International Journal of Network Security & Its Applications (IJNSA), Vol.3, No.3, May 2011

Enterprise Delegation for Service Based Systems

Coimbatore Chandersekaran[1], William R. Simpson[2]

[1] The Secretary of the Air Force (SAF/A6) 1500 Wilson Blvd., Rosslyn, VA 22209, US
Coimbatore.Chandersekaran.ctr@pentagon.af.mil

[2] The Institute for Defense Analyses, 4850 Mark Center Drive, Alexandria, VA 22311, USA
rsimpson@ida.org

Abstract. *Sharing information and maintaining privacy and security is a requirement in distributed environments. Mitigating threats in a distributed environment requires constant vigilance and defense-in-depth. Most systems lack a secure model that guarantees an end-to-end security. We devise a model that mitigates a number of threats to the distributed computing pervasive in enterprises. This authentication process is part of a larger information assurance systemic approach that requires that all active entities (users, machines and services) be named, and credentialed. Authentication is bi-lateral using PKI credentialing, and authorization is based upon Security Assertion Markup Language (SAML) attribution statements. Communication across domains is handled as a federation activity using WS-* protocols. We present the architectural model, elements of which are currently being tested in an operational environment. Elements of this architecture include real time computing, edge based distributed mashups, and dependable, reliable computing. The architecture is also applicable to a private cloud.*

Keywords: Credentialing, Authentication, Authorization, Delegation, Attribution, Least Privilege, Public Key Infrastructure, Security Assertion Markup Language (SAML), WS-*

1. Introduction

Today's Information Technology (IT) systems are under continual attack by sophisticated and resourced adversaries seeking to ex-filtrate information, deny services, and create other forms of mayhem. The strength of the threat is almost directly proportional to the assets being protected. An example might be a banking industry enterprise such as a clearing house for electronic transactions, allied health operations which provide needed information to health care providers while trying to maintain privacy concerns, defense industry applications, even credit card consolidation processes that handle sensitive data both fiscal and personal. The attacks have been pervasive and continue to the point that nefarious code may be present, even when regular monitoring and system sweeps remove readily apparent malware. One class of attack is the Man-in-the-Middle (MITM). This attack manifests itself in various ways including *Wi-Fi Traffic Intercept, Rogue Access Points, Browser (HTTP) Domain Naming Service (DNS) Cache Poisoning, Overriding Same Origin Policy, etc*. The principal of these attacks lies in eavesdropping on, or injecting into network traffic, intercepting and responding on behalf of anticipated communication end-points. The attacks are not limited to a single layer and can be present in any layer of the Open System Interconnect (OSI) model. Thus, a MITM can efficiently manifest itself when a less than comprehensive set of safeguards have been employed. Recently, MITM attacks have bypassed security that leverages single authentication by posing as the target of a communication. This discounts the previously held notion that deploying a single, two-factor authentication mechanisms could provide protection against MITM.

Despite these attacks environment, the web interface is a useful approach to providing access to many distributed users. One way to continue operating in this environment is to not only know and vet your users, but also your software and machines (all active entities). Even that has limitations when dealing with the threat environment. Today we regularly construct seamless encrypted communications between machines through Secure Socket Layer (SSL) or other

DOI : 10.5121/ijnsa.2011.3310

Transport Layer Security (TLS). These do not cover the "last mile" between the requestor (either a user or service) on one end, and the service on the other end. This last mile is particularly important when we assume that malware may exist on any machine, opening the transactions to exploits, ex-filtration, session high-jacking, data corruption, MITM, masquerade, denial of service, and other nefarious behavior.

Though much has been published about securing the enterprise against adversaries such as, MITM attacks, the enterprise and distributed computing infrastructure remains vulnerable to both internal and external adversaries. Security solutions have failed to mitigate the threats from a perspective of strong bi-lateral and end-to-end authentication. That is, accounting for the identity of all recipients of an initiated communication. The current process of authentication which terminates at intermediate points during service execution exposes the requestor to hostile threats, such as, those elaborated earlier. The use of proxies, reverse proxies, abstract addressing and other techniques present a large number of intermediate attack points.

The challenge to building an end-to-end secure computing model is to provide a mechanism by which messages originating from any entity remain targeted, integral, and confidential all the way to the its destination, regardless of whether or not the message is routed through intermediary nodes.

In this paper, we describe a process model that mitigates the cited threats. We devise an architecture by which we can provide integrity and confidentiality of messages across distributed boundaries preceded by bi-lateral authentication of active entities. All active entities are named, registered, credentialed and authorized to participate in any given environment.

The remainder of the paper is structured as follows. Section 2 provides the basic tenets around which the enterprise security is formulated. Section 3 describe the generic overview of our approach; service paradigm, bi-lateral authentication, and cascading authentication. Section 4 provides SAML process requirements. Section 5 provides some data on the first operational tests. Section 6 reviews related work. Finally, we conclude in Section 7.

2. Tenets of Information Assurance (IA) Architecture Efforts

This section provides nine tenets that guide decisions in an architectural formulation and implementation approaches [12]. These tenets are separate from the "functional requirements" of a specific component (e.g., a name needs to be unique); they relate more to the needs and requirements of the solution that guide its implementation.

- The **zeroth** tenet is that the *enemy is embedded*. In other words, rogue agents may be present and to the extent possible, we should be able to operate in their presence, although this does not exclude their ability to view some activity.
- The **first** tenet is *simplicity*. This seems obvious, but it is notable how often this principle is ignored in the quest to design solutions with more and more features. That being said, there is a level of complexity that must be handled for security purposes and implementations should not overly simplify the problem for simplicity's sake.
- The **second** tenet, and closely related to the first is *extensibility*. Any construct we put in place for an enclave should be extensible to the domain and the enterprise, and ultimately to cross-enterprise and coalition. It is undesirable to work a point solution or custom approach for any of these levels.
- The **third** tenet is *information hiding*. Essentially, information hiding involves only revealing the minimum set of information to the outside world needed for making effective, authorized use of a capability. It also involves implementation and process hiding so that this information cannot be farmed for information or used for mischief.

- The **fourth** tenet is *accountability*. In this context, accountability means being able to unambiguously identify and track what active entity in the enterprise performed any particular operation (e.g. accessed a file or IP address, invoked a service). Active entities include people, machines, and software process, all of which are named registered and credentialed. By accountability we mean attribution with supporting evidence. Without a delegation model, it is impossible to establish a chain of custody or do effective forensic analysis to investigate security incidents.
- Th**is fifth** tenet is *minimal detail* (to only add detail to the solution to the required level). This combines the principles of simplicity and information hiding, and preserves flexibility of implementation at lower levels. For example, adding too much detail to the access solution while all of the other IA components are still being elaborated may result in wasted work when the solution has to be adapted or retrofitted later.
- The **sixth** is the emphasis on a *service-driven* rather than a product-driven solution whenever possible. Using services makes possible the flexibility, modularity, and composition of more powerful capabilities. Product-driven solutions tend to be more closely tied to specific vendors and proprietary products. That said, commercial off-the-shelf (COTS) products that are as open as possible will be emphasized and should produce cost efficiencies. This means that for acquisition functionality and compatibility are specified as opposed to must operate in a Microsoft forest [18] environment.
- The **seventh** tenet is that *lines of authority* should be preserved and IA decisions should be made by policy and/or agreement at the appropriate level.
- The **eighth** tenet is *need-to-share* as overriding the need-to-know. Often effective health, defense, and finance rely upon and are ineffective without shared information.

3. Approach

In this section we provide a detailed approach. First, we develop the concepts of naming, credentialing, authentication, authorization of all entities to participate in the environment. This is followed by our representation of a service-based paradigm, which details the components of a service. This is followed by our process model of bi-lateral authentication with the section closing on cascading authentication. We follow with Security Assertion Markup Language (SAML) processes for maintaining access control compatible with the cascading authentication. Note we assume a single enterprise where we have control of these details. For cloud computing this means we must have a private cloud that is not shared by other enterprises.

3.1 Upfront Requirements

Naming criteria for entities requires names that are unique over space and time. All entities are given a unique common name, and an alias for the common name that appears in the list of identity attributes in a registry. Entity credentials are issued to the entity using a trusted certificate authority and the certificate provides asymmetric PKI keys the private key will be under control of the certificated entity, and the certificates may be stored in software caches or hardware modules. A key length of 256-bit or more is recommended. Bi-lateral authentication uses certificates provided as credentials to authenticate entities to one another followed by the push of a SAML token for authorization. In the next subsection we provide an overview our representation of a service-based paradigm.

3.2 A Service-based Paradigm

All web applications, services, and devices exercise access controls and use SAML Assertions [5] in their decision process. The requestor will not only authenticate to the service (not the server or device), but the service will authenticate to the requestor. The interface is termed a

"Robust" Application Programming Interface (API), or in the case of a browser or presentation system it is a "Robust" browser. This terminology is used to avoid specifying the implementation details which may be by a browser appliqué (a small program embedded in the browser), an appliance, a set of class libraries or other mechanisms to implement the functionality. Several pilots are under way in each of these approaches. Figure 1 shows the constituent makeup of a service.

