Federated Identity Management in a Tactical Multi-Domain Network

Anders Fongen
Norwegian Defence Research Establishment
Norway
anders.fongen@ffi.no

***Abstract*—Identity Management maintains information regarding actors of an Information System, like users, equipment and services. One important service is to disseminate and validate *credentials* for the purpose of *authentication* and *access control*. Within the context of military tactical communication network the identity management services should, due to the disadvantaged nature of these networks, minimize their network demand and connectivity requirements. Security protocols for tactical network should be efficient, prudent and be based on well justified use cases. The contribution of this paper is the rationale and the prototype of an identity management system designed with these properties in mind, including services for authentication and access control. The discussion will suggest a set of architectural patterns for the development and deployment of an identity management system, as well as justifications for the simplified protocol operations.**

***Keywords*-Identity management, Disadvantaged networks, Authentication**

I. Introduction

This article is an extension of a previously published conference paper [1].

Current systems for Identity Management (IdM) are constructed with an emphasis on "Single Sign On" (SSO) service and authentication of clients. *Federations* of IdM systems are supposed to recognize clients from other domains, but are often found to require replication of user registries or the creation of a separate "federated" user registry. They are observed to require identification and enrollment separate from any existing public key infrastructure (PKI) and do not offer authentication of services. Existing IdMs are also seen to rely on invocation of authentication servers for each authentication process, resulting in larger bandwidth and connectivity requirements. [2]

These properties are inadequate and inefficient in the perspective of a dynamic military network of large scale. A military network will extend its operation across wired and wireless links, and observe a wide range of data rates and connectivity conditions. One network will connect to other networks belonging to other parties of a coalition in order to offer well managed services without loss of autonomy.

A. Federablity and Ubiquity

The ability for an IdM to enter into federations with other IdM domains in a scalable, controllable and manageable manner is called its *Federability*. The *Ubiquity* of an IdM indicates its ability to operate on a wide range of equipment and network environments. The properties of an IdM which decide its ubiquity and federability will be identified in the course of this article.

Associated with identity management is the process of *Authentication*. Authentication relies on the provision of identity credentials and offers "establishment of identity". Authentication protocols can be designed for disadvantaged environments with small demand for network resources based on a realistic threat analysis: some threats are too far-fetched to justify expensive protocol details.

It is the purpose of this article to offer an analysis on how an identity management system may be deployed and operated in a tactical, disadvantaged network. The article will also provide a description of a prototype IdM for disadvantaged networks, and the rationale of its design.

The remainder of the article is organized as follows: Section II provides a short introduction to Identity Management Systems, and Section III discusses necessary properties for a successful IdM. Section IV briefly discusses limitations of existing IdM systems, and Section V introduces the prototypical Gismo IdM which is used for experimentation and a main object for this article. Section VI discusses the properties of authentication protocols used in disadvantaged networks, while Section VII presents the experimental protocols in detail. Section VIII presents the challenges when porting the Gismo IdM to the Android platform. Finally, Sections IX and X presents experimental plans and some conclusive remarks.

II. Motivational Background

Identity Management (IdM) are collections of services and procedures for maintaining subject information (key pair, roles) and to issue credentials for the purpose of authentication, message protection and access control. From the client perspective, the credentials issued by the IdM services enable it to access many services inside a community under the protection of mutual authentication and encryption. From the server perspective, IdM enables it to offer its credentials to clients in order to provide service authentication and to control access to its resources.

A. *Federated Identity Management*

Several federated IdM schemes have been developed, some of which offer single sign on (SSO) for web clients [3], [4], [5]. The SSO protocols exploits the redirection mechanism of HTTP in combination with cookies and POST-data so that an Identity Provider (IdP) can authenticate the client once and then repeatedly issue credentials for use within the federation. This arrangement requires IdP invocation for each "login" operation, and does not offer mutual authentication, i.e., service authentication. [2].

In the situation where the client is an application program (rather than a web browser), there are more opportunities for the client to take actively part in the protocol operations, e.g., by checking service credentials, contacting the IdP for the retrieval of own credentials, caching those credentials etc. The research efforts presented in this paper assume that the clients are able to run custom built programs.

The usual meaning of the word "federated" is that several servers share their trust in a common IdP for subject management and authentication. It does not necessarily imply any trust relationship between independent IdPs so that they can authenticate each others' clients. For the following discussion, we will call the group of clients and services which put their trust in the same IdP as a *community of interest* (COI). A trust relation between independent IdPs is called a *cross-COI relation*.

B. *Mobile and Federated IdM requirements*

An essential property of an IdM is its ability to integrate with other components for management of personnel and equipment. The list below contains other necessary properties, some of which are to be explained later in the article.

- An IdM should be able to use resources from the existing PKI (keys, certificates, revocation info) and offer its services to different platforms, with different presentation syntax and for different use cases.
- An IdM instance should be able to form trust relations with other IdM instances in order to accommodate guests and roaming clients.
- An IdM must provide support for role/attribute based access control
- An IdM must support protocol operations for mutual authentication.

For IdM used in mobile systems, there are requirements related to the resource constraints found in these systems:

- The protocol operations of an IdM must use small PDU sizes and allow the use of caches to reduce the system's connectivity requirements.

C. *The relation between IdM and Access Control*

Services can enforce access control on the basis of the *identity* of an authenticated client, or based on *roles* or *attributes* associated with the client. For the purpose of the accommodation of roaming users, it is absolutely necessary to make access control decisions based on roles/attributes, not identity. "Identity based" access control requires that all roaming clients are registered into the visited IdM, which is an unscalable solution.

