Delegation, Attribution and Least Privilege

by
C. Chandersekaran, the Institute for Defense Analyses (IDA)
William R Simpson, the Institute for Defense Analyses (IDA)

Abstract

The Air Force has been mandated to share information among authorized users both with the enterprise and across enterprise where a need and agreement are established -- for its normal course of business. No services, delayed services, inadequate services and information flow all hinder or prevent the normal course of business. Information sharing means maintaining availability, performance, integrity, and reliability.

- Availability – covers the traditional aspects of being there when needed, but in this information sharing environment it also means discovery and accessibility. The later derives from the MDE environment being built into the IIB and a great deal of monitoring will be devoted to MDE services.
- Performance – information delayed is information denied. Excessive latency will not be tolerated and the need to share has a time component as well as an authoritative and currency component. Authority and Currency are dealt with by the COIs and the performance aspects will be monitored by the network. Performance includes latency, bottlenecks, and saturation.
- Integrity – covers the correct handling, tamper resistance and awareness, authorization and authentication. Most of these elements are dealt with by the IA architecture. For Federation, this step is particularly important.
- Reliability – covers the ability to complete delivery of information, fail-over, coop and backup of critical information. Many of these are hardware related and dealt with through hardware monitoring and redundancy of hardware and software and data. Fail-over may be a management function when provisions are made for state tracking and re-direct when failures of hardware and software occur.

Delegation, Attribution and Least Privilege are an implicit part of this information sharing. 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 a user mode solutions when given the choice between a kernel mode and user mode solution if the two solutions provide the same results. Discussions in this paper will be restricted to OSI model levels four and above.

Delegation, Least Privilege and Attribution Person to Person

Delegation is the handing of a task over to another person, usually (although not limited to) 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 another 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. Delegation does not include normal job function as described in the organizational chart and manifest by attributes roles and groups to effectively carry out the responsibilities of the job. The Executive Officer acting on behalf of his Commander may be part of his job description and is not a delegation issue. Delegation is ceding one or more of those job functions to another

person. Delegation is not transitive. If the General asks the Colonel to check his e-mail, the Colonel cannot ask the lieutenant to check the General's e-mail. The Colonel may suggest to the General that the Lieutenant do it, and the General may revoke the Colonel's delegation and give it to the Lieutenant. Delegation is personal. When the delegator moves on, is replaced, dies or retires, all delegations from that position are revoked.
Attribution is provided when the user of any privilege is identified as acting on behalf of himself or the individual who authorized the delegation.
Least Privilege is preserved by providing the agent with only that level of privilege necessary to do the task without exceeding his/her own authority.

Delegation, Least Privilege and Attribution Person and Service to Service

Delegation is implicit when invoking a service. In the Air Force enterprise an individual is assumed to delegate to a service the right to act upon its behalf. 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. Delegation only lives during the session under consideration.
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 agent with only that level of privilege necessary to do the task without exceeding his/her own authority.

Raised Authorization on Personal Initiative and Responsibility

This is not a delegation issue and must involve an administrator to raise such privilege. For IT purposes, the individual must provide rationale to the administrator of IT services which will be duly logged when authorized and approved for attribution purposes.

Purpose

This paper will define the elements and process required for delegation, attribution and least privilege. The Air Force Enterprise Architecture provided in the reference[1] (not available to all) is assumed, particularly the use of a Security Token Server, credentialing of all active entities, and the use of SAML 2.0 for authorization.

Since the processes for services and people are inherently different for delegation, they will be discussed separately in the following sections.

Air Force Enterprise Vision

Top Level Tenets

Any service management solution for the enterprise (and indeed, any solution for any component of the enterprise) should be tested against a set of fundamental evaluation criteria or tenets. These tenets are separate from the "functional requirements" of a specific component (e.g., access control needs to be defined); they relate more to the attributes of the solution that make it able to be implemented, extensible, cost-effective, and supportive of the fundamental objectives of the enterprise. Our proposed top-level tenets are the following:

[1] Air Force Information Assurance Enterprise Architecture, Version 1.25, SAF/XC, 11 April 2008.

