributes it to other nodes within the ad-hoc network. Furthermore, each node maintains its own local certificate repository that contains a limited set of certificates of peers selected according to the star shortcut hunter algorithm [9]. When two users *u* and *v* communicate with each other and user *u* wants to obtain an authentic public-key of user *v*, both users merge their certificate repositories in order to find an appropriate certificate chain from *u* to *v* in the merged repository [9]. The trust is assumed to be transitive in this self-organised PKI architecture.

4. SYSTEM REQUIREMENTS

In an ad-hoc network deprived of any infrastructure support, every mobile node has to rely on peers to obtain routing, group membership and security information. Mobile nodes have to cooperate with each other in order to secure the network and thus trust among peers plays an important role to ensure system security. The framework presented in this paper advocates the use of credential assertions, obtained from peers, in order to ascertain a user's identity, role, membership in groups or other attributes. These assertions permit authorisation decisions to be made based on credential information that has been attested to by trusted peers.

Furthermore, this approach allows for the full list of user credentials to be consolidated and expressed in XML format, thus providing interoperability with different PKI data structures such as X.509, PGP or SPKI. Herzberg et al. [7, 8] have highlighted the interoperability problems raised by the variety of formats and types of credentials and the resulting difficulties in managing credentials and making authorisation decisions. The consolidation, extraction and formulation of credentials into XML provide the necessary interoperability between different certificates and different certification domains. Credential information is extracted and consolidated in Credential Assertion Statements (CASs), which can be verified by trusted peers and asserted as legitimate at the time of verification. CASs are issued by the users themselves based on the credentials they possess and are signed with their private key. When a user presents its CAS and corresponding Assertion Signature Statements (ASSs), it is necessary to verify that the user actually possesses the private key with which the CAS was signed. This can be achieved via a challenge-response protocol and verification of the signature in the CAS. This property ensures that the CAS is truly genuine from the claiming entity because only he/she could have signed the statement using his/her private key, and nobody else.

5. THE ARCHITECTURE

The architecture of the framework is shown in figure 1. It combines aspects present in PGP [16], XML Signature [1] and SAML [5] and comprises four main application modules, which collectively can ensure the flexible verification of the user's identity and credentials. The four architecture modules are: the XML credential generator, the security assertion module, the verification and validation module and the key management module, which also deals with trust issues. Their functionality is explained in more detail in the following sections.

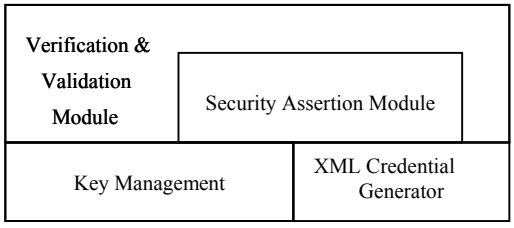

Figure 1. Framework architecture

5.1 Key Management

Every mobile device maintains a key ring that contains a list of trusted public-keys and similarly to PGP, all public-keys in the ring are assigned a trustworthiness level. Additional public keys can be added to the ring provided that their validity has been ascertained beforehand. This would normally include checking the certificate's signature and ensuring that the certificate is

[1] Conditions imposed by stakeholders for access to system resources such as data, instrument, computational and storage capacity.

indeed signed by a trusted certification authority (CA). It is also important to check that the certificate has not expired and has not been revoked by the CA. The public-keys of trusted CAs can also be imported into the key ring. It is the user's responsibility to ensure that only valid public-keys are imported into the key ring and that an appropriate trustworthiness level is assigned to each key. The validity of a credential presented by a user is then determined solely on the basis of the assertions presented regarding that credential and the trustworthiness of the entities signing the assertions. A configurable policy for each user or device determines how partial trustworthiness of assertion issuers is combined in order to estimate the validity of the credential. For example, a certificate will be considered valid if backed by at least a specified minimum number of assertions. The basic philosophy behind this module is to increase the chances of verifying credentials based on information received from peers, thereby avoiding the need for an on-line connection. The key management module ensures that only trusted public-keys i.e., keys in the key ring are considered when checking assertions and assertions made by any entities not listed in the key ring are disregarded.