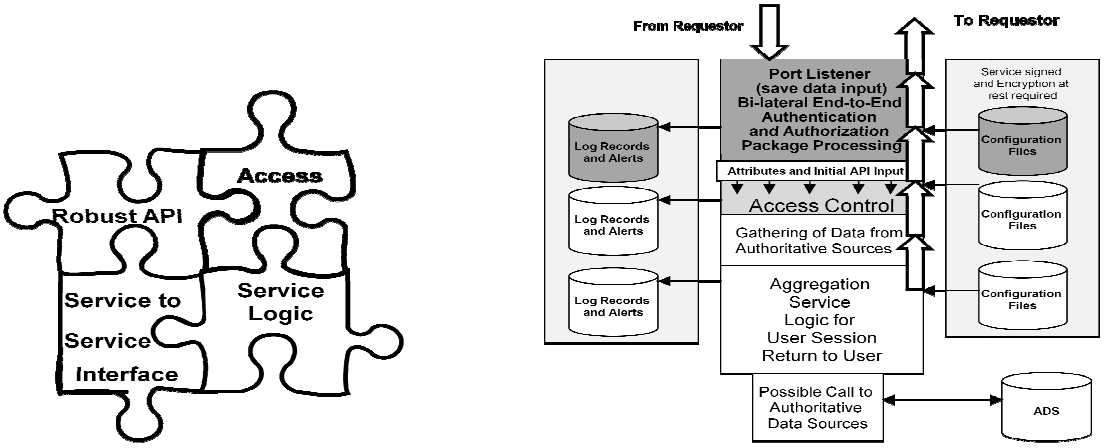

Figure 1: Components of a Service

The Robust API must be compatible with the Robust Browser. The Robust browser allows the use of WS-* protocols for security and exchange of information in XML. It also either provides the presentation protocols or translates them to HTML for browser display. Without the robust browser, the initial service request is limited to HTTPS protocols (mutual authentication SSL based upon PKI credentials) and using HTML for presentation purposes. Under these circumstances the first service in the chain is termed as a Web Application. In either case we enforce bi-lateral end-to-end authentication (using PKI credentials of each of the active entities – people, machines, applications and services) and authorization by the use of SAML tokens. Infromation is derived from authoritative Data Sources (ADS) as labeled in the figure.

The access component is responsible for holding access control privileges in the operation of a service. The service logic component is responsible for what a service does. For example, aggregating and retrieving of data. The service to service interface is handled in the following paragraphs. It is therefore important that each service exercise compatible code segments, libraries or other mechanisms. Service to service calls (or web application to service calls) are handled in accordance with Figure 2.

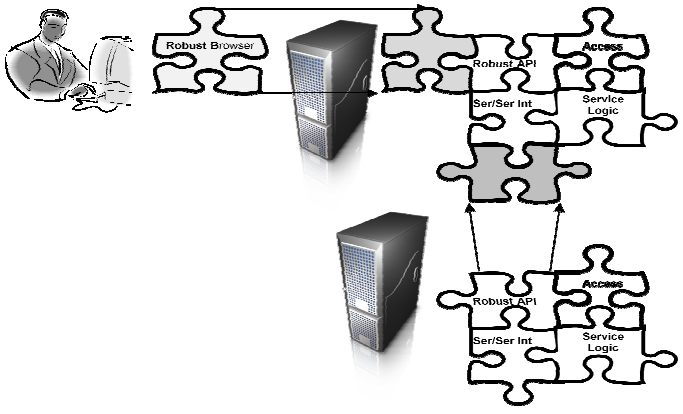

Figure 2: **Compatible Services**

Figure 3 shows two types of Services; an Aggregation Service and an Exposure service. The Aggregation Service may expose data and it may also call exposure services. Exposure services provide data from designated Authoritative Data Sources[1] (ADS). The aggregation service then aggregates the data modifies their output as necessary and returns the data to the user. The requests to exposure services are made through the interface termed robust API. It does this through an addressed message to the API using WS-* protocols for security, including SAML credentials for authorization, and exchange of information is provided in XML. The Exposure Service provides data from an authoritative data source. The "robust" Service call may be different between services than between browser and service. The "robust" APIs will also be different for different environments (e.g., .NET or J2EE). The "robust" part of the API consists of (see Figure 1):

- Port Listener
- Retain data input for reuse
- Complete the bi-lateral end-to-end authentication
- Consume the assertion package for authorization
- Pass Authorization credentials and initial input to the service

The initiating part on the "robust" Browser and the Service-to-Service invocation must meet the compatibility issues, including the initiation of bi-lateral end-to-end authentication and the passing of a SAML token for authorization.

3.3 Bi-lateral End-to-End Authentication

As a pre-requisite to end-to end communication an SSL or other suitable TLS is setup between each of the machines. Each communication link in the Figure 3 will be authenticated end- to-end with the use of public keys in the X.509 certificates provided for each of the active entities. This two way authentication avoids a number of threat vulnerabilities. The requestor initially authenticates to the service provider. Once the authentication is completed, an SSL connection is established between the requestor and the service provider, within which a WS-Security package will be sent to the service. The WS-Security [7, 10] package contains a SAML token generated by the Security Token Server (STS) in the requestor domain. The primary method of authentication will be through the use of public keys in the X.509 certificate, which can then be used to set up encrypted communications, (either by X.509 keys or a generated session key). Session keys and certificate keys need to be robust and sufficiently protected to prevent malware exploitation. The preferred method of communication is secure messaging using WS Security, contained in SOAP envelopes. The encryption key used is the public key of the target, ensuring only the target can interpret the communication.

[1] These data sources must be pre-designated by communities or programs as the authoritative sources. These are updated frequently and checked for integrity and accuracy. They may be mirrored for efficiency of operations.

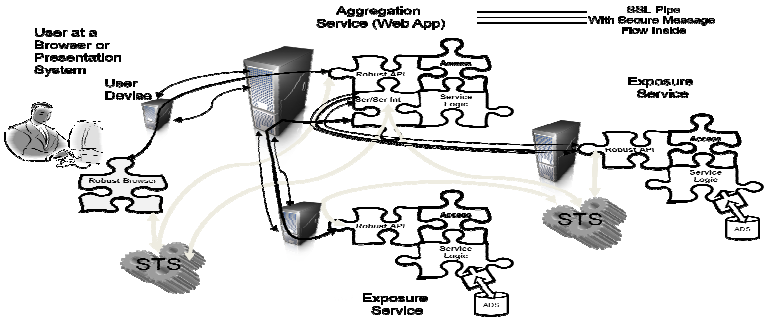

Figure 3: Steps in invoking an Aggregation Service

3.4 Cascading Authentication

This section outlines a process for cascading authentication, a key concept of our approach. This process involves a sequence of certificates that provide the history and delegation of the service chain. The chain of authentication may be used to shape the SAML assertions for least privilege and are sent with each service request allowing the recipient determine authorization (if any) that will be provided the sender of a message. The authentication involves presenting the PKI certificate(s) of the requestor to the service and vice-versa. Cascading authentication presents all of the PKI certificates in the chain so that after authentication the target will have knowledge of each step in the chain. Figure 4 illustrates our concept of cascading authentication.

The SAML may then be pruned or modified to reflect this whole chain, and the logs would contain the ***OnBehalfOf*** based upon the chain of credentials. This way, one knows whom one is acting on behalf of who at all times. Delegation of authority is then defined by the chain of credentials presented for authorization.

By delegation we simply mean the handing of a task over to another entity by software service calls. A second form of delegation, personal delegation, must be handled separately. This involves an individual tasking another individual to produce work for him. This second type of delegation is described in Annex A.

The software delegation is the assignment of authority but not responsibility to another software entity to carry out specific activities. Further, it is assumed that any service invoking another service is delegating its authority to complete whatever portion of the service it has been authorized to perform. Delegation for a service is transitive and not personal. This delegation occurs at levels 5 and above in the OSI model. Levels 4 and below are handled by defined middleware definitions. Delegation only lives during the session under consideration. We now introduce two terms that are closely tied to delegation; attribution and least privilege.

Attribution is provided when the service exercising privilege is identified as acting on behalf of the requestor who (implicitly) authorized the delegation. *Least Privilege* is preserved by providing the entity with only that level of privilege necessary to do the task without exceeding his/her own authority.

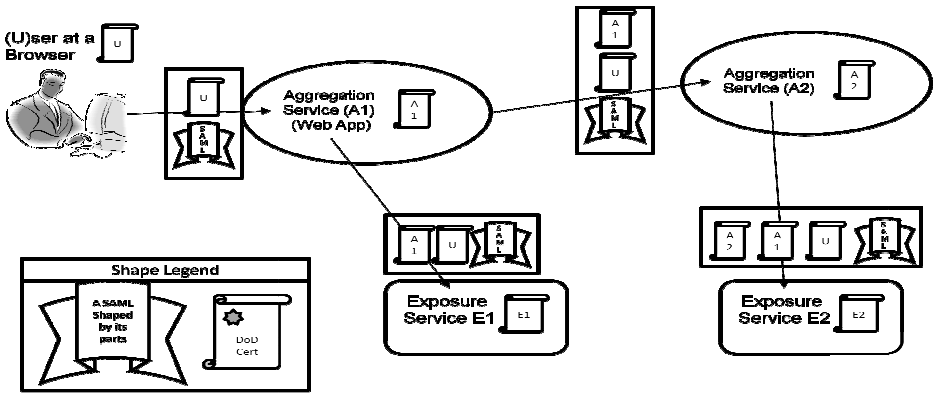

Figure 4: Cascading Authentication Architectural Overview

4. Shaping the SAML

4.1 Basic Use Case

The basic use case is given in the Figure 5 and involves a user invoking an aggregation service which in turn invokes aggregation and other services.