- The **zeroth** tenet is that the ***enemy is embedded***. Current threat evaluation indicates that at the unclassified and NIPR level, attacks are often successful, and discovery and ferreting out the results of these attacks is difficult and problematic at best. In many cases attackers may get inside of the exploit discovery and patch loop. In others, successful Phishing and Spear Phishing attacks have been launched. Rogue agents may be present and to the extent possible, we should be able to operate in their presence, although not exclude their ability to view some activity. The tenets below together with the architecture embody this approach.
- 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. However, at a certain point (usually a lower point than you would suspect), these added features come at the cost of greater complexity, less understandability, greater difficulty in administration, higher cost, and/or lower adoption rates that are unacceptable to the organization. Therefore, simplicity absolutely must be a primary goal of any access solution. Supporting cross-enclave and enterprise scenarios will automatically add a certain degree of complexity that will be challenging enough to handle in any case. Extension to coalition adds yet another level of complexity. 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 forest 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. For example, a user of a service needs to know the input parameters required to call it, and the output it gets in return. It does not need to know the algorithms or internal variables the service uses to implement the capability. Hiding this information keeps these details secret from the consumer of the capability, makes it harder to exploit and increases implementation flexibility. Any information that is not shielded from inadvertent discovery may be used in later attacks.
- The **fourth** tenet is ***accountability***. In this context, accountability means being able to definitively identify and track what entity in the enterprise performed any particular operation (e.g. accessed a file or IP address, invoked a service). To enable accountability, it is necessary to prohibit online "impersonation", in which principals share their credentials with another actor rather than delegating their authority. Without a delegation model, it is impossible to establish a chain of custody or do effective forensic analysis to investigate security incidents.
- This **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, COTS products that are as open as possible will be emphasized and should produce cost efficiencies.
- The **seventh** and final tenet is that *lines of authority* should be preserved and IA decisions should be made by policy and/or agreement at the appropriate level.

A Persona-Based Framework for User Based Delegation and Least Privilege

The Need for Delegation in IT Systems

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[2].

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. Delegation does not include normal job function as described in the organizational chart and manifest by attributes roles and groups to effectively carry out the responsibilities of the job. The Executive Officer acting on behalf of his Commander may be part of his job description and is not a delegation issue. Delegation is ceding one or more of those job functions to another person. Delegation is not transitive. If the General asks the Colonel to check his e-mail, the Colonel cannot ask the lieutenant to check the General's e-mail. The Colonel may suggest to the General that the Lieutenant do it, and the General may revoke the Colonel's delegation and give it to the Lieutenant. Delegation is personal. When the delegator moves on, is replaced, dies or retires, all delegations from that position are revoked.

- 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.
- 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.
- Initial examination of virtual machine architectures indicate that the use of personae for designation of these systems may assist in attribution and forensics. The details have not been worked out as of this time.

---

[2] Definition adapted from Wikipedia.

- 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.

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 taking on of the persona constitutes an acceptance of the delegation. 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 from 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 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 expires automatically when the delegator changes organizational position or otherwise leaves. A link must be established between the delegator and the delegation so that they are modified simultaneously. 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".

Architectural Details

The certificate authority must use known and registered (or in specific cases defined) certificate revocation and currency checking software.

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). The figure below 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, delegation constraints, PKI and other credentials. In order for this service to work, the semantics of policy must be worked out by the COI. 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 roles and groups that are allowed delegation. General attributes such as rank, or clearance cannot be delegated. The latter is important because a number of rules will

be invoked. In the absence of offered roles/groups, the individual specified roles/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 roles and 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.