The trustworthiness associated with public keys is similar to PGP [16]. However, whilst in PGP the trustworthiness associated with a key determines to what extent the user is trusted to act as 'introducer' (i.e., for authentication); in this paper, the trustworthiness associated with the key determines to what extent the user is trusted to sign assertions. Assertions may regard any credentials including authentication credentials and credentials on subject attributes. The trustworthiness levels associated with a key are:

- **Full** – The assertion is believed to be true and thus the credential assertion statement (CAS) verified by this assertion is fully trusted. No further verification against the credentials is needed, for example by contacting the Certification and Attribute Authorities.
- **Partial** - The assertion statement is only partially trusted and thus additional corroboration from other peers is necessary before the CAS can be considered valid.
- **Untrustworthy** - This key is explicitly distrusted for making assertions.
- **Don't know** – There is no expressions of trust made about this public-key, therefore the credential assertion statement verified by this public key should be ignored.

The need for a **partial** trustworthiness level is due to the fact that in certain circumstances, a user may not wish to assign a **full** trustworthiness level to a peer to issue assertions. Note that it would also be possible to attach constraints to the keys in the key ring in order to limit on which type of credentials a user is trusted to issue assertions. For example, a user may be trusted to verify credentials on membership of professional organisations but may not be trusted to verify credentials regarding a user's role in his company. However, this extension would require further work. In the current architecture of the framework, there is no practical difference between *untrustworthy* and *don't know* trust for the purpose of verifying CASs. In both cases the assertions are disregarded. The reason for distinguishing between them is to provide support for future extensions, which would explicitly flag and take into account users who have issued misleading assertions in the past.

Note that an assertion signature statement only certifies that the information presented in the CAS was verified at a particular point in time. It provides no guarantees that the credentials on which it was based have not been revoked since. Policies can be specified on how recent the assertions must be in order to be considered valid. However, ad-hoc networks are typically transient in nature with relatively short lifetime. Thus the time scales between the moment an assertion has been issued and the moment in which it is used are usually relatively short.

5.2 The XML Credential Generator

This module is used to group the user's credentials together in order to create a readable credential assertion statement (CAS). Information expressed in X.509 Certificates and SPKI are extracted and then converted into an XML form to produce a CAS. The CAS must then be signed by the user. However, as in current PKI implementations, the CAS need not be encrypted. Confidentiality of messages in transit would typically be provided through encrypted communications.

The proposed CAS contains X.509 certificate data and attribute certificate data. The CAS must be signed by the user and the signature must conform to XML Signature standard [1]. The <X509Data> tag consists of all elements that can be found in the X.509 Certificate. The same approach is applied to the attribute certificate, which is expressed in <AttributeData> tag. Both tags list the user's identity and credentials as well as other data structures expressed in their respective certificates. These tags can be repeated if the user possesses multiple certificate chains from different trust domains. The following example illustrates a CAS that specifies the user's identity named Soroban and its email address.

```
<?xml version="1.0" encoding="UTF-8"?>
<Credential>
    <X509Data>
        <X509SerialNumber>4</X509SerialNumber>
        <X509IssuerName>EMAILADDRESS=help@doc.ic.ac.uk,
                        CN=Computing Support Group, OU=Computing
                        Dept, O=Imperial College, L=South Kensington,
                        ST=London, C=UK
        </X509IssuerName>
        <X509PublicKey>30 81 9f 30 0d 06 09 … </X509PublicKey>
        <X509ValidNotBefore>2001-11-20 </X509ValidNotBefore>
        <X509ValidTo>2002-11-20</X509ValidTo>
        <X509SubjectName>CN=soroban, OU=Computing Dept,
                         O=Imperial College, L=South Kensington,
                         ST=London, C=UK
        </X509SubjectName>
        <X509SignatureAlgo>MD5withRSA</X509SignatureAlgo>
        <X509Signature>72 76 bb d4 75 67 65 c4 5a e8 … </X509Signature>
        <X509Certificate>30 82 03 f4 30 82 03 5d 4c 1f …</X509Certificate>
    </X509Data>
    <AttributeData>
        <AttributeName>email</AttributeName>
        <AttributeValue>soroban@abc.org</AttributeValue>
        <AttributeValidNotBefore>2001-11-20</AttributeValidNotBefore>
        <AttributeValidTo>2002-11-20</AttributeValidTo>
        <AttributeSubject>30 81 9f 30 0d 06 09 … </AttributeSubject>
        <AttributeIssuer>72 76 03 f4 30 67 …</AttributeIssuer>
        <AttributeSignature>03 f4 30 72 76 bb …</AttributeSignature>
        </AttributeCertificate>75 67 65 03 5d 4c …</AttributeCertificate>
    </AttributeData>
</Credential>
```