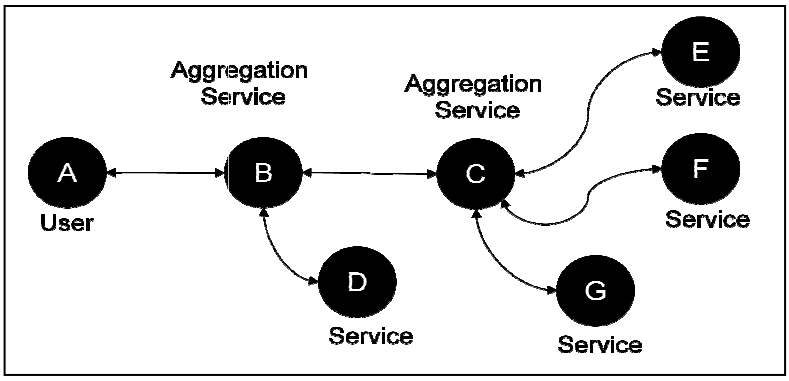

Figure 5: Use Case for Service Delegation

4.2 Communication for Authentication/Authorization

Each communication link in Figure 5 will be authenticated end-to-end with the X.509 certificates provided for each of the active entities. Authorization will be based upon the Security Assertion Markup Language (SAML).[2] The delegation, attribution and least privilege will be handled by modification to the SAML token provided by the STS. The SAML token for user A to aggregation Service B is provided in the Table 1 below:

4.2.1 Pruning Attributes[3]

An individual or service requesting another service may contain many elements that are not relevant to the service request. This makes the SAML request overly large, increases the cycles for SAML consumption and evaluation may introduce additional latency and is a potential source for escalation of privilege. In order to combat these factors, the attribute assertion should be reduced to the minimum required to accomplish the service request.

[2] Security Assertion Markup Language (SAML) is part of the OASIS set of Web Service Standards.

[3] Since authorization decisions may require any of a combination of attributes, groups, and/or roles, these will be referred to generically as elements in the rest of this chapter.

Table 1 SAML 2.0 Format for User Request

Item	Field Usage	Recommendation	Notes
SAML Response			
Version ID	Version 2.0	Required	
ID	(uniquely assigned)	Required	
Issue Instant	Timestamp	Required	
Issuer	Yes	Required	STS Name
Signature	Yes	Required	STS Signature
Subject	Yes For User A	Required	Must contain the X.509 Distinguished name or equivalent
Attribute Assertion			
Subject	Yes For User A	EDIPI (user common name)	For Attribution
Attributes, Group and Role Memberships	Yes For User A	Required	May be pruned for least privilege
Conditions			
NotBefore	Yes	Required	TimeStamp - minutes
NotAfter	Yes	Required	TimeStamp + minutes
OneTimeUse	Yes	Required	Mandatory

4.2.2 REQUIRED ESCALATION OF PRIVILEGE

Certain services may require privilege beyond that of the original client. Examples include the Security Token Server (STS) that when called is expected to have access to the Active Directory (AD) and UDDI, even when the client does not have such privilege. An additional example would include payroll services that can provide average values without specifics. The service must be able to access all records in the payroll data base, even if the client it is acting on behalf of does not have this privilege. For purposes of this methodology, these required elements will be dealt with separately in both data pruning and service to service calls. Service developers should take care that the required escalation of privilege is required and that the newly aggregated data do not impose additional access restrictions. The data that has been aggregated and synthesized should be carefully scrutinized for such sensitivities. The process is not unlike the combining of data from multiple unclassified but sensitive data sources that may rise to a higher classification level when they are all present in one place.

4.2.3 DATA REQUIREMENTS - PRUNING ELEMENTS

In order to accomplish the reduction of the SAML assertion, the STS must know the target and the elements that are important to the target. Table 2 below presents such a data compilation. This table will be used in the subsequent example. An element is an attribute, role or group used in the authorization decision.

Table 2 Group and Role Pruning Data Requirement

Service	Uri	Relevant Attributes, Groups and Roles	Escalation of Privilege Required
AFPersonnel30	…//afnetdol.pers.af23:622	Element1, Element3, Element4, Element5, Element6	Element6
PERGeo	…//afnetdol.perst.af45:543	Element4, Element5,	Element6

Service	Uri	Relevant Attributes, Groups and Roles	Escalation of Privilege Required
		Element6	
PerReg	**…//afnetdol.persq.af45:333**	**Element4**	
PerTrans	**…//afnetdol.persaw.af45:21862**	**Element6**	
BarNone	**…//afnetdol.persaxc.af45:1234**	**Element5**	
DimrsEnroll	**…//afnetdol.persws.af45:23567**	**Element1, Element3**	
…	**…**	**…**	
Endfile			

The combining of these elements is given for calling step i by:

Let N_{i+1}=New SAML Elements for i to call i+1
Let P_i = Prior Elements
Let R_{i+1}= Service Required Elements
Let H_i = Service Held elements
Let E_i = Required Escalation Elements

Then:

$$N_{i+1} = (P_i \cap (R_{i+1} \cap H_i)) \cup (E_i \cap R_{i+1}) \quad (1)$$

Where: $\cap$ is the intersection of sets and $\cup$ is the union of sets, $\emptyset$ is the empty set (no members). The formula may be read as the common elements in the prior SAML and the intersection of the held elements and those required by the next call (($Pi \cap (R_{i+1} \cap H_i)$) - normal least privilege). These are added ($\cup$) to the required escalation elements that are required to be extended by the next call (($E_i \cap R_{i+1}$) - extended least privilege by escalation of privilege). The initial call has no prior elements and P_1 is defined as the initial set of privilege elements. This reduces N_1 to:

$$N_1 = H_0 \cap R_1 \quad \text{(Normal least privilege)} \quad (2)$$

4.3 Subsequent Calls Require Saving the SAML Assertion

After the SAML is consumed and authorization is granted, the service must retain the SAML Attribute Assertion (Part of the Larger SAML Token) above. Specifically, the subject fields and the elements field to be used in further authorization. The specific instance is shown in Table 3.

Table 3 Retained Portion of SAML Token

Attribute Assertion			
Subject	**Yes For User A**	**EDIPI**	**For Attribution**
Attributes, Group and Role Memberships	**Yes For User A**	**Required**	**Mask for follow-on least privilege**

4.3.1 SAML Token Modifications for Further Calls

The Attribute Assertion of Table 4 is returned to the STS for modification of the normal SAML token. The SAML Token for the unmodified service call is given below:

Table 4 Unmodified SAML for Service B of Use Case

Item	Field Usage	Recommendation	Notes
SAML Response			
Version ID	**Version 2.0**	**Required**	

Item	Field Usage	Recommendation	Notes
ID	**(uniquely assigned)**	**Required**	
Issue Instant	**Time-stamp**	**Required**	
Issuer	**Yes**	**Required**	**STS Name**
Signature	**Yes**	**Required**	**STS Signature**
Subject	**Yes For Service B**	**Required**	**Must contain the X.509 Distinguished name or equivalent**
Attribute Assertion			
Subject	**Yes For Service B**	**Common Name for Service B**	**For Attribution**
Attributes, Group and Role Memberships	**Yes For Service B**	**Required**	$N_{i+1} = (P_i \cap (R_{i+1} \cap H_i)) \cup (E_i \cap R_{i+1})$
Conditions			
NotBefore	**Yes**	**Required**	**TimeStamp - minutes**
NotAfter	**Yes**	**Required**	**TimeStamp + minutes**
OneTimeUse	**Yes**	**Required**	**Mandatory**

The Attribute Assertion is modified in the following way.

- The subject is modified to read "Service A OnBehalfOf" the returned SAML subject which in this case is the EDIPI (Electronic Data Interchange Personnel Identifier) of the user.
- The attribute, group and role membership (elements) are modified to include only elements that appear in both the Service B registry and the returned SAML Attribute Assertion.
- The modified SAML Token is provided in Table 5 below:

Table 5 Modified SAML Attribute Assertions for Further Calls

Item	Field Usage	Recommendation	Notes
SAML Response			
Version ID	**Version 2.0**	**Required**	
ID	**(uniquely assigned)**	**Required**	
Issue Instant	**Timestamp**	**Required**	
Issuer	**Yes**	**Required**	**STS Name**
Signature	**Yes**	**Required**	**STS Signature**
Subject	**Yes For Service B**	**Required**	**Must contain the X.509 Distinguished name**
Attribute Assertion			
Subject	**Yes contains A and B**	**Common Name (cn) B OnBehalfOf EDIPI**	**For Attribution**
Attributes, Group and Role Memberships	**Yes B restricted by A**	**Required**	$N_{i+1} = (P_i \cap (R_{i+1} \cap H_i)) \cup (E_i \cap R_{i+1})$
Conditions			
NotBefore	**Yes**	**Required**	**TimeStamp - minutes**
NotAfter	**Yes**	**Required**	**TimeStamp + minutes**
OneTimeUse	**Yes**	**Required**	**Mandatory**

Subsequent calls from Service A would use the modified token. Further, the subsequent service called would save the SAML Attribute Assertion for its further calls.