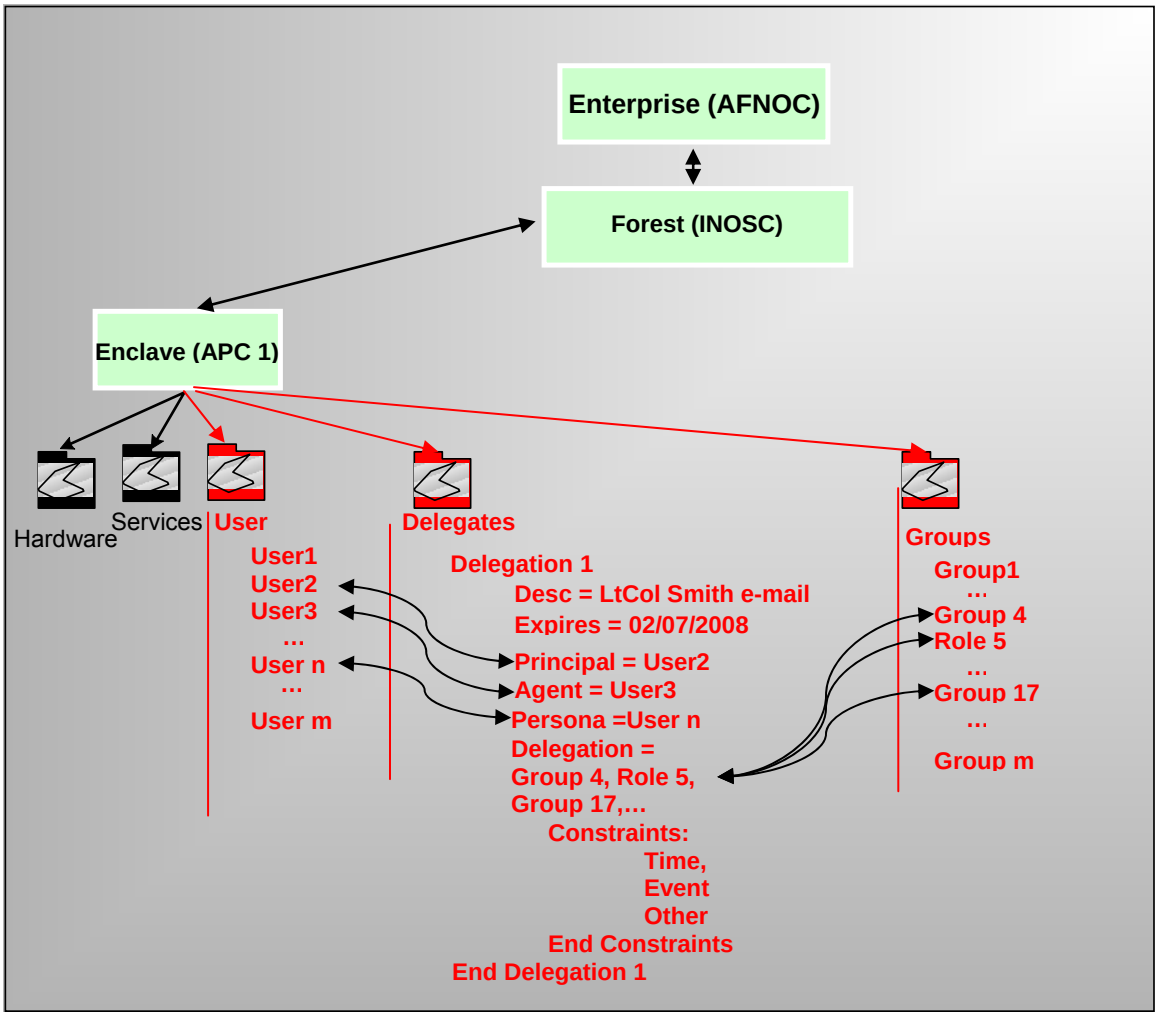

Figure 1 Principal-Agent Delegation Architecture – Registration

Least Privilege as a Principal-Principal Delegation

User Based Least Privilege[3]

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.

[3] Definition adapted from Wikipedia.

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 a user mode solutions when given the choice between a kernel mode and user mode solution if the two solutions provide the same results.

The Need for a User Based Least Privilege in IT Systems

The need for a user based least privilege in IT systems often arises out of the need to manage the extent of activity that can take place, or the cost of errors by the human operator. This happens when, for example, a user requests a catastrophic action and acknowledges the request without mulling over the consequences (formatting a hard drive, for one). Under these circumstances the user will be left to establish the least privilege to accomplish the task. Least privilege also applies to service to service interactions and is not dealt with here.

Registration Service for Least Privilege 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 Directory Information Tree (DIT). The figure below shows the delegation registration process. The delegation registration service is invoked by either user 6 or the enclave[4] 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 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 and/or groups 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, q, r), 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 roles, 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 or roles, the individual specified groups must be heavily screened for overall and specific policy. Finally, the delegate personae (users p, q, r) are populated with access attributes, roles and 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,q,r are the credentials associated with the original identity in self-designation (user 6).

[4] An enclave is defined as a set of capabilities realized by hardware, software, networks, devices, and people.