An X.509 certificate defines the binding of a public-key to a distinguished name or subject of the certificate. The <X509Certificate> element is used to store the certificate in its original form (ASN.1 DER encoding).

An attribute certificate binds an attribute name to a subject; this attribute can be memberships to a group, roles in an organisation, or any other credentials to allow for a better authorisation decision.

5.3 The Security Assertion Module

The security assertion module's main functionality is to issue assertions to other users after verifying credentials listed in the CAS successfully. The user's credentials can be verified by checking the signatures in the identity and attribute certificates and checking whether the certificates have been revoked. Once the certificates contained in the CAS are successfully verified, an Assertion Signature Statement (ASS) can be issued and distributed to the other members of the ad-hoc network. It is in the users interest to obtain assertions from peers certifying the validity of its CAS. Thus, the CAS can be widely distributed to other users who are encouraged to verify the credentials specified in the CAS and attest to its validity. In ad-hoc networks where at least one of the users has some form of intermittent connectivity, that user can verify credentials and issue assertions (ASSs) to other peers in the group. Credential verification can take place either by contacting the Certification Authorities for authentication certificates or the attribute authorities for attribute certificates. Verifications must be traced to the 'root' CA or attribute authority unless the user already has authentic copies of those authorities' public keys. In addition, the certificate revocation lists for the presented credentials must be checked.

Figure 2 shows how assertion signature statements are issued. Alice sends a self-signed copy of her CAS to Carol. Upon receiving the CAS, Carol first verifies Alice's signature on the CAS; then she verifies the certificates embedded in the CAS (<X509Certificate> and <AttCertificate> tags>) in their original form, and finally she verifies that the data in the CAS (<X509Data> and <AttributeData> tags) matches the information in the certificates. Verifying the certificates may require connecting to the relevant Certification and Attribute Authorities to obtain their public keys and to check the revocation lists. Note that a user may issue assertions (i.e., ASSs) without necessarily needing to cryptographically verify the credentials in the CAS if he possesses *out of band* knowledge on the validity of the credentials. This occurs frequently in daily life where users are prepared to vouch for their friends and colleagues based on the *a priori* knowledge they possess of them. Such flexibility can be abused and facilitates collusion between users who can issue assertions about each other. However, the safeguard is that the recipient of a CAS and corresponding ASSs must have explicitly declared to trust the issuer of the ASS for that assertion to be taken into account.