The principles of *Role/Attribute Based Access Control* (RBAC/ABAC) are well investigated [6]. The names and meaning of the roles/attributes that are used to make access decisions must be coordinated as a part of an IdM trust relationship. For this reason, the number of roles/attributes used for access control needs to be kept low.

It is the responsibility of an IdM to manage the roles/attributes of a subject, some of which may enter into access control decisions, others may be used by the service for configuration purposes etc. The presence of subject attributes is the main functional difference between IdM credentials and X.509 public key certificates.

III. Candidate Design Patterns

One of the contributions of this article is a set of proposed design patterns for the construction of a scalable IdM with loose coupling between management domains. The patterns are:

A. *Use Existing PKI*

In most organizations, there are formal procedures related to employee and inventory information. Quality of that information is crucial in order to prevent/detect fraud and theft. Some organization have also implemented a Public Key Infrastructure (PKI) (or are planning to do so) for the purpose of public key management. A PKI in operation will be the result of a long planning process, complicated software deployment and configuration, and the development of several new managerial interfaces between the HR and IT departments. An operational PKI represents a significant investment that should be retained when an IdM is being developed.

B. *Federate Domains for Guest Access*

Back then, there was the idea of a PKI which could operate on a very large scale, e.g., for every citizen of a nation, and serve a large number of applications. Today, a national PKI is believed to provide keys only for limited communication between citizens and the public sector. Other PKIs will provide keys for banks, others for Internet shopping and again, others for professional communication.

IdMs have the potential to bridge the gaps between different *domains* of key administration, meaning that they can manage trust relations between domains in an articulated manner. Domain federations allow subjects to bring their credentials across domains for controlled access and trust.

C. Roles Matter, not Identity

The rule in "traditional" user management in standalone computers has been never to grant privileges directly to subjects. Subjects should be assigned to *groups*, and groups given access rights. *Role-based* or *attribute-based* access control [6] is built on this idea, which is several decades old and well proven.

This separation makes lots of sense in a distributed environment. It means that only the IdM service needs to maintain actual identities, whereas the providers of business services maintain the mapping between access rights and *roles* or *attributes*.

In a domain federation, this separation is crucial. Although some IdM systems for domain federations provide mapping between user names on different systems (hopefully for legacy reasons only), the only scalable approach is to allow the users to be represented by a set of roles/attributes.

D. Domains are Autonomous

All domains of identity management wish to be autonomous. They establish identification procedures based on their own business and security policies, according to national legislation and the ethics of their profession. They will determine what services will be made available to residents and guests of the domain. They decide by themselves the access rights that are associated with subject attributes. *Domain federations should not impose any federated authorities.*

Another matter of domain autonomy is role (or attribute) *privacy*. The attributes associated with a subject may be of sensitive nature, since they may reveal information about the subject's authority. Consequently, the domain must be in control of how attributes are exposed inside and outside the domain [7].

E. Avoid belt-and-suspenders protocols

The network cost associated with the operation of a PKI is substantial, and inhibits this operation in parts of the network where the bandwidth is narrow or the connectivity is episodic [8]. Networks with such conditions include wireless mobile networks (MANET) and military tactical networks. Wireless networks are more exposed to intrusion attacks than a wired network. Ironically, the parts of the network that really need the protection that a PKI could offer, are thus the parts least suited to use it!

Consequently, the networking protocols (and the security policies they result from) must ensure that the network resource requirements do not exceed the expected performance of the technology in place. This may require a closer inspection of the risk estimate, and some belt-and-suspenders security requirements may have to be relieved.

F. Trust has a lifetime

This pattern is firmly related to the previous paragraph. It is a matter of reducing the network traffic through a "trust has a lifetime" decision. For example, a validated public key is believed to be valid for some time, and will not need to be revalidated during this period. This principle is well established through the distribution interval of certificate revocation lists (CRLs).

This principle reduces the number of necessary operations from both the client and the server to the IdP services. They do not longer need to receive credentials and validation information for each business operations, since this information can be cached and re-used for a while.

G. Limit the unconditional trust

The last design pattern is related to the number of *trust anchors*. A trust anchor is a subject whose signature is unconditionally trusted. All trust relationships are derived from a trust anchor through a chain of signatures. The security of the entire system collapses if a trust anchor gets compromised. Therefore, the number of trust anchors should be low for the sake of system security and robustness [9].

IV. Existing IdM Architectures

The proposed design is related to the SAML 2.0 architecture for federated identity management [10] and the WS-Security [11] and WS-Trust standards [12], but this model aims to provide better answers to the challenges of mobile and tactical environments.

Based on a survey of existing models for federated identity management like Liberty Alliance [5], Shibboleth [3], and OpenID [4], it is an observation that they are *not* well suited for low-bandwidth, mobile or disadvantaged networks for the following reasons:

- They require much connectivity, in the sense that every new connection with a service involves operations on the identity provision servers.
- They require a coordinated replication of user registries, so that an excessive amount of work is needed to maintain user information in a highly dynamic network.

The same survey also indicates that these approaches to identity federation create rather strong coupling between the security domains; they either require mapping between local user identities, or mapping between local and federated identities. Both approaches could be replaced by an RBAC (role based access control) [6] arrangement that removes the need for replicated user identities in order to weaken the coupling between the domains.

Please observe that the term "federated" in this article refers to federation of servers from different communities with independent security requirements. The term "federation" as used in the related literature may refer to a group of servers in the same domain, in which case coordination is a much simpler problem.