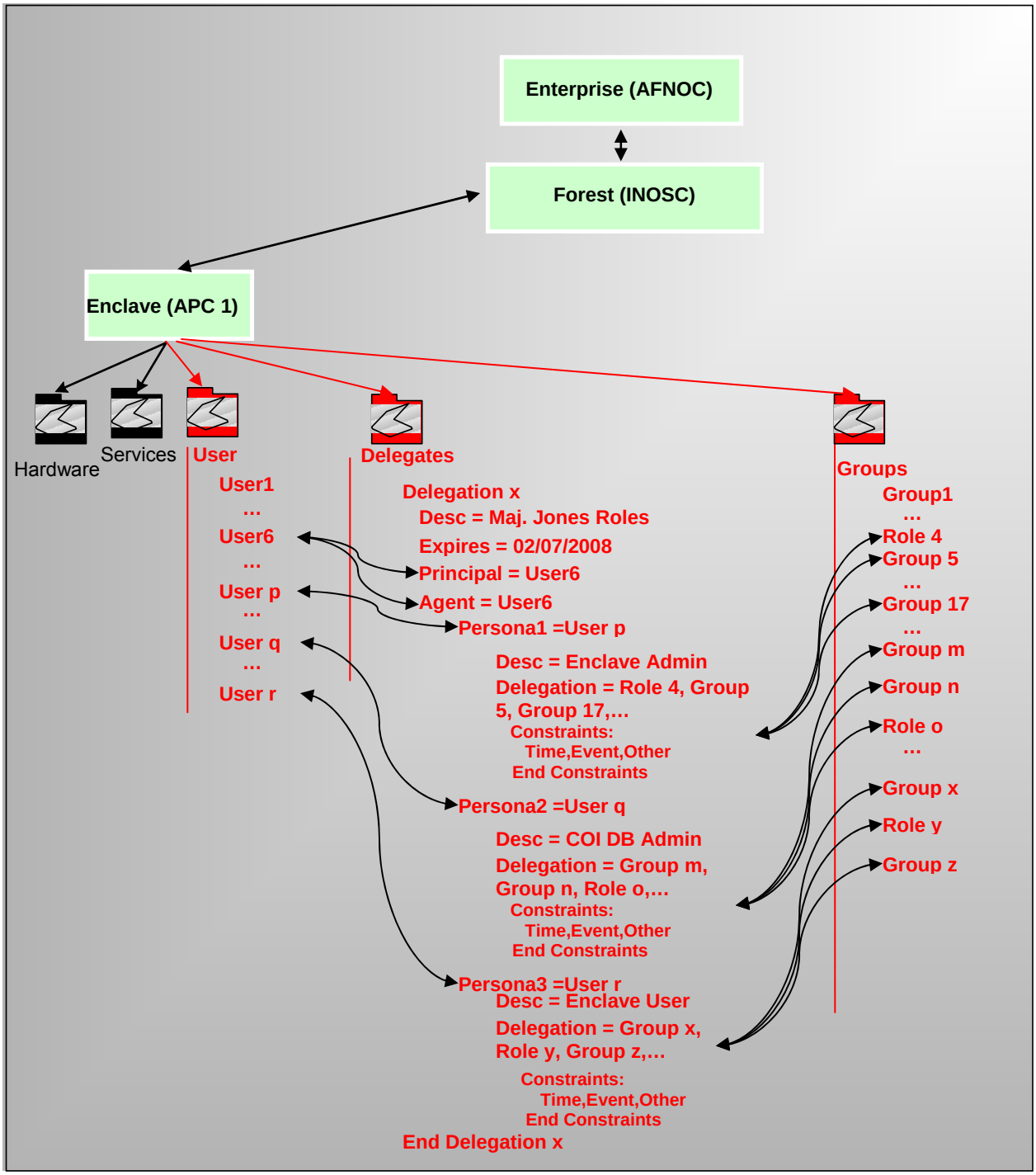

Figure 2 Pricipal-Principal Delegation Architecture – Registration

Registration Service for Admin-Principal Delegation

Admin-Principal policies are promulgated by the appropriate authority. Such policies may apply to a large class of individuals (as in the movement of a group of individuals between assignments) or to a specific instance (as in the movement of an individual between assignments). The admin-principal delegation registration creates a user persona for the old assignment with an appropriately short expiration and a second persona that is the new assignment of a longer expiration. This process involves three branches of the Directory Information Tree (DIT). The figure below shows the

delegation registration process. The delegation registration service is invoked and current policy is checked to see if User 8 can be provided two identities. The registration authority creates the persona (user z), together with names and PKI and other credentials associated with the old assignment. In order for this service to work, the semantics of policy must be worked out by the COI. 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. The principle constraint is time, but a constraint model may be added as in other delegations. At this point, the administrator is offered roles and groups that are allowed for the new assignment. The latter is important because a number of rules will be invoked. In the absence of offered roles and groups, the individually specified groups must be heavily screened for overall and specific policy such as no delegation of clearances. Finally, the original user designation (user 8) is populated with access groups from the new assignment and the user's attributes. The new persona is temporary and appears in the DIT as any other user. User credentials associated with user z are the credentials associated with an old assignment and identity.