Subsequently, for each credential verified successfully, Carol can formulate a new Assertion Signature Statement (ASS) signed with her private key and indicating that she has verified the credential successfully. Since there is an ASS for each valid credential, it provides the flexibility to the user to use multiple CASs with different sets of credentials. This avoids the need for a user to reveal all the credentials he/she possesses when accessing an individual resource. For example, academic credentials such as position in a university, membership in research group, professional organisations such as ACM, IEEE etc can be grouped into a CAS, while other credential information relating to financial and on-line banking information can be grouped into another set of CAS. Therefore, the user has the flexibility of choosing the credentials that he/she wishes to reveal and hide other unrelated credentials. An alternative would be for Carol to sign the entire CAS as a whole. The recipient of the credentials could therefore verify the entire CAS in one operation provided he/she trusts Carol. The disadvantage is that if Alice does not want to embed all her credentials in a single CAS, each different CAS would need to be verified by Carol and a different ASS would need to be generated. Once generated and signed, the ASSs are returned back to Alice.

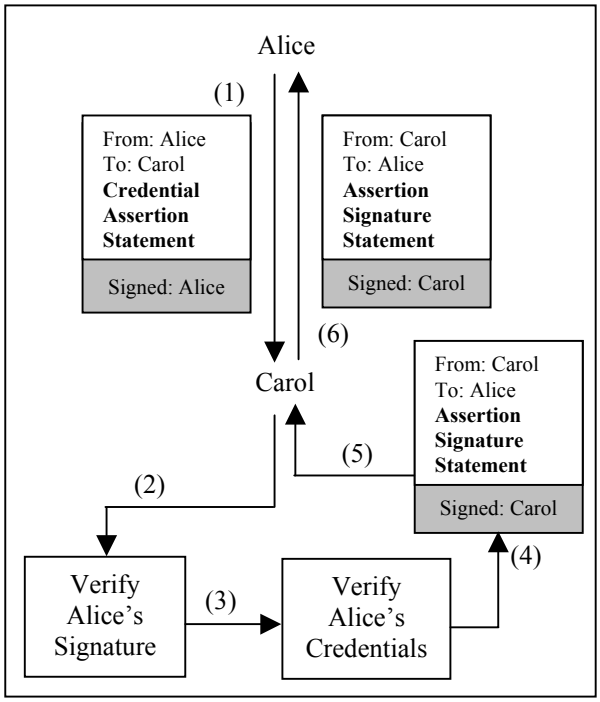

Figure 2. Verifying and signing a credential assertion statement

Typically, an ASS consists of several tags such as signature method, digest method, reference, digest value, signature value, key info and timestamp etc. Fundamentally, the ASS should conform to the XML Signature Specification [1]. The ASS may contain information on where to obtain additional credentials for the signer of the ASS. Note that since all the credentials in a CAS may be confirmed by assertions from the same source and since the recipient has the public key of the signer in his key ring, the number of verifications that need to be performed is greatly reduced. This is a considerable advantage in environments where the computational capabilities and battery resources of the devices are limited.

Fundamentally, the approach proposed in this paper aims to use assertions as a means of reducing the number of checks and

eliminating the need for a continuous connection to the network. However, the use of assertions poses a significant revocation problem. It is difficult to revoke assertions because the issuer will not continue monitoring the revocation status of the certificates for which the ASSs were issued. In addition, in mobile ad-hoc network environments, revocation is very difficult to implement because the network topology is frequently changing and the connectivity of mobile nodes to each other is not guaranteed. To a certain extent, the problem can be mitigated since the recipient of a CAS and the corresponding assertions may choose to either check himself the certificates if it has a network connection or to ignore assertions that have been issued a significant amount of time before they are presented. The latter case is a matter of policy and different policies can be specified with regards to different types of certificates and CASs. In order to enhance the revocation capabilities in mobile ad-hoc network, issuers of ASSs may choose to broadcast to peers the revocation status of certificates for which an ASS has been requested but which have been revoked. Finally it is also possible to broadcast the identities of those entities who frequently issue ASSs for certificates which do not verify. Peers can then choose to revise the trustworthiness level placed in those entities.

5.4 The Verification and Validation Module

The verification and validation module (V&V) is used to determine whether a CAS is authentic and based on authentic credentials. When a user presents his/her CAS together with the corresponding ASSs, the V&V checks the signatures in the ASSs against its key ring to determine whether the assertions can be trusted.