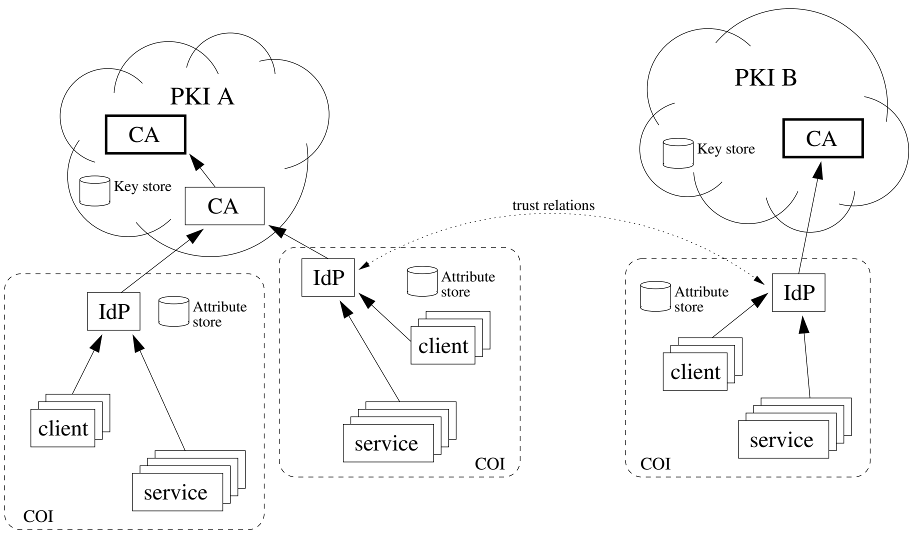

Figure 1. The functional components of the Gismo IdM system.

V. The Gismo Architecture

Following the guidelines given in Section III, an IdM prototype has been built for the purpose of experimentation [13]. The prototype has been implemented in Java for operation in a service oriented environment. The protocol data units have been given two different syntax representations:

1) For the operation in a Web Services environment the PDUs are coded in XML-based syntax like SOAP, WSSec, SAML etc.
2) In environments with poor support for WS-standards (like the Android platform), the PDUs have been coded as serialized Java objects, from now on simply called "POJO" (Plain Old Java Objects).

The presentation layer diversity is discussed in Section VIII.

The functional components of the Gismo IdM and their relations are shown in Figure 1. Observe that the IdP serves one single COI, and the trust relations are formed between COIs, not domains. Key management is handled by the PKI whereas the attribute management is done by the IdPs on the COI level. ("Gismo" is the acronym for the Norwegian expression "Fundamental IT security for mobile operations".)

A. The Domain

In the context of this project, the term "Domain" means a population of services and subjects with the following set of properties:

- Members (services and subjects) belong to one domain only
- All members of a domain share the same *Certificate Authority* (CA) and *trust anchor*.

B. Community of Interest

Inside a domain, there are one or more *Communities of Interest* (COI). For each COI, there is one *Identity Provider* (IdP). Members of a COI are subjects (either client or server), and they can be member of several COIs (inside the same domain). Two subjects can have authenticated communication (client-server or message exchange) if they are members of the same COI, or if they are members of two COIs with a *trust relationship*.

C. The Identity Statement

The Identity Statement (IS) is similar to a public key certificate in the sense that it attests a binding between a public key and the identity information of the "owner" of the private key. In addition, the IS contains a set of roles/attributes associated with the represented identity.

The identity statements are issued and signed by the identity provider, and are therefore valid only inside the COI served by that IdP.

There is *no revocation checking* associated with identity statements. An IS is therefore meant to be short-lived, i.e., expire after a duration comparable to the issue interval of certificate revocation lists.

D. The Identity Provider

The Identity Provider (IdP) is a CA-like service which issues identity statements for the members of the COI. Upon requests from subjects, their IS are issued and returned to the clients for use in different authentication procedures.

Another important task for an IdP is to provide identity statements for *guests*. If a subject sends an IS issued by an IdP with which there exists a trust relationship, a *guest IS* is issued. The guest IS contains the same information as the original IS, except that attributes may have been added or removed. It also bears a new signature, generated by this IdP.

Please observe on Figure 1 that even there exist PKIs and CAs which issue X.509 certificates, they are only visible to the IdP. The COI members only relate to the IdP services and the identity statement. The PKI services may therefore be replaced with different technology without affecting the COI members.

E. Cross-COI Operation

An important property of an IdM architecture is the ability to offer services to members of a different organization in a well controlled manner. This property is an important part of the Gismo IdM and is based on *guest IS* to indicate the approval of a guest identity, and the *cross-COI IS* to indicate the trust relationship between to COIs. Together with an RBAC/ABAC based access control framework, guest may be given access under a fine-grained policy.

Trust relationships between two COIs are expressed by a pair of IS where they attest each other's public keys and identities. These *cross-COI IS* link the signature on an IS from a remote COI to the IdP of the local COI, and conveys the delegation of trust from the local IdP to the remote IdP.

F. Proof of validity

Members of a COI trust the CA of the domain, i.e., the CA is their *trust anchor*. They also need to trust the IdP, since the identity statements bear its signature. The IdP may be declared as a trust anchor, too, but there are good reasons (mentioned in Section III-G) why the number of trust anchors should be kept to a minimum.

The trust in the IdP could be derived from the CA through a PKI-style *validation* of the IdP's certificate, which is not a desirable solution since it generates much network traffic and breaks the encapsulation of the underlying PKI.

Rather, it is a preferred solution that the IdP is the only central service that the members know about, and that the IdP itself can provide a "proof of validity" for its key and certificate. Given this proof, any member can conclude that the key of the IdP is authentic and not revoked at the moment.