4.4 AN ANNOTATED NOTIONAL EXAMPLE

A User in the User Forest (Ted.Smith1234567890) through discovery finds the dashboard service on Air Force Personnel (AFPersonnel30) that he would like to invoke. The discovery has revealed that access is limited to users with Element1, Element3, Element4, Element5 or Element6, but that users without all of these authorizations may not receive all of the requested display. Ted does not have all of the required Elements, but is authorized for personnel data within CONUS and has Element membership in Element 1, Element 2, Element 3, Element 4, Element 7, and Element 12 + 27 other Elements not relevant. The AFPersonnel30 will typically display the following dashboard on Air Force Personnel:

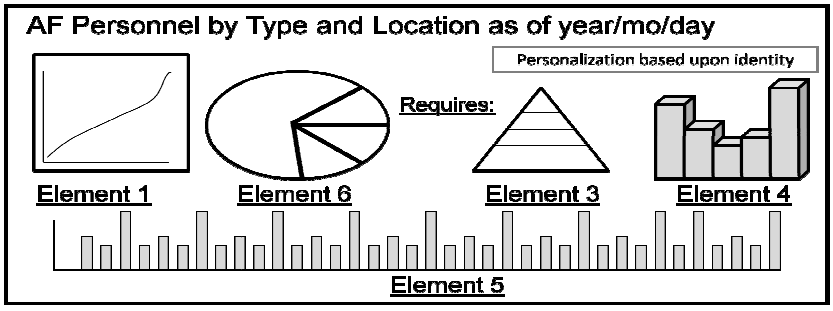

Figure 6 AFPersonnel30 with Display Outputs

The elements required would not typically be displayed. A partial calling tree for AFPersonnel30 is provided in Figure 7. The widgets that form the presentation graphics have not been included, but would be part of the calling tree, they do not have access requirements that modify the example and have been deleted for reduction of complexity. In the figure we show the elements that make up the privilege for each service (holds) and the elements required for access to the service (requires). This data is linked to Table 2, and must be synchronized with it. The element privileges for services without subsequent calls are unimportant, and many additional groups may be present but will be pruned on subsequent calls.

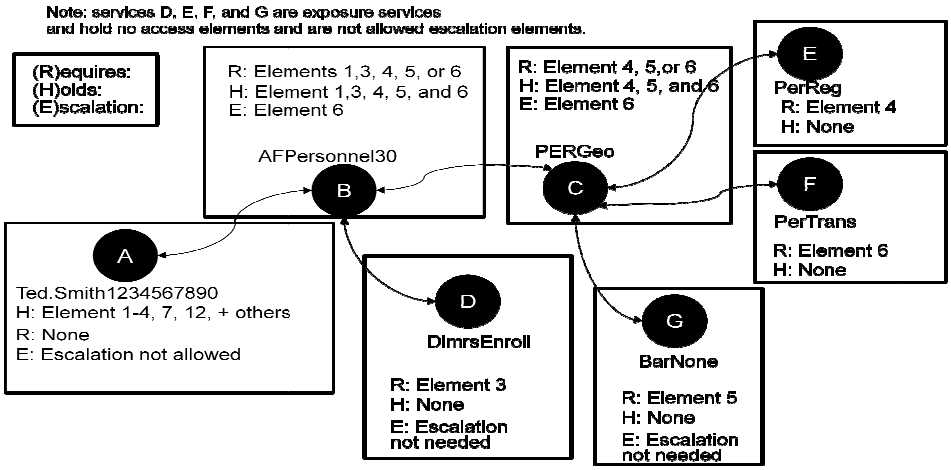

Figure 7 AFPersonnel30 Calling Tree

Note that each link in the calling graph requires bi-lateral authentication using certificates provided as credentials to each of the active entities, followed by the push of a SAML token for authorization. The first such token is presented in Table 6:

Table 6 Ted Smith SAML Push to AFPersonnel30

Item	Field Usage
SAML Response	
Version ID	**Version 2.0**
ID	**0qwdrt009kkmn**
Issue Instant	**080820081943**
Issuer	**Enterprise STS12345**
Signature	**Lkhjsfoioiunmclscwl879ooeeujl99vcd78ffgg3422ft…**
Subject	**CN = TED.SMITH1234567890, OU = CONTRACTOR, OU = PKI, OU = DOD, O = U.S. Government, C = US**
Attribute Assertion	
Subject	**TED.SMITH1234567890**
Attributes, Group and Role Memberships	**Element1, Element3, Element4**[4] $N_1 = (R_2 \cap H_1)) \cup (E_1 \cap R_2)$ $=((1, 2, 3, 4, 7, 12, +27) \cap ((1,3\text{-}6)$ $= (1,3,4))$ =((Element1, Element3, and Element4))
Conditions	
NotBefore	**080820081933**
NotAfter	**080820081953**
OneTimeUse	**Yes**

The Attribute Assertion Section is saved for subsequent calls. The call from AFPersonnel30 to service PERGeo will look like Table 7.

Table 7 AFPersonnel30 SAML Push to PERGeo

Item	Field Usage
SAML Response	
Version ID	**Version 2.0**
ID	**0qwdrt009kkmn**
Issue Instant	**080820081944**
Issuer	**Enterprise STS12345**
Signature	**Lkhjsfoioiunmclscwl879ooeeujl99xfg654bbgg34lli…**
Subject	**CN = e3893de0-4159-11dd-ae16-0800200c9a66, OU=USAF, OU=PKI, OU=DOD, O=U.S. GOVERNMENT, C=US**
Attribute Assertion	
Subject	**AFPersonnel30 OnBehalfOf TED.SMITH1234567890**
Group and Role Memberships	**Element 4**[5]**, Element6**[6] $N_{i+1} = (P_i \cap (R_{i+1} \cap H_i)) \cup (E_i \cap R_{i+1})$ $=((1, 3, 4) \cap (4 \cap 4\text{-}6) \cup (6 \cap 4\text{-}6)$ $=((1, 3, 4) \cap (4)) \cup (6)$ =(4,6) + Element 4 and Element 6

[4] An element is an attribute, role, group or combination of the previous. Elimination of Element 2, Element 7, Element 12 and other elements based on pruning (see Table 6 under AFPersonnel30)

[5] An element is an attribute, role, group or combination of the previous. Elimination of Element 1 and Element 3 based on pruning (see Table 5 under PERGeo)

[6] Element 6 is a required escalation elements

Item	Field Usage
Conditions	
NotBefore	080820081934
NotAfter	080820081954
OneTimeUse	Yes

The SAML Attribute Assertion is where the work is done. The subject has been modified to include the names of the calling tree and the Elements have been pruned to include only common items between the calling elements in the tree. Figure 8 shows the completion of the calling tree, including only the SAML Attribute Assertions in the blocks below. Note that the calls to BarNone fail access (SAML does not contain required element 5) and while being stealth to the calling routine (which will return with no data after timeout) this failure will trigger alarms to SOA management monitors as follows:

Failed authorization (BarNone) attempt PERGeo on behalf of AFPersonnel30 on behalf of Ted.Smith1234567890 No data returned

The returned dashboard (without the element requirement annotations) is presented in Figure 9. Note that Element 6 privilege was provided by service escalation.