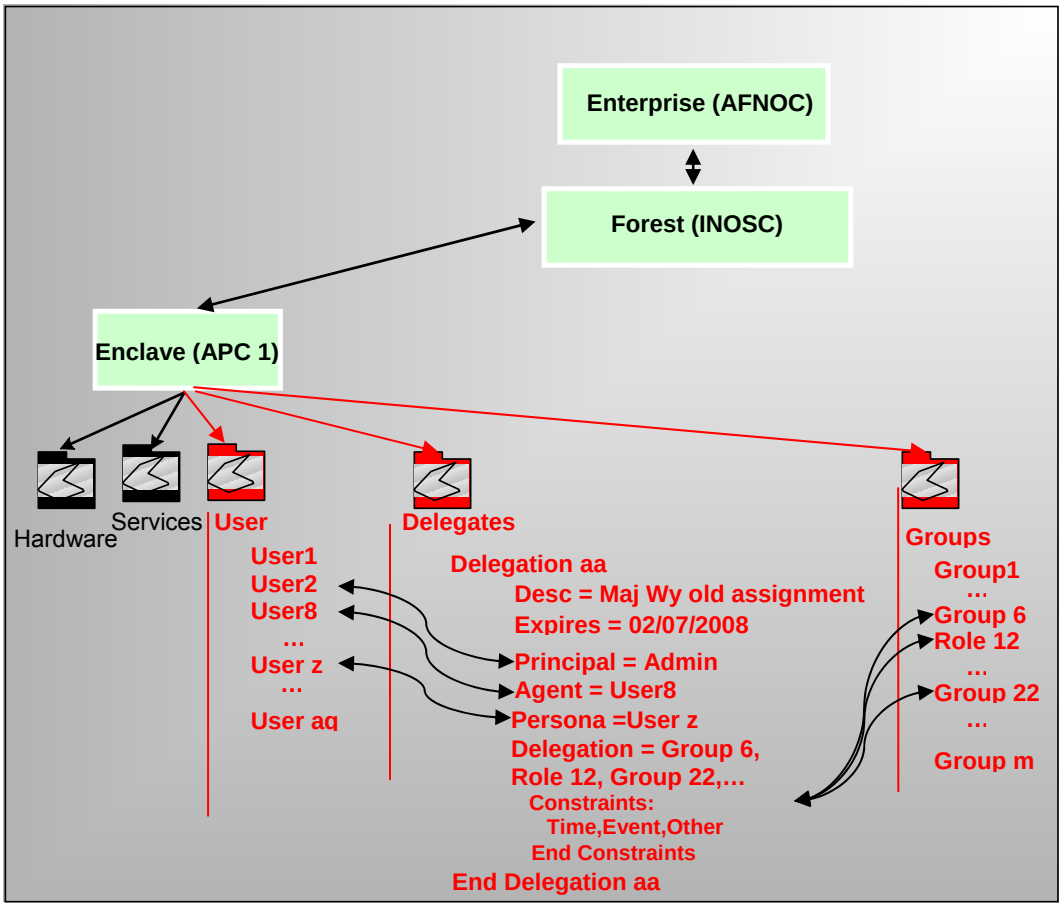

Figure 3 Admin-Principal Delegation Architecture – Registration

Naming for Persona

Delegate personae will be named using naming criteria for users. The user will also be given an common name, or an alias for the common name 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 common name or alias for the persona will be the agent EDIPI appended to

"OnBehalfof" label appended to the EDIPI of the principal. For other delegations the alias for persona will be the alias of the user using the persona.

Naming for Delegation Elements

It is recommended that delegation elements simply be named sequentially as shown in the above figure. This will provide information hiding. Release of a delegation should not renumber the delegation groups.

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 id 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. The figure below 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.