The proof of validity may have several forms, depending on whether the trust anchor CA is the direct or indirect issuer of the IdP's certificate. It should contain all certificates from (and including) the IdP's certificate and up to (not including) the trust anchor (normally the root CA). It should also provide proof that none of the certificates on this list are revoked at the moment.

The proof of non-revocation cannot be a revocation list, since it is not possible to provide positive information in it, only negative. One cannot assume that a key is valid only because it is not listed as revoked. What is needed is a positive revocation status (meaning not revoked), which is the output of a *validation server*, e.g., one that is based on the SCVP or OCSP protocols. These responses must be signed with a key that is attested by the trust anchor through a signature chain.

The CA could issue an SCVP response on a regular basis which the IdP could hand out on demand, but that would require a custom built CA and a violation to the rule in Section III-A. Standard PKI services must be used, which would likely be the signed and timestamped output from certificate status providers (using OCSP) if available. If the trust anchor refuses to issue revocation status in any other form than through CRLs then one is out of luck and needs to declare the IdP as the trust anchor for the members of the COI.

In essence, the separation of the IdP from the trust anchor is a matter of reducing the number of trust anchors, as well as a matter of *trust anchor protection*. An IdP is reachable for everyone, and with a trust anchor key inside it becomes an attractive attack target.

G. Attribute Protection

Subject attributes in an IS (elsewhere also called *roles*) are name/value pairs which can describe any aspect of the subject. It can be used to store the subject's native language in order to improve the user interface of a service etc., or describe the subject's authorizations for access control support.

Attributes may contain sensitive information which should be adequately protected. The ultimate protection is for the IdP to issue an IS for the purpose of one particular service, encrypted with the public key of this service. On the other hand, that arrangement makes the IS non-cacheable and requires frequent connection to the IdP, effectively making it into a single point of failure.

The Gismo IdM approach is taking a middle road. An IS issued for use in a COI should be cacheable and be used for all services and conversations withing the COI until the IS expires. When an IdP receives an IS from a guest who is requesting a guest IS, only attributes marked for export are copied into the guest IS, the other are removed. Since there exists a trust relationship between these two IdPs it is reasonable to trust a "foreign" IdP to do this honestly and correctly. It is also reasonable to allow services and subjects in the same COI to share attribute knowledge, since the COI membership with shared goals and shared responsibility also implies a level of trust (and since they might obtain this information anyway through listening on the shared data links).

In those cases where the intra-COI traffic need to be protected from other activities on the same network links, a Virtual Private Network arrangement should be employed.

Since the use case of our prototype is related to military applications, the value of privacy protection has not been highly regarded.

VI. The Authentication Protocols

Several authentication protocols have been devised under the Gismo IdM project, with the goal to reduce the number of protocol round trips and to explore the relation between network cost and risk.

The "proof of possession" principle is in common use in authentication protocols. The requester proves that it is in possession of a secret (that only this subject knows) in order to prove its identity. The secret can be a private key, and the proof of possession can be implemented in at least two ways:

- The requester can sign the request message with its private key
- The responder can encrypt the response with the public key of the requester.

Although these options both offers authentication, only the first also offers protection of message integrity. Without this protection, an attacker may alter the content of the request without detection.

Another attack scenario is the *replay attack* where the attacker gathers messages in transit and re-injects them into the network at a later stage. For these attacks to be detected, messages can be timestamped (so excessively old messages are discarded) and duplicates should be recognized within the allowed time frame.

Protection against replay attack in authentication protocols is quite costly, since it requires the service to remember previous requests (identified by e.g., nonces) for the maximum allowed clock skew period, *also during a crash* (i.e., across "incarnations"). This is a hard problem, since lightweight service platforms (like embedded systems) may not be able to offer the transactional stable storage which is needed to implement this mechanism.

A. Stateless services do not require replay protection

Under the conditions that the service is stateless, i.e., a request is not altering the state of the system (e.g., a lookup service), replay protection is not needed, provided that only the intended client can read the reply. The authentication protocol may under such circumstances simply encrypt the reply with the public key of the client to achieve this effect. For protection of message integrity, the client should add a signature to the request message.

B. Authentication "as we go"

Another matter is the number of protocol round trips. During a separate authentication phase, client and service can mutually authenticate themselves before the actual service call is made. A more effective approach is to piggyback the client authentication on the service request, and the service authentication on the response, as shown in Figure 4. This reduces the number of round trips, but the risk remains that a mere request to a fraudulent service may compromise sensitive information. This is, in the author's opinion, a far-fetched risk: An attacker who is able to stage such an advanced attack would benefit more from simple eavesdropping than a "hit and run" tactic. A fraudulent service which is not able to authenticate itself would trigger an intrusion alarm and a subsequent hunt for the intruder.

Under other conditions, e.g., a protected and authenticated conversation, a more traditional approach would still be the best choice where mutual authentication and session key exchange takes place before the information flow starts.

The replay protection of request messages is difficult because client may approach the service for the first time, when no shared state can exist common to both. For the response message the protection is easier to obtains. A nonce in the request message may be copied to the reply message and protected under the signature of the service. The request message establishes a shared state which simplifies the subsequent replay protection.

C. Clock skew elimination

For the purpose of reducing the period during which messages need to be remembered for duplicate detection may be greatly reduced through the presence of a common clock source and an associated synchronization protocol. While in a tactical network a separate synchronization protocol like NTP consumes costly bandwidth, the choice of this experiment has been to piggyback clock information in the messages from the IdP.