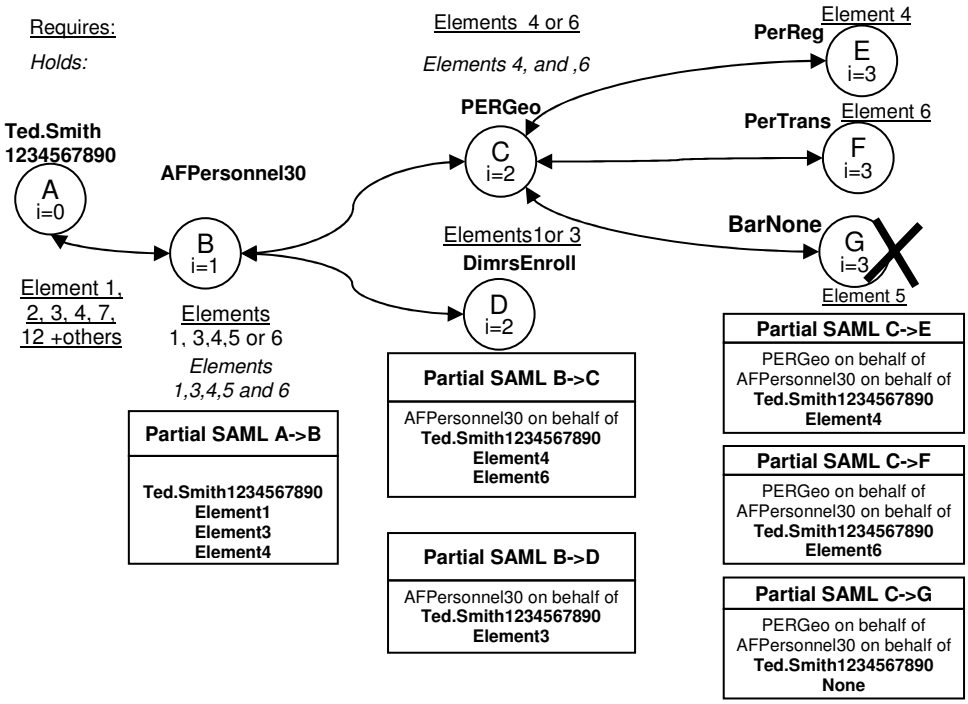

Figure 8 SAML Attribute Assertion of the Calling Tree

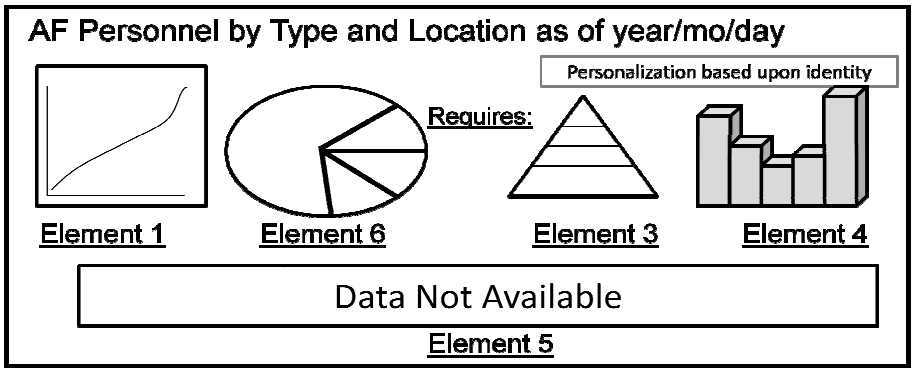

Figure 9 Dashboard Service AFPersonnel30 Case Result (with Annotation)

5. Initial Testing of Operational Quality of Service

An initial operational implementation of the architecture with full bi-lateral authentication at each step and SAML authorization produced by the Identity Provider (IdP) side of the STS was tested in June of 2010. Latency (in seconds) is measured for each step, and a total is computed for each invocation of the routines. The data are presented in Table 8 below.

The table uses the following nomenclature:

- A: Get SAML from IdP, starting at web server
- B: Get SAML directly from IdP
- C: Access services Home page
- D: Access services starting at IdP
- E: Access services, go to search page, perform search
- F: Access services starting at IdP, go to search page, perform search

The Initiation of the SAML (IdP-SAML) is the bottleneck (as indicated by analysis of the detailed data – not presented below), since its latency increases the most with increasing load. In addition, overall network traffic seems to be a contributing factor, since IdP-SAML performance degrades under both increased user loads and increased network traffic.

Throughput (successfully completed transactions per second) was maximized at between 100 and 200 users for all tests. Throughput is not a linear function of the number of users. For flow F (which is the preferred process), failure rates increased from 100-300 users while throughput remained the same at roughly 1.9 requests per second. It also depends on any wait or think time between requests. Initial data indicate a reasonable Quality of Service (QoS) with 200 users in flow F. Further optimization of the process may further improve these numbers.

Table 8 Latency and Loading

Test	Total execution single user (seconds)	Total execution 100 users (seconds)	Total execution 200 users (seconds)	Total execution 300 users (seconds)
A1 – Test set 1 Invocation through SAML response	**1.74**	**1.85**	**5.59**	**11.68**

A2 – Test set 2 Invocation through SAML response	**1.71**	**1.83**	**3.84**	**11.59**
B IdP indirect invocation	**.73**	**.84**	**4.99**	**11.21**
C1 – Test set 1 Invocation through service initiation	**7.86**	**8.00**	**21.25**	**35.75**
C2 – Test set 2 Invocation through service initiation	**4.33**	**7.13**	**21.52**	**34.05**
D IdP indirect invocation through service initiation	**3.54**	**7.38**	**20.51**	**34.39**
E Invocation through service initiation times 3 [error percent] and {success rate}	**9.89 [6%] {.035/sec}**	**41.46 [44%] {1.056/sec}**	**67.70 [22%] {1.844/sec}**	**80.55 [43%] {1.873/sec}**
F IdP indirect invocation through service initiation times 3 [error percent] and {success rate}	**8.78 [5%] {.038/sec}**	**32.71 [2%] {1.890/sec}**	**65.04 [21%] {1.890/sec}**	**92.50 [37%] {1.869/sec}**

6. Related Work

A search of the literature suggests that there has been no coordinated effort or models related to what we propose with the exception of the Globus Grid Security Infrastructure [13]. It is worth mentioning a few seminal and open standard works that make significant contributions towards the realization of our propose model. Needham and Schroeder [2] laid the foundation of public key infrastructure (PKI) upon which PKI-based works credit. Burrows et. al., [1] introduced the logic of authentication, which enable analyst to formalize the assumptions and goals of a security protocol, and to attempt to prove its correctness. When one fails to find a proof, the place at which one gets stuck often shows a potential point of attack. This analysis model turn out to be very powerful upon which the "BAN Logic" and many formal tools were developed and extended to tools used in design of protocols. Credit is further due to FIPS 196 publication on entity authentication using public key cryptography [11] and OASIS for the specification of SAML and the WS-* protocols [5,7,8,9,10],The Liberty Alliance Project [4] and the Shibboleth Project [4]. Credits are also due to some general-purpose and specialized solution for distributed system security, in particular, Kerberos, DCE, SSH, SSL, CRISIS (security component of Web-OS) [16] and Legion [17].

7. Discussion

This approach is part of a larger Information Assurance architecture to provide a more complete solution. It is worth noting that several key pieces are missing to complete this scenario. On the user end we need WS-enabled browser with the ability to communicate with a Security Token Server (STS). The STS will facilitate the exchange of credentials, aid in setting up the initial SSL, and provide the SAML package for consumption. The robust browser may be on a desktop or a mobile device or may be manifested as an appliance on the user's work station. On the service provider end we need the software to encrypt/decrypt secure message and to consume the SAML package. The latter is not trivial since it must be checked for signature, tampering, timeouts and other factors. If we assume for the moment that the user is tightly bound to the browser, then the user security context is maintained through the device and all the way to the

initial service. We need software that will read and store the authentication chain, and we need software in the STS to act upon this knowledge. This context will assist in attribution and delegation and in monitoring insider behavior activity. The remaining threats of insider activity, ex-filtration of static data and denial-of service (DoS) attacks must be handled by other means, but behavioral modeling, static encryption and dynamic ports and protocols still apply to these threats. Both the robust browser and the robust API are under development, and the initial authentication processes have been demonstrated in a pilot program.

8. Conclusions

In this paper we outline a process model that provides an end-to-end authentication as a prerequisite to authorization that accommodates intermediary nodes across distributed boundaries without sacrificing local autonomy. The model outlined herein involves many components, and will require additional software development for the pilot system to provide complete cascading of authentication. This paper has been developed to encourage the discussion and exchange ideas in making the model robust and complete for adoption in practice.

9. Acknowledgements

The authors would like to acknowledge the support of the Secretary of the Air Force's Warfighting and Integration CIO office in the development of efforts outlined in this paper.

10. References

1 Burrows, M., Abadi, M. and Needham, R., M., "A logic of authentication,' ACM Transaction on Computer Systems, vol. 8, no. 1, pp. 18-36, 1990.

2 Needham R.M., and Schroeder, R. M., 'Using encryption for authentication in large networks of computers'. Communication of the ACM, vol. 21, no. 12, pp. 993-999, 1978

3 "Internet2", Shibboleth Project, Available at http://shibboleth.internet2.edu/, 2007

4 OASIS. Identity Federation. Liberty Alliance Project. Available at http://projectliberty.org/resources/specifications.php, 2004.

5 OASIS. Profiles for the OASIS Security Assertion Markup Language (SAML) V2.0. Available at http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security, March 2005.

6 "Guide to Secure Web Services: Recommendations of the National Institute of Standards and Technology", NIST-US Department of Commerce Publication, August 2007.

7 "Web Service Security: Scenarios, Patterns, and Implementation Guidance for Web Services Enhancements (WSE) 3.0", Microsoft Corporation, 2005

8 "WS-ReliableMessaging Specification", OASIS, June 2007

9 "WS-SecureConversation Specification", OASIS, March 2007

10 "WSE 3.0 and WS-ReliableMessaging", Microsoft White Paper, June 2005, http://msdn2.microsoft.com/en-us/library/ms996942(d=printer).aspx

11 FIPS PUB 196, Federal Information Processing Standards Publication. "Entity Authentication Using Public Key Cryptography", February 18, 1997

12 Air Force Information Assurance Strategy Team, Air Force Information Assurance Enterprise Architecture, Version 1.70, SAF/XC, 15 March 2009.