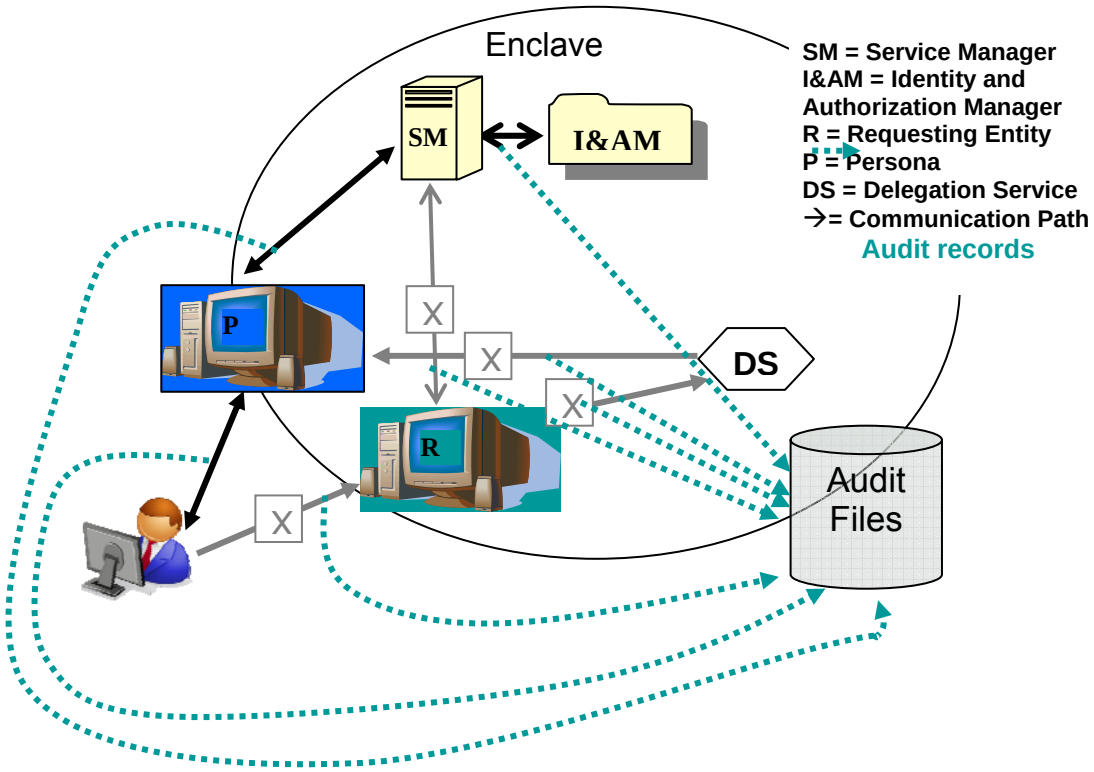

Figure 4 Delegation Invocation Process

The 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 provides attribution. Audit files may reside within the enclave or elsewhere.

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.

Delegation Use Cases and Services

Tables 1 list the key elements that derive from the registration use case 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 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.

Tables 2 list the key elements that derive from the invocation use case 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 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 3 identifies key services that must be built to support these use cases.

Table 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.

Delegation, Attribution and Least Privilege when Services are Involved

Delegations where services are involved are treated separately and subject to the following assumptions:

- User based requests:
  - A request for service within the AF enterprise is an *implicit* request to a service provider to do what you are allowed to on my behalf to satisfy this request.
  - Group/Role definition is fine grained enough to signify access throughout the process.
- Service based requests:
  - A request for service within the AF enterprise is an *explicit* request to a downstream service provider to do what you are allowed to on my behalf to satisfy this request.

- Group/Role definition is fine grained enough to signify access throughout the process.
- Non-aggregation services are atomic.

- Other
  - Only considering web-service calls at or above OSI level 5.
  - Calls below level 5 on the OSI stack are not made by SAML authorization and do not follow this paradigm.