The experimental clock protocol is illustrated in Figure 2 and its operations are performed as follows:

1) The IdP includes with every issued identity statement a clock value, which is the number of milliseconds since a chosen t_0.
2) Upon reception of the IS, the subject starts counting milliseconds starting with the clock value included in the IS.
3) When sending a request message, the client includes the current millisecond counter value in the request, and protects this value with its signature.
4) When receiving a message, the service compares the counter value in the request with its own counter value (initiated with the counter value of its IS). The message is rejected if the difference is outside an allowed range.

Figure 2 shows how an IS issue at time t_x is marked with the timer value t_x, but received at $t_x + \Delta t_x$. The subject's counter at value at time t_z is therefore $t_x + t_z - (t_x + \Delta t_x)$, so this value is included in the service invocation which takes place at t_z. The IS issued for the service is marked with t_y, but received at $t_y + \Delta t_y$, at which point the service starts counting from the value t_y. The service receives the request at $t_z + \Delta t_z$, at which time its counter value is $t_y + t_z + \Delta t_z - (t_y + \Delta t_y)$. The difference between this value and the value included in the request is $\Delta t_z + \Delta t_x - \Delta t_y$. The network delays during IS issue cancel each other out and reduce the variance

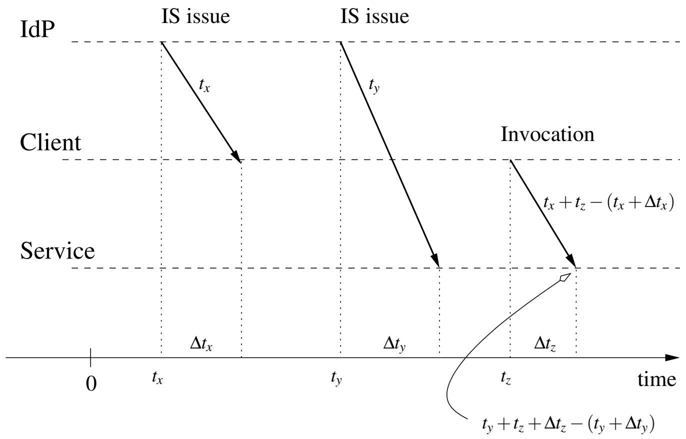

Figure 2. The clock synchronization mechanism included in the IdP and service invocation protocols.

of the expression. The clock drift during the IS issue interval is regarded as negligible.

The request message should be discarded if the value difference exceeds a threshold interval. The treshold interval should be chosen to give a low probability for false positives (due to network delay) yet maintaining a fair amount of protection.

Even without detection of duplicate messages, replays are now given a very short time window to succeed. It is a claim by the author, supported by observations in the cyber defense community, that replay attacks that must take place within one second (example value) have very limited application. A successful replay attack is likely to be the result of a period of traffic eavesdropping and analysis, followed by injection of carefully selected messages, all in a manner that minimizes the probability of detection. A replay attack for the purpose of *Denial of Service* is highly unlikely since the advanced position and capabilities of the attacker will be immediately detected and eliminated.

It is therefore proposed that with a clock synchronization scheme like the one outlined above, detection of duplicates is not strictly necessary. If duplicate detection is required, it is much simpler to implement since the time window is much smaller. If the server crashes, it can simply withhold its service for the duration of the time window during restart. This details relieves the server from the need to recognize duplicates across service incarnations, which was identified as a costly operation earlier in this section.

VII. Protocol and Data Structure Details

At this point the design principles and the main functional components of the Gismo IdM have been explained, and the article will commence with a description of the data structures and protocols in greater detail.

A. The Identity Statement

As previously described in Section V-C, the authentication mechanisms relies heavily on the data structures called *Identity Statement* (IS). Formally, the identity statement of principal x signed by the IdP of COI a is denoted $(Id_x)_a$ and has this structure:

$(Id_x)_a = Name_x + PublicKey_x + Attributes_x + TimeCounter + Signature_a$

$Attributes_x$ denotes a set of name-value pairs which describes the roles etc. of the subject. It may be used for access control purposes. $Signature_a$ indicates that the entire statement is signed by the IdP of COI a. The IdP of COI a will from now on be denoted IdP_a.

In the proposed system, the identity statement is formatted according to one of the following methods:

- The SAML 2.0 syntax requirements, which means that it is coded in XML. The SAML assertion is used in a so-called "Holder of Key" mode.
- As serialized Java objects.

A discussion on the existence of parallel syntax representations will be given in Section VIII.

B. Identity Statement Issuance

The discussion in Section V-G identified the need to protect subject attributes outside the Community of Interest (COI), which means that only members of a COI should be allowed to ask the Identity Provider (IdP) for an IS regarding a COI member.

There is no easy way to distinguish a member from a non-member (without a costly authentication phase). The design choice has therefore been to issue an IS only to the subject itself. Prior to the issue of an IS, the subject need to have its private key loaded, through which it can prove its identity to the IdP. The proof can be implemented as:

- an SSL-based authentication phase prior to the IdP invocation
- a signature on the IdP request (susceptible to replay attacks)
- encryption of the IdP response with the subject's public key.

In Figure 3, which shows the IS issue protocol, the client authentication is not shown, but regarded as a protocol implementation detail.

A part of the IdP service semantics is that the subject's key pair is *validated* before the IS is issued. If the key pair is generated by a PKI (as suggested in Section III-A) the IdP should use the available PKI-based validation mechanisms for this purpose, and deny the issue request if the key is invalidated or revoked.