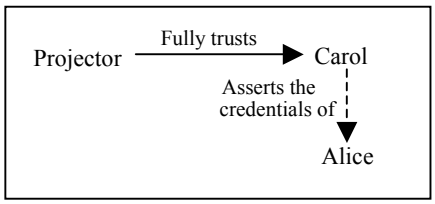

Figure 3. Relationships between the Projector, Carol and Alice

Consider the case where, Alice has been invited to give a seminar on "Mobile Ad-hoc Security" in the Department of Computing and she has to use the data projector for her presentation. The data projector does not have a fixed network infrastructure but can communicate with Alice's laptop via a wireless network e.g., Bluetooth. The projector has no means of authenticating Alice or to ascertain that Alice is an *invited speaker* but its authorisation policy dictates that *invited speakers* are permitted to use it. Carol, who has invited Alice to give the seminar is a full faculty member in the Department and thus trusted by the projector to issue assertions. Carol can therefore issue Alice ASSs regarding her authentication credentials and credentials as an *invited speaker*. When Alice presents her CAS and ASS issued by Carol to the projector, the signatures in the ASS are checked against entries in the projector's key ring. If there is a match indicating that Carol has asserted Alice's credentials and thus Alice's credentials are believed to be valid, authorisation to use the projector is granted. As shown in figure 3, the trust management is similar to PGP. However, it is not based uniquely on identity as in PGP, but caters for any types of credentials. As long as the projector fully trusts Carol, any ASSs issued by Carol imply that Carol has verified the validity of credentials listed in the corresponding CAS. As such, re-verification of credentials presented by Alice when making an authorisation decision is not necessary.

By and large *trust* is not transitive. If A trusts B to issue assertions and B trusts C to issue assertions, nothing suggests that A should believe assertions issued by C. As also underlined by Christianson and Harbison [2], transitivity of trust results in B simply adding trust relations to A without A's consent thus leading to *unintentional transitivity.* In addition, implementing transitive trust requires A to maintain information on or be made aware of B's trust relationships and this recurs until the transitive closure of the trust graph is completed. On the other hand, transitive trust is desirable as it provides better scalability to large systems. As the number of trust relationships increases, the chances of being presented with ASSs from trusted entities also significantly increase.

There are however exceptions to the non-transitivity rule that are worth considering in order to increase the chances of verifying a user's credentials. Consider for example the case shown in figure 4. User *A* fully trusts user *B* and user *B* fully trusts user *C*. *C* issued an ASS, denoted as $ASS_{I\text{-}ACM\text{-}C}$ to user *I* asserting that it has successfully verified *I's* credential that he/she is a member of ACM. Similarly, C may have issued $ASS_{L\text{-}IEEE\text{-}C}$ to L asserting that it has successfully verified L's credential that L is a member of the IEEE. Lets say that *A* is also a member of ACM, but not a member of IEEE. When *I* presents the CAS_I and $ASS_{I\text{-}ACM\text{-}C}$ to *A*, *A* has a copy of ACM's public-key, thus allowing *A* to check the signature of the credentials in CAS_I. However, *A* may not be able to verify whether the credential has been revoked. If A trusts B and is aware that B trusts C, it may be reasonable to argue that A should trust $ASS_{I\text{-}ACM\text{-}C}$ and hence believe that information which it knows was true at a point in the past was still true when C has verified it. More generally stated, the argument is that transitive trust may be considered when the trust is used only to corroborate information, which can be partially verified. However, the fact that A has trusted $ASS_{I\text{-}ACM\text{-}C}$ does not imply that all C's assertions will be trusted henceforth. For example, $ASS_{L\text{-}IEEE\text{-}C}$ may not be trusted if A is not a member of the IEEE and has no means of at least partially verifying information in the CAS matching the issued assertions.