13 Overview: Globus Grid Security Infrastructure, http://www.globus.org/security/overview.html. Last Retrieved, April 2009

14 Foster, I., Kesselman, C., Tsudik, G and Tuecke S., "A Security Architecture for Computational Grids Proc", 5th ACM Conference on Computer and Communications Security Conference, pp. 83-92, 1998

15 Welch, V., Foster, I., Kesselman, C., Mulmo, O., Pearlman, L., Tuecke, S., Gawor, J., Meder, S. and Siebenlist. F., "X.509 Proxy Certificates for Dynamic Delegation", 3rd Annual PKI R&D Workshop, 2004

16 Belani, E., Vahdat, A., Anderson, T. and Dahlin, M., "The CRISIS wide area security architecture", In Usenix Security Symposium, January 1998

17 Lewis, M and Grimshaw, A In Proc. 5th IEEE Symposium On High Performance Distributed Computing, "The Core Legion Object Model", IEEE Computer Society Press, 1996. Pages 562-571

18 Chandersekaran, C., Simpson W., and Trice, A., The 1st International Multi-Conference on Engineering and Technological Innovation: IMET2008, "Cross-Domain Solutions in an Era of Information Sharing", Volume I, Orlando, FL., June 2008, pages 313-318

19 Chandersekaran, C. and Simpson W., The 2nd International Multi-Conference on Engineering and Technological Innovation: IMETI2009, Volume I, "Information Sharing and Federation", Orlando, FL., July 2009, pages 300-305

20 Chandersekaran, C. and Simpson W., The International Conference on Complexity, Informatics and Cybernetics, Volume II, pages 84-89**,** "A Persona Framework for Delegation, Attribution and Least Privilege", Orlando, FL., April 2010, Co-authored by.

21 Chandersekaran, C., Ceesay, E. and Simpson W., The 3rd International Conference on PErvasive Technologies Related to Assistive Environments: PETRAE10, "An Authentication Model for Delegation, Attribution and Least Privilege", Samos, Greece, June 2010, 7 pp.

22 Chandersekaran, C. and. Simpson W, The 3rd International Multi-Conference on Engineering and Technological Innovation, "A SAML Framework for Delegation, Attribution and Least Privilege", Orlando, FL., July 2010, pages 303-308

23 Chandersekaran, C. and Simpson W., The 3rd International Multi-Conference on Engineering and Technological Innovation, "Use Case Based Access Control", Orlando, FL., July 2010, pages 297-302

Annex A: Persona Based Delegation

A.1 Need for Personal Delegation

Delegation is the handing of a task over to another person, usually a subordinate. It is the assignment of authority and responsibility to another person to carry out specific activities. It allows a subordinate to make decisions, i.e., it is a shift of decision-making authority from one organizational level to a lower one. Delegation, if properly done, is not abdication. The opposite of effective delegation is micromanagement, where a manager provides too much input, direction, and review of 'delegated' work[7].

The need for delegation in IT systems often arises out of the need to manage time and prioritize an activity, establish a posture of least privilege, and/or provide for transitioning between assignments.

- Time management issues happen when a user has a tasking that requires careful consideration of time and activity investment. In an IT system it may take the form of an administrative assistant reading and screening e-mail, or a task group leader seeking information and options to be placed in the reading files of a decision maker.

[7] Definition adapted from Wikipedia.

- Least privilege issues occur when an individual is assigned two or more roles within the organization, with differing privilege sets. Ideally, we wish the user to only have access to the minimum set of privileges associated with the role they are currently acting as in the system.
- Transitioning issues occur when an overlap exists between new and old assignments that have different access and privilege, but both must be maintained for an overlap period.
- All aspects of a delegation cannot be foreseen, but current practice of giving away login details or letting someone else use an access card (e.g., in a US DoD context, a Common Access Card or CAC), or even generating multiple logins, are unacceptable from an attribution standpoint. Delegation must be formalized so that appropriate audit and forensics can be done when system anomalies occur, or compliance measurements concerning security policy is required.

A.1.1 Personal Delegation in a Large Military Organization

In the context of a large military organization (such as the US Air Force), there are also additional complexities associated with delegation. For example, individuals can only be authorized to view documents and data no higher than the security clearance level they have been granted (e.g., Secret, Top Secret). These restrictions have to be enforced in addition to any restrictions associated with any other delegated privileges In addition, consider the case of military units that must rapidly deploy to a theater of engagement to replace another unit. Many delegation activities must take place during the transition period when both units overlap in the field.

A.2 Proposed Architecture

In this paper we propose a solution that uses a created persona for the delegate that is activated through a delegation service. A persona is a special category of user that embodies only delegated privileges, and which is explicitly assumed only after the "real" human user taking on that persona explicitly chooses it. The existence of a persona delegation is flagged in the user file, and the logon script will include a call to the delegation service for revised identification of the user. The system opens a session with delegation credentials that are inherited for the individual providing the delegation. The delegation must be recorded and registered in advance through a delegation registration service, and the delegation must be approved by written policy. The delegate persona is the responsible for actions and attribution. Actions taken by the delegate persona are recorded by audit records that have the session number assigned and the delegate persona identity (id). The delegate persona is persistent, although it should have an expiration date at the end of which it is renewed or expires ("persona non grata"). The delegate persona can be retrieved as a delegate by query to the delegation data base. When a related persona is created, the attributes under the user are modified. The last entry is provided with "Delegate," as an indication for delegation services. This field may have a default of "Normal," and a created Persona may have a value "Persona."

A.2.1 ARCHITECTURAL DETAILS

A.2.1.1. Registration Service for Principal-Agent Delegation

Principal-Agent policies are promulgated by the appropriate authority. Such policies may apply to a large class of individuals (as in the pre-screening of e-mails by administrative assistants) or to a specific instance (as in the task group lead). The principal-agent delegation registration creates a user persona that links two individuals and the delegated authority. This process involves three branches of the Directory Information Tree (DIT). Figure A-1 shows the delegation registration process. The delegation registration service is invoked and current policy is checked to see if User 2 can actually delegate. If User 2 can delegate by policy, then he is asked for the identification of the agent. If User 2 by policy can accept delegation then the registration authority creates the persona (user n), together with names and PKI and other credentials. In order for this service to work, the semantics of policy must be worked out by the Community of Interest (COI)[8]. It is expected that the policy elements will change from time to time, and the registration service should be able to read these from an input file.

At this point, the principal is offered groups that are allowed delegation. The latter is important because a number of rules will be invoked. In the absence of offered groups, the individual specified groups must be heavily screened for overall and specific policies (e.g., a principal cannot delegate privileges associated with his security clearances). Finally, the delegate persona (user n) is populated with access groups from the delegation and the agent's attributes. The delegate persona is persistent and appears in the DIT as any other user. User credentials associated with user n are the credentials associated with a new identity created by the registration service.

[8] COI are formal entities in the Air Force architecture.

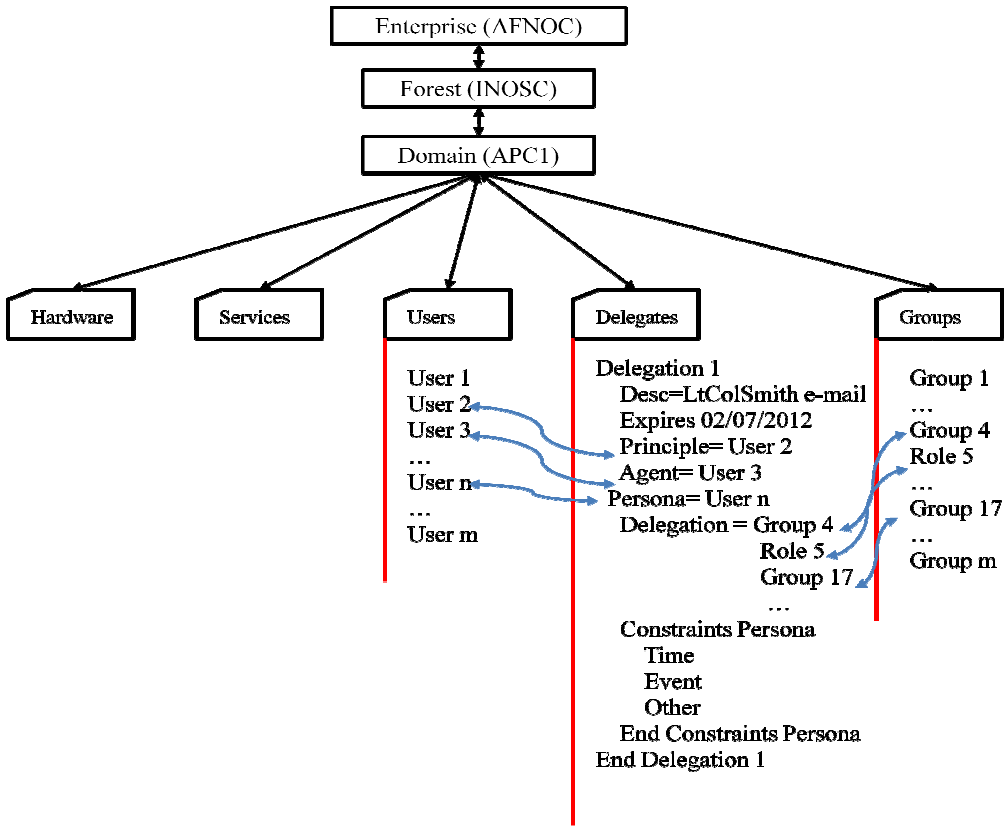

Figure A-1 Principal-Agent Delegation Architecture

A.2.1.2 Least Privilege as a Principal-Principal Delegation

A.2.1.3 User Based Least Privilege[9]

In computer science and other fields, the principle of minimal privilege, also known as the principle of least privilege or just least privilege, requires that in a particular abstraction layer of a computing environment every module (such as a process, a user or a program on the basis of the layer we are considering) must be able to access only such information and resources that are necessary to its legitimate purpose. The principle of least privilege is widely recognized as an important design consideration in enhancing the protection of data and functionality from faults and malicious behavior.
In operating systems like Windows, there is no security enforcement for code running in kernel mode and therefore such code always runs with maximum privileges. The principle of least privilege therefore demands the use of user mode solutions when given the choice between a kernel mode and user mode solution if the two solutions provide the same results.