C. Issuance of Guest Identity Statement

The IdP is responsible for the issuance of guest identity statements as explained in Section V-D. Presented with

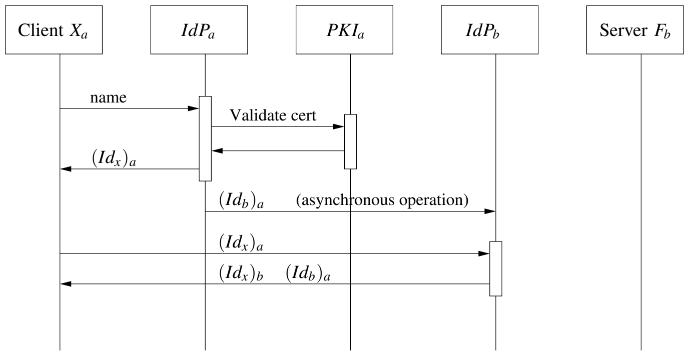

Figure 3. The identity statement issuing protocol. In this case a guest IS is being issued in two steps.

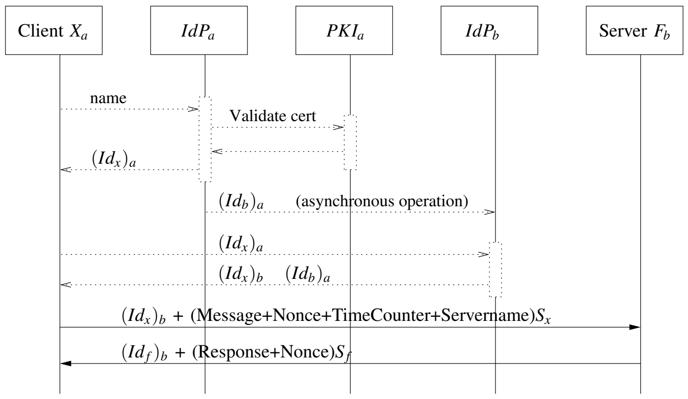
Client X_a
IdP_a
PKI_a
IdP_b
Server F_b
name
Validate cert
$(Id_x)_a$
$(Id_b)_a$ (asynchronous operation)
$(Id_x)_a$
$(Id_x)_b$ $(Id_b)_a$
$(Id_x)_b$ + (Message+Nonce+TimeCounter+Servername)S_x
$(Id_f)_b$ + (Response+Nonce)S_f

Figure 4. The authentication protocol for the stateful service.

$(Id_x)_a$, the IdP_b (IdP of COI b) can issue the identity statement $(Id_x)_b$ provided that there exists a trust relationship between COI b and a expressed by an identity statement issued by IdP_b with IdP_a as the *subject*. This is called a cross-COI IS and expressed as $(Id_a)_b$. With the guest IS $(Id_x)_b$, the subject x which is a member of COI a, can authenticate itself to members (e.g., services) of COI b.

For two-way authentication in a guest COI, e.g., for the client from COI a to trust the signed response from a member of COI b, the reverse cross-COI IS is needed, termed $(Id_b)_a$, to link the signature key to the client's trust anchor. Therefore, $(Id_b)_a$ is included in the response of the guest IS issuance. $(Id_b)_a$ is issued to IdP_b by IdP_a (as a normal IS issue) and stored by IdP_b for the purpose of guest IS issuance.

TABLE I
ABBREVIATIONS USED IN THE FIGURES

Client X_a	Client X of COI a
IdP_a	Identity provider of COI a
PKI_a	Validation services in domain a
Server F_b	Server F in COI b
$(Id_x)_a$	Identity statement for identity x, issued by IdP_a
$(msg)S_x$	Message *msg* signed with private key of x
$(msg)E_x$	Message *msg* encrypted with public key of x

Figure 3 illustrates the guest IS issuing protocol as a two stage process involving two IdPs. Key validation takes place only in the first stage. The optional proof of validity (Section V-F) is assumed to have been issued at an earlier occasion.

D. The Authentication Protocol

Section VI provides a discussion on the effectiveness of authentication protocols. The Gismo IdM offers a range of authentication protocols with different properties, two of which are presented in this paper. Figure 4 shows a protocol suited for a server with the necessary resources to implement replay protection. The data elements needed for mutual authentication (signature, timecounter, nonce, servername) are *piggybacked* on the request and response messages in order to save a protocol round trip. The remaining security risk, which results from this choice is marginal, is pointed out in Section VI. The optional duplicate/replay detection happens locally in the server and does not affect the protocol data units.

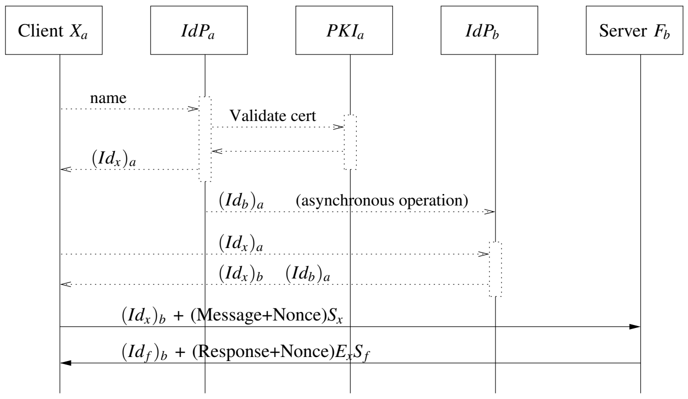
Client X_a
IdP_a
PKI_a
IdP_b
Server F_b
name
Validate cert
$(Id_x)_a$
$(Id_b)_a$ (asynchronous operation)
$(Id_x)_a$
$(Id_x)_b$ $(Id_b)_a$
$(Id_x)_b$ + (Message+Nonce)S_x
$(Id_f)_b$ + (Response+Nonce)E_xS_f

Figure 5. The authentication protocol for the stateless service.