Basic Use Case

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

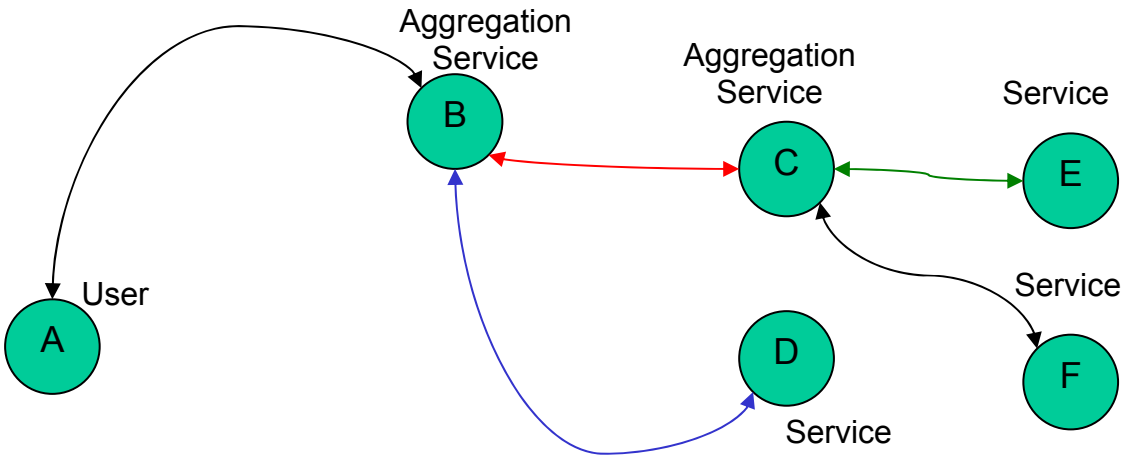

Figure 5 Basic Use Case for Service Delegation, Attribution and Least Privilege

Standard Communication for Authentication / Authorization

Each communication link in the figure above will be authenticated end-to-end with the X.509 certificates provided for each of the active entities. 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 below:

Table 4 SAML 2.0 Format for User to Aggregation Service.

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**

Item	Field Usage	Recommendation	Notes
Attribute Assertion			
Subject	**Yes For User A**	**Edipi**	**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**

Pruning Attributes[5], Groups and Roles

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.

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 figures 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.

Data Requirements for the Pruning of 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. The table below presents such a data compilation. This table will be used in the subsequent example. An element is either an attribute, role or group used in the authorization decision.

[5] 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 5 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**	**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 combing of these elements for least privilege is given 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: $N_{i+1} = (P_i \cap (R_{i+1} \cup H_i)) \cup (E_i \cap R_{i+1})$
Where: $\cap$ is the intersection of sets and
$\cup$ is the union of sets
Ø is the empty set (no members)

The formula may be read as the common elements in the prior SAML and the union of the held elements and those required by the next call (($P_i \cap (R_{i+1} \cup H_i)$) - 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.

Subsequent Calls Require the Saving of 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 below:

Table 6 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**

SAML Token Modifications for Further Calls

The Attribute Assertion of Table 5 is returned to the STS for modification of the normal SAML token. The SAML Token for the unmodified service call is given below:

Table 7 Unmodified SAML for Service B of Use Case

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 or equivalent**
Attribute Assertion			
Subject	**Yes For Service B**	**Cn for Service B**	**For Attribution**
Attributes, Group and Role Memberships	**Yes For Service B**	**Required**	$N_{i+1} = (P_i \cap (R_{i+1} \cup H_i)) \cup (E_i \cap R_{i+1})$
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 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 below:

Table 8 Modified SAML Attribute Assertion 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 or equivalent**
Attribute Assertion			
Subject	**Yes contains A and B**	**Cn B OnBehalfOf edipi**	**For Attribution**
Attributes, Group and Role Memberships	**Yes B restricted by A**	**Required**	$N_{i+1} = (P_i \cap (R_{i+1} \cup H_i)) \cup (E_i \cap R_{i+1})$
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.

An Annotated Notional Example

A User in the AFNETOPS 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:

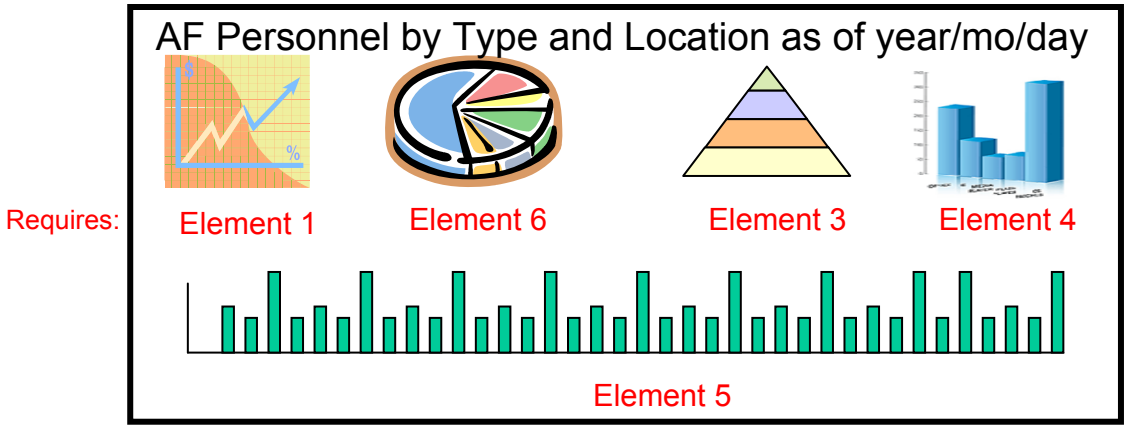

Figure 6 AFPeronnel30 with Display Outputs and Element Requirements

The elements required would not typically be displayed. A partial calling tree for AFPersonnel30 is provided in Figure 11. 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 5, 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.