A.2.1.4 Registration Service for Principal-Principal Delegation

Principal-Principal policies are promulgated by the appropriate authority. Such policies may apply to a large class of individuals (as in the assignment of multiple roles) or to a specific instance (as in the task breakdown for the individual). The principal-principal delegation registration creates a user persona that links two instances of an individual and the delegated authorities (or roles in some instances). This process involves three branches of the (DIT). In Figure A-2 we show the delegation registration process. The delegation registration service is invoked by either user 6 or the enclave[10] administrator on behalf of user 6 and current policy is checked to see if User 6 needs least-privilege delegation. If User 6 can delegate by policy, then he is asked for the identification of the roles or other descriptors for each self

[9] Definition adapted from Wikipedia.
[10] An enclave is defined as a set of capabilities realized by hardware, software, networks, devices, and people.

delegation including privileges associated with each. User 6 has three roles designated. The first is overall enclave administrator, the second is the COI data base manager, and the third is as a normal enclave user. Disjointness in roles will help insure that users carefully chose the role for each session. If roles are proper subsets of one another, then the maximum privilege is usually taken. This is an important principle for administration (make roles disjoint to the extent possible).

The registration authority creates the personae (user p, and q), together with names and PKI and other credentials. In order for this service to work, the semantics of self delegation must be worked out by the COI (this may be as simple as roles initially). The COI may wish to work out super groups, where a super group is a group of groups that can be used to represent a role, task, or other unique combination of authorities. It is expected that the policy elements will change from time to time, and the registration service should be able to read these from an input file. At this point, the principal or administrator is offered groups (or super groups) that are allowed in the defining of roles. The latter is important because a number of rules will be invoked. In the absence of offered (super)groups, the individual specified groups must be heavily screened for overall and specific policy. Finally, the delegate personae (users p, and q) are populated with access groups from the delegation and the agent's attributes. The self-delegate persona is persistent and appears in the DIT as any other user. User credentials associated with user p and q are the credentials associated with the original identity in self-designation (user 6).

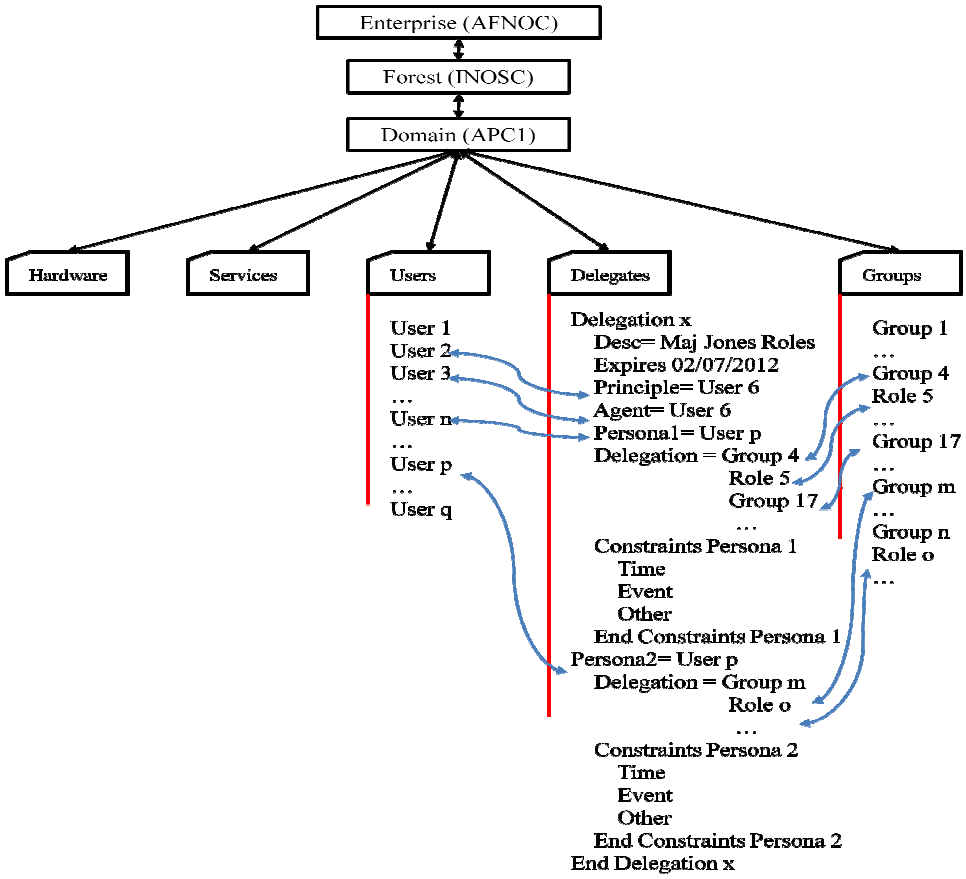

Figure A-2 Principal-Principal Delegation Architecture

A.3 NAMING FOR PERSONA

Delegate personae will be named using naming criteria for users. The user will also be given an alias that appears early in the list of identity attributes. For Principal-Agent delegation this alias will be created as "OnBehalfof" added to the EDIPI of the principal. The first name under attributes will be given the "OnBehalfof" label and the last name will be the name of the principal. For other delegations the alias for persona will be the alias of the user using the persona.

A.4 Naming for Delegation Groups

It is recommended that delegation groups simply be named sequentially as shown in Figures A-1to A-3. This will provide information hiding. Release of a delegation does not have to renumber the delegation groups.

A.5 Delegation Invocation Service

As described above, no user has the authority to log in as the persona. In order for persona to be invoked, a user delegation service must be called. It is recommended that every user that has a delegation also have a flag in his/her file and the initial logon script calls the delegation service on his behalf. When a related persona is created, the attributes under the user are modified. The last entry is provided with "Delegate", as an indication for delegation services. This field may have a default of "Normal", and a created Persona may have a value "Persona". The user delegation service will examine the DIT delegation structure for the user and offer him/her the agencies recorded in the DIT. For example, User 3 may be an agent for User 2 with persona n and an agent for User 7 with persona m. Only one delegation may be made at a time. The delegation service will then change the user identity for the session to the appropriate persona for the balance of the session. Personas will not be authorized to invoke the delegation service so that no chaining of delegations is possible. Figure A-4 shows the delegation invoking process. Once the delegation is invoked, the old user is replaced by the persona (or not, if no delegation is chosen) and all access to delegation mechanisms and the old user are broken. Each action is audited as discussed in the next section.

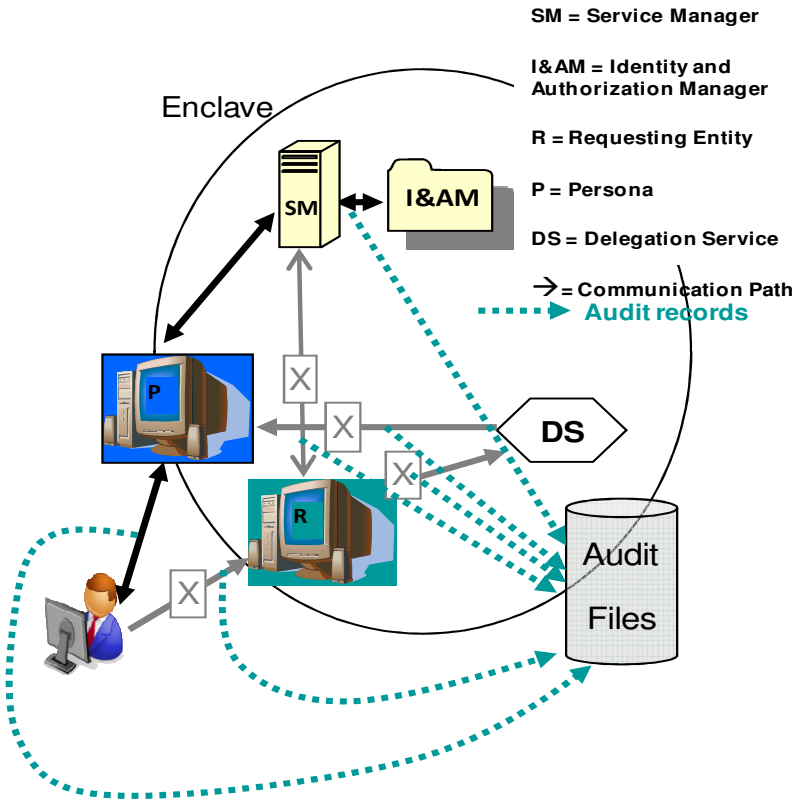

Figure A-3 Delegation Invocation Process

A.6 Importance of Audit in Delegation

There are many delegations that happen throughout a session. Most are done by impersonation (appearing to be another entity). Lower level (level 1-4) service-to-service delegations may be done by impersonation; however in every instance the session id is preserved. Tight logging must include session id so that an intrusion detection program, security analysis program, or an individual can obtain a trace of activity by session id. The session id is the tie to the invocation of delegation, which provides attribution. Audit files may reside within the enclave or elsewhere

A.7 Delegate Persona Vulnerabilities

As with any vulnerability, the final implementation, including the code developed for services will determine vulnerabilities to the system. However, several vulnerability areas come to mind.