Figure 5 illustrates the much simpler authentication to a stateless service. All requests are processed since they do not alter the system state (other than consume resources), but the authentication requirements are enforced through the encryption of the response. The request is signed to protect its integrity. The response is signed for the purpose of server authentication, and includes a nonce for protection against response replay. The nonce is not remembered across invocations and introduces no state space in the sever.

E. A replacement for the X.500 Distinguished Name

In an X.509 public key certificate, the subject is identified though the use of an X.500 Distinguished Name (DN). The author's DN might be: `CN=Anders Fongen, O=FFI,C=NO`. Several forms are likely, and each form can have a number of string representations, e.g., `/CN=Anders Fongen/O=FFI/C=NO`.

During an authentication process, this is the subject identifier that relates to the operation taking place.

A signed document, sent from the E-mail address `anders.fongen@ffi.no`, relates its signature to the identifier `CN=Anders Fongen,O=FFI,C=NO`. It is not possible to decide if the two identifiers relate to the same subject. It is possible to write X.500 DN in a form that maps to an E-mail identifier in the RFC-822 form, but in general, there are no mapping rules.

The X.500 DN made sense back when there was X.400 E-mail which used this form for message addressing. Today, when the RFC-822 form is prevalent, authentication should take place on the E-mail address of the subject, or other identifiers that relates to the subject in subsequent operations (like DNS-name or IP address). One reason that X.500 DN is still in use is that the LDAP directory protocol mandates its use as a lookup key.

The Gismo IdM has been built to use the *Subject Alternative Name* extension of the X.509 certificate (created by the PKI) in the generation of identity statements, which means that the X.500 DN is visible only to the IdP, not to the subjects. This extension can hold the subjects RFC-822 E-mail identifier, a DNS domain name, and more. It allows for a more straightforward processing of access rights, usage policies etc., since the authenticated identifier is more intuitively related to the subject.

Due to the use of X.500 DN in the LDAP lookup protocol, it is not feasible to lookup a certificate using the Subject-Alt-Name extension. The X.500 DN must be used in the IdP request for an identity statement. This poses no problem, since the subject knows its own X.500 DN from its certificate (which must be preinstalled). It is also necessary to use X.500 DN in cross-COI identity statements for the same reason.

VIII. Identity management for mobile units

The inclusion of mobile units in a framework for identity management and mutual authentication is highly desired since the focus of this research is tactical military systems. The resulting design will to a large extent be influenced by the resource situation in mobile networks; Narrow bandwidth, frequent disconnections and network partitions, limitations of software platform and user interface.

The choice of this prototype has been to include mobile units which are based on the Android operating system [14]. Android systems are readily programmable in the Java programming language. Java programs are compiled into a distinct bytecode and executed in a virtual machine called Dalvik. Source code portability from Java Standard Edition (Java SE) to the Android platform is good, but limited by the availability of some builtin packages. Packages related to user interface, remote invocation or J2EE operations will not port. Otherwise, the portability allows the relevant Gismo IdM code to compile to a Dalvik engine with little effort. An important property of Dalvik is that the object serialization format is compatible with the Java VM so that the two platforms can exchange their native objects in serialized form.

Gismo IdM was initially built with the use of XML syntax representation of external data units. The identity statements were represented as SAML assertions [10] and the service requests/responses were encoded according to the SOAP message standard and the authentication protocol data was put in the SOAP headers using the WSSec standard [11]. The software library used to support the manipulation of SAML and WSSec objects (Sun XWSS 2.0) is not available under Android and will probably never be. Besides, the SAML and WSSec standards are so complex that "barefoot" processing using string operations etc. is not feasible.

Therefore, the port of Gismo IdM to Android required a different syntax representation of external protocol data units. The choice was made to use serialized Java objects for this purpose. Serialized Java objects (from now on denoted POJO - Plain Old Java Objects) can be exchanged between any Android/Dalvik, Scala and Java SE systems (although not Java ME, which lacks a serialization engine). The choice was made for reasons of network efficiency and ease of programming.

Of course, moving from a platform neutral XML syntax to a platform specific POJO representation affects interoperability. Gismo IdM participants do not talk to other IdM systems, since it employs non-standard protocols. Therefore, interoperability becomes more of a *portability* problem, i.e., which systems can the Gismo IdM implementation code be ported to?

The SOAP based implementation can only be ported to systems with library support for WSSec and SAML. The Java library used in the prototype (Sun XWSS 2.0) provide good portability for Java SE enabled platforms. For other systems, lack of SAML support may inhibit the port since it is not feasible to write that code from scratch. An in-house attempt to port the authentication protocol to .NET platform failed due to bugs in the WSSec library.

The POJO based implementation can only be ported to systems with the Java serialization engine. Such systems include Java SE, Dalvik and Scala, although the latter has not been tested.

It is therefore not true that an IdM is portable simply because it uses "open standards", due to the sheer complexity of the specification. Neither is it true that a POJO based protocol is less interoperable than a SOAP/SAML based protocol.

No "one-size-fits-all" solution appears to be available, but rather than making separate stovepipe IdM systems for the different environments, it appears sensible to design an IdM which allows different syntax representations of the data elements to co-exist.

A. SOAP vs. POJO interoperability

The GISMO IdM contains nodes which use different presentations for identity statements and service invocations. In order for two nodes to communicate, they must use the same communication stack, including the presentation layer. A client using serialized POJOs can therefore not communicate with an IdP or a service requiring SOAP message syntax and vice versa. For a client to reach the services it needs, regardless its choice of presentation syntax three approaches can be taken:

1) Make services (and the IdP) dual-stack.
2) Make a general proxy for automatic conversion between the presentation forms (POJO and SOAP), e.g., based on Sun JAXB.
3) Make a specific proxy for each service

Option 1 is not a possible solution, since we introduced the dual representation form precisely due to the lack of SAML/WSSec support in mobile systems. Some nodes may be equipped with two stacks, but not all.