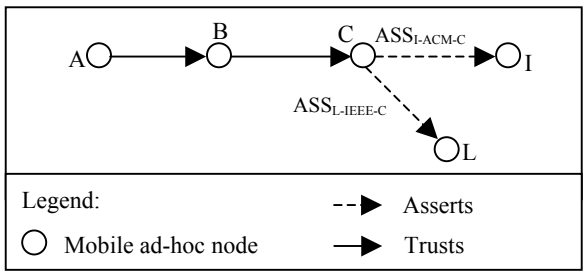

Figure 4. Permissible transitive trust relationships

There are other scenarios in which selective transitive trust may be justified. In an ad-hoc network, participants are not necessarily

equally well equipped. Some of the entities may have an administrative role managing the membership of the ad-hoc network, providing security services to peers or being in other ways in a "position of trust". These entities may have trust relationships with other entities, which have a similar position in other nearby ad-hoc networks. In order to allow members of the different ad-hoc networks to communicate, one can either specify a whole additional set of trust relationships or can choose to trust all external entities trusted by the ad-hoc networks administrator.

6. DISCUSSION

The Security Assertion Markup Language (SAML) [5] uses the concept of trust assertions in e-commerce transactions over the Internet and is particularly targeted at web-services. It also advocates the use of XML as an interoperable means of encoding and exchanging security information. The approach presented in this paper introduces two novel aspects. First, it introduces the concept that credentials can be consolidated in order to provide integration of legacy PKI systems and limit the number of cryptographic verifications needed and second, it introduces assertions about the successful verification of credentials. These assertions can be issued by any entity in an ad-hoc network, whereas SAML assertions are typically issued by either authentication services, attribute authorities or policy decision points.

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.

Several trade-offs exist in the choice of the information exchanged. In particular, the granularity of the CASs and ASSs needs to be chosen carefully. Possible solutions range from generating a single CAS, which aggregates all the credentials that a user possesses to using individual credentials without aggregating them into CASs. Using a single CAS has considerable privacy implications since users would typically not want to reveal all their credentials regardless of the context in which they are used. We have considered here the case where there are a limited number of CASs, which groups those credentials that need to be used for the main categories of user activities e.g., work related activities within the company, external work activities, professional memberships etc. Similarly, ASSs can be issued for the CAS as a whole, for individual credentials or both. Issuing ASSs only for the CAS as a whole is not practical because the trusted entity must successfully verify all the credentials listed in the CAS before issuing the ASS. Should it not be able to verify one of them, either because the corresponding authority cannot be reached or because it does not possess the necessary keys, the ASS cannot be issued. Furthermore, each time the user needs to generate a new CAS, it must obtain new ASSs even if the CAS groups permissions that have already been verified. On the other hand, issuing ASSs for the individual credentials held in a CAS implies that the recipient of a CAS must verify the signatures and correspondence between each credential and each ASS.

The transitivity of trust relations has been often discussed [2, 9]. Whilst a compelling argument can be made in favour of trust transitivity for scalability reasons, an equally compelling argument can be made against it for security reasons. We have considered here that by and large trust is not transitive but that selective transitivity may be permitted under well-defined constraints. In particular we have presented cases where the assertions accepted through transitive trust are used only to corroborate partially verified information and where transitive trust is placed only in entities that fulfil particular privileged roles within the ad-hoc network.

Revocation problems are particularly difficult in ad-hoc networks because the topology of the network may change and entities may suddenly become unreachable. Using assertions further exacerbates the problem as authorisation decisions can be made on the basis of credentials and assertions without recurring to further verifications and thus bypassing traditional revocation mechanisms such as certificate revocation lists. Although time-based constraints can be used to mitigate the problem, they are not always sufficient in order to provide satisfactory results. However, in ad-hoc networks based on wireless communications, all messages are essentially broadcasted. Thus, revocation information can be quickly disseminated across the network, although the recipients would need to maintain some history of revoked certificates and assertions. Adequate means of providing revocation and relating this to the timing constraints under which assertions are accepted need to be further investigated.