Spoofing

No user can login as a delegate. In order to spoof the delegate persona, the spoofer would have to be an insider, or have breached the system. Since delegation is registered, the spoofer would have to create his own persona by having access to the DIT. Activating the delegate persona is logged and attribution is assigned to the user who activated the delegation.

Elevation of Rights

Recursive calls to the delegation service are prohibited. Elevation of rights during creation of the delegate persona is prohibited. The intruder (insider or external) would first have to edit the persona which would require access to the DIT and knowledge of the delegate, or creation of a new delegate.

A.8 Delegation Use Cases and Services

Tables A-1 and A-2 list the key use cases that must be implemented to provide delegation registration and delegation invocation services. These capabilities may form one basis for developing new standards for delegation (e.g., a new WS-* standard). Table 3 identifies key services that must be built to support these use cases.

Table A-1 Delegation Registration Use Cases

Function	User Role	Interface Notes
Invoke Registration authority	Invoke Service	User Identity Details and authorities
Identify Delegation Agent Principal-Agent Delegation	Any Potential Authorized User	Must be able to read delegation policy, and access DIT. Must screen delegation pair and limit choices.
Identify Delegation Agent Principal-Principle Delegation	Administrator	Must be able to read delegation policy, and access DIT. Must screen delegation pair and limit choices.
Identify Delegation Agent Admin-Agent Delegation	Administrator	Must be able to read delegation policy, and access DIT. Must screen delegation pair and limit choices.
Identify delegation attributes	Any Potential Authorized User	Probably a choice of attributes are presented that meet policy. Otherwise choices must be screened.
Release of Delegation	User identified as principal in one or more delegations	Presentation of choices for delegate deletion. Persona is removed from registry. Expiration is also a release of delegation.

Table A-2 Delegation Invocation Use Cases

Function	User Role	Interface Notes
Invoke Delegation	Login script invokes Service	User Identity Details and authorities. Present delegations for the user that have been registered
Chose delegation for session	Any Potential Authorized User	Must be able to read delegation policy, and access DIT. Must redirect user to persona and break all links with prior user.
End Delegation	Any Persona	Terminate session only.

Table A-3 Delegation Invocation Services Needed

Service	Level for Service	Other Services Needed
Set up Delegation Service	Admin	Provide rules and linkages to delegation services, update rules as policy changes.
Create Delegation	Any Potential Authorized User	User Identity Details and authorities. Present delegations for the user that have been registered
Delete Delegation	Any Principal for Principal-Agent delegations, others require admin authority	Must be able to read delegation policy, and access DIT. Must be able to eliminate persona.
Invoke Delegation	Any Potential user flagged in login script	Must be able to read delegation policy, and access DIT. Must redirect user to persona and break all links with prior user.

A.8.1 Notes and Assumptions

The following assumptions about delegation are made

- The delegate persona is persistent, but with expiration dates so that it must be renewed. This reduces instances of unintended access to the system by unauthorized users.
- Only one delegation is allowed per session
- The only way to end delegation is to terminate the session. This simplifies the user experience and the implementation of delegation.
- Audit logging is verbose (every transaction of relevance is recorded) during delegation process.
- Session ID is a key element of every audit record. This enables the audit process to determine accountability, since session ID is tied to the persona.

A.9 Conclusion

We have presented a framework for improving delegation involving personas. This framework provides greater flexibility, usability, and accountability for the delegation process, with a minimum of additional infrastructure and services required. We are currently vetting this solution with the larger Air Force community, and believe that it has great promise for improving the practice of delegation and accountability throughout the enterprise.

A.10 References to Annex A

A-1. Liu, Ranganathan, Riabov, "Specifying and Enforcing High-Level Semantic Obligation Policies," *policy*, pp. 119-128, Eighth IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY'07), 2007

A-2. Jacques Wainer , Akhil Kumar , Paulo Barthelmess, DW-RBAC: A formal security model of delegation

A-3. and revocation in workflow systems, Information Systems, v.32 n.3, p.365-384, May, 2007

A-4. He Wang , Sylvia L. Osborn, Delegation in the role graph model, Proceedings of the eleventh ACM symposium on Access control models and technologies, June 07-09, 2006, Lake Tahoe, California, USA

A-5. James B. D. Joshi , Elisa Bertino, Fine-grained role-based delegation in presence of the hybrid role hierarchy, Proceedings of the eleventh ACM symposium on Access control models and technologies, June 07-09, 2006, Lake Tahoe, California, USA

A-6. Jacques Wainer , Akhil Kumar, A fine-grained, controllable, user-to-user delegation method in RBAC, Proceedings of the tenth ACM symposium on Access control models and technologies, June 01-03, 2005, Stockholm, Sweden

A-7. Roberto Tamassia , Danfeng Yao , William H. Winsborough, Role-based cascaded delegation, Proceedings of the ninth ACM symposium on Access control models and technologies, June 02-04, 2004, Yorktown Heights, New York, USA

A-8. Jacques Wainer, Paulo Barthelmess, and Akhil Kumar. WRBAC - a workflow security model incorporating controlled overriding of constraints. International Journal of Cooperative Information Systems, 12(4):455--486, 2003.

A-9. Walt Yao. Fidelis: A policy-driven trust management framework. In Trust Management, First International Conference, iTrust, volume 2692 of Lecture Notes in Computer Science, pages 301--317. Springer, 2003.

A-10. Longhua Zhang , Gail-Joon Ahn , Bei-Tseng Chu, A rule-based framework for role-based delegation and revocation, ACM Transactions on Information and System Security (TISSEC), v.6 n.3, p.404-441, August 2003

A-11. Xinwen Zhang , Sejong Oh , Ravi Sandhu, PBDM: a flexible delegation model in RBAC, Proceedings of the eighth ACM symposium on Access control models and technologies, June 02-03, 2003, Como, Italy

A-12. JongSoon Park, YoungLok Lee, HyungHyo Lee, and BongNam Noh. A role-based delegation model using role hierarchy supporting restricted permission inheritance. In Proceedings of the International Conference on Security and Management, SAM '03, pages 294--302. CSREA Press, 2003.

A-13. Chun Ruan , Vijay Varadharajan, Resolving Conflicts in Authorization Delegations, Proceedings of the 7th Australian Conference on Information Security and Privacy, p.271-285, July 03-05, 2002

A-14. Jean Bacon , Ken Moody , Walt Yao, A model of OASIS role-based access control and its support for active security, ACM Transactions on Information and System Security (TISSEC), v.5 n.4, p.492-540, November 2002

A-15. Vijayalakshmi Atluri , Avigdor Gal, An authorization model for temporal and derived data: securing information portals, ACM Transactions on Information and System Security (TISSEC), v.5 n.1, p.62-94, February 2002

A-16. Åsa Hagström , Sushil Jajodia , Francesco Parisi-Presicce, Duminda Wijesekera, Revocations-A Classification, Proceedings of the 14th IEEE Workshop on Computer Security Foundations, p.44, June 11-13, 2001

A-17. Evgeny Dantsin , Thomas Eiter , Georg Gottlob , Andrei Voronkov, Complexity and expressive power of logic programming, ACM Computing Surveys (CSUR), v.33 n.3, p.374-425, September 2001

A-18. E. Barka , R. Sandhu, Framework for role-based delegation models, Proceedings of the 16th Annual Ezedin S. Barka and Ravi Sandhu. A role-based delegation model and some extensions. In 23rd National Information Systems Security Conference, October 2000. http://csrc.nist.gov/nissc/2000/proceedings/papers/021.pdf.

A-19. Computer Security Applications Conference, p.168, December 11-15, 2000

A-20. Ravi Sandhu , Qamar Munawer, The ARBAC99 Model for Administration of Roles, Proceedings of the 15th Annual Computer Security Applications Conference, p.229, December 06-10, 1999

A-21. Cheh Goh , Adrian Baldwin, Towards a more complete model of role, Proceedings of the third ACM workshop on Role-based access control, p.55-62, October 22-23, 1998, Fairfax, Virginia, United States

A-22. Ravi S. Sandhu , Edward J. Coyne , Hal L. Feinstein , Charles E. Youman, Role-Based Access Control Models, Computer, v.29 n.2, p.38-47, February 1996

A-23. Ronald Fagin, On an authorization mechanism, ACM Transactions on Database Systems (TODS), v.3 n.3, p.310-319, Sept. 1978

A-24. Patricia P. Griffiths , Bradford W. Wade, An authorization mechanism for a relational database system, ACM Transactions on Database Systems (TODS), v.1 n.3, p.242-255, Sept. 1976