Option 2 has not been studied in detail, but requires a combination of WSDL-compilation and JAXB-assisted conversion. It is not likely to be possible to convert on-the-fly any POJO to a SOAP message which conforms to the WSDL-file of a particular web service.

Option 3 has been studied and tested, and represents an attractive approach. A proxy service takes the parameter values and passes them to a precompiled web services stub (generated by the WSDL compiler). The return value from the stub is passed back to the caller of the POJO service. Example code lines required for this function are shown below:

```
public class MainClass {
  public Serializable service(WeatherRequest wr,
                         Properties props) {
    try {
      Weather w = new Weather();
      String result = w.getWeatherSoap()
         .getWeather(wr.town);
      return result;
    } catch (Exception e) { return e; }
  }
}
```

Option 3 is also attractive since it gives the developer control over service aggregation and orchestration. One service call to a POJO service need not be passed on as one single web service invocation. Many individual calls may be made, and they may be sequenced or tested in any manner. Aggregated operations are useful because they potentially reduce the network traffic to and from the mobile unit, which is likely to be connected through a disadvantaged link. The proxy can even cache results for subsequent service calls.

For options 2 and 3 there is a problem related to signature values. Equivalent POJO and SOAP messages will have different signature values, and the integrity of the message is broken during a conversion. The proxy can sign the converted object using its own private key, which would require that the service accepts that the proxy vouches for the original client in the authentication phase.

IX. Experimental evaluation

The Gismo IdM has been subject to a series of experimental evaluations, mostly for testing correctness of algorithms and implementation, and to study configuration and deployment options. The experiments have confirmed the correctness of the protocol design and the feasibility of the implementation. During 2012 the Gismo IdM will be a part of larger field experiment where the secure exchange of information within a military coalition will be in focus. Technologies in use will include IPSec, IPv6, XMPP messaging protocol, Gismo IdM, security gateways, Android, Linux etc. The Gismo IdM will be evaluated with the perspectives of interoperability with other security technologies, its performance when XMPP is used as transport protocol, and its stability and resilience when run in a disadvantaged network.

X. Conclusion

The necessity to offer identity management across a wide range of equipment and communication technologies is the background for this article. A number of properties has been identified as important for the successful construction, deployment and operation of an identity management system. Investment protection, domain autonomy and prudent resource consumption are key elements.

The Gismo IdM prototype has been described in detail. It is an implementation of the proposed design principles and a basis for experimental evaluation. Its dual stack implementation of PDU syntax representation raises interoperability concerns which have been discussed.

Further work on the Gismo IdM includes an experimental evaluation in a military field maneuver of mobile coalition partners, where performance and interoperability properties will be assessed. Also, the work on interoperable syntax representation for identity statements and service requests/responses will be continued. The goal is to look for representations that are able to retain the signature integrity to a greater extent that what is presently possible.

References

[1] A. Fongen, "Architecture patterns for a ubiquitous identity management system," in *ICONS 2011*. Saint Maartens: IARIA, Jan. 2011.

[2] J. Hughes, S. Cantor, J. Hodges, F. Hirsch, P. Mishra, R. Philpott, and E. Maler, *Profiles for the OASIS Security Assertion Markup Language (SAML) V2.0.*, OASIS Standard, March 2005.

[3] "Shibboleth." [Online]. Available: http://shibboleth.internet2.edu/ [retrieved Dec 21, 2011]

[4] "OpenID." [Online]. Available: http://openid.net/ [retrieved Dec 21, 2011]

[5] "The Libery Alliance." [Online]. Available: http://www.projectliberty.org/ [retrieved Dec 21, 2011]

[6] R. Sandhu, D. Ferraiolo, and R. Kuhn, "The NIST model for role-based access control: towards a unified standard," in *RBAC '00: Proceedings of the fifth ACM workshop on Role-based access control*. New York, NY, USA: ACM, 2000, pp. 47–63.

[7] A. Bhargav-Spantzel, A. C. Squicciarini, and E. Bertino, "Establishing and protecting digital identity in federation systems," *J. Comput. Secur.*, vol. 14, pp. 269–300, May 2006.

[8] A. Fongen, "Scalability analysis of selected certificate validation scenarios," in *IEEE MILCOM*, San Diego, CA, USA, Nov. 2010, pp. 1–7.

[9] C. Wallace and G. Beier, "Practical and secure trust anchor management and usage," in *Proceedings of the 9th Symposium on Identity and Trust on the Internet*, ser. IDTRUST '10. ACM, 2010, pp. 97–107.

[10] N. Ragouzis, J. Hughes, R. Philpott, E. Maler, P. Madsen, and T. Scavo, *Security Assertion Markup Language (SAML) V2.0 Technical Overview*, OASIS Committee Draft, March 2008.

[11] K. Lawrence and C. Kaler, *Web Services Security: SOAP Message Security 1.1*, OASIS Standard Specification, 2004.

[12] ——, *WS-Trust 1.4*, OASIS Standard, 2009. [Online]. Available: http://docs.oasis-open.org/ws-sx/ws-trust/v1.4/os/ws-trust-1.4-spec-os.pdf [retrieved Dec 21, 2011]

[13] A. Fongen, "Identity management without revocation," in *SECURWARE 2010*. Mestre, Italy: IARIA, July 2010.

[14] ——, "Federated identity management for Android," in *SECURWARE 2011*. Nice, France: IARIA, July 2011.