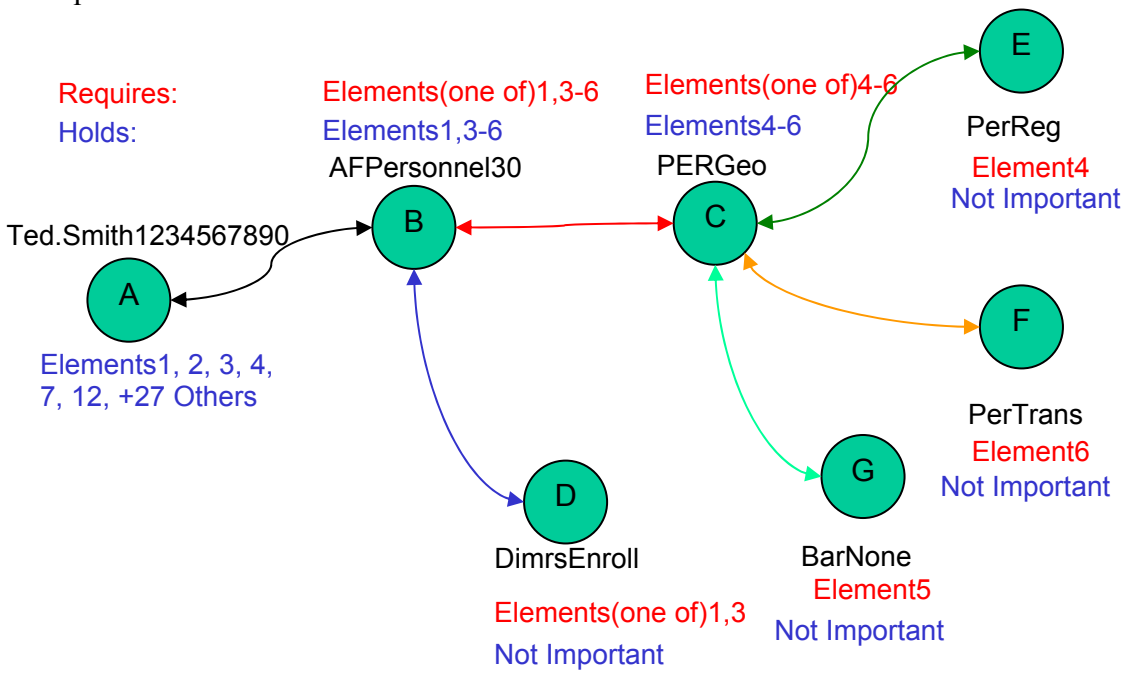

Figure 7 AFPersonnel30 Calling Tree

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 below:

Table 9 Ted Smith SAML Push to AFPersonnel30

Item	Field Usage
SAML Response	
Version ID	**Version 2.0**
ID	**0qwdrt009kkmn**
Issue Instant	**080820081943**
Issuer	**AFNETOPS 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**[6] $N_1 = (P_1 \cap (R_2 \cup H_2)) \cup (E_1 \cap R_2)$ $=((1, 2, 3, 4, 7, 12, +27) \cap ((1,3\text{-}6) \cup (1,3\text{-}6)) \cup (\emptyset \cap 1,3\text{-}6)$ $=((1, 2, 3, 4, 7, 12, +27) \cap ((1,3\text{-}6)) \cup (\emptyset)$ $=((1, 3, 4))$
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 10.

Table 10 AFPersonnel30 SAML Push to PERGeo

Item	Field Usage
SAML Response	
Version ID	**Version 2.0**
ID	**0qwdrt009kkmn**
Issue Instant	**080820081944**
Issuer	**AFNETOPS 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**

[6] An element is an attribute, role, group or combination of the previous. Elimination of Element 2, Element 7, Element 12 and 27 other elements based on pruning (see Table 5 under AFPersonnel30)

Item	Field Usage
Group and Role Memberships	**Group4[7], Element6[8]** $N_{i+1} = (P_i \cap (R_{i+1} \cup H_i)) \cup (E_i \cap R_{i+1})$ $=((1, 3, 4) \cap (4 \cup 4\text{-}6) \cup (6 \cap 4\text{-}6)$ $=((1, 3, 4) \cap (4)) \cup (6)$ $=(4,6)$
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 12 shows the completion of the calling tree, including only the SAML Attribute Assertions in the blocks below.