7. CONCLUSION

In this paper, we have presented an initial approach towards providing flexible credential verification in mobile ad-hoc networks based on assertions issued by trusted peers. This approach was chosen because in the absence of on-line connectivity to a network infrastructure, and in particular to the issuers of certificate and attribute authorities, verification of user credentials must rely upon the information provided by the peers. We further consider that participants in the ad-hoc network are not complete strangers in the sense that some degree of trust relationships exists or can be established between them. This approach is also particularly suited when some degree of intermittent connectivity exists and draws significant advantages from *out of band* knowledge between peers.

We have briefly presented an overall architecture of the main functional modules necessary to implement this framework and described their role within the framework.

The use of Credential Assertion Statements (CAS) encoded in XML not only provides interoperability with legacy PKI systems but also helps to limit the number of checks (and thus

computational resources used) when verifying credentials. Issuing assertions on individual credentials rather than CASs provides a more flexible approach that caters for user privacy considerations and limits the unnecessary disclosure of credentials.

We have discussed the use of trust relationships and have highlighted specific and constrained cases in which transitivity of trust may be considered.

Substantial work remains to be done towards the use and specification of policies which limit the conditions under which trust is granted and which limit the use and applicability of assertions. Finally, work remains to be done towards the implementation of the framework.

8. ACKNOWLEDGEMENT

We gratefully acknowledge financial support from the EPSRC under grant GR/L 96103/01.

9. REFERENCES

[1] Bartel, M., Boyer, J., Fox, B., LaMacchia, B., and Simon, E. XML-Signature Syntax and Processing. W3C Recommendation, The Internet Society & W3C, 12 Febuary 2002.

[2] Christianson B., and Harbison, W.S. Why Isn't Trust Transitive?. In Security Protocols International Workshop, University of Cambridge, 1996.

[3] Ferraiolo, D., and Kuhn, R., Role-based Access Controls. In 15th National Computer Security Conference. NIST, October 1992, 554-563.

[4] Frodigh, M., Johansson, P. and Larsson, P. Wireless Ad-hoc Networking – The Art of Networking Without a Network. Ericsson Review, 4, 2000.

[5] Hallam-Baker, P., and Maler, E. (Eds.). Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML). Available online at http://www.oasis-open.org/committees/security/docs, 10 January 2002.

[6] 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.

[7] Herzberg, A., and Mass, Y. Relying Party Credential Framework. In Topics in Crytology – CT RSA 2001, The Cryptographer's Track at RSA Conference, San Francisco, CA, 8 – 12 April 2001.

[8] Herzberg, A., and Mass, Y. Relying Party Credential Framework. In Special Issue on Security Aspects in E-Commerce of the Electronic Commerce Research Journal, 2002.

[9] Hubaux, J.P., Buttyan, L., and Capkun, S. The Quest for Security in Mobile Ad Hoc Networks. In Proceedings of ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHOC), Long Beach, CA, 4-5 October, 2001.

[10] Linn, J., Trusts Model and Management in Public Key Infrastructures. RSA Labs, 6 Nov 2000.

[11] Rivest, R. L., and Lampson, B. SDSI – A Simple Distributed Security Infrastructure. April 1996.

[12] Satyanarayanan M. Pervasive Computing: Vision and Challenges. IEEE Personal Communications 8(4), August 2001, 10 - 17.

[13] Sriganesh, R.P. Implementing Single-sign On in Java Technology-based Web Services. In Sun's 2002 Worldwide Java Developer Conference, San Francisco, CA, 25 – 29 March 2002.

[14] Thompson, M., Johnston, W., Mudumbai, S., Hoo, G., Jackson, K., and Essiari, A. Certificate-Based Access Control For Widely Distributed Resources. In Proceedings of the 8th USENIX Security Symposium, August 23-26, 1999.

[15] Weiser, M. The Computer for the 21st Century. Scientific American, September 1991.

[16] Zimmermann, P. The Official PGP User's Guide. MIT Press, Cambridge, MA, 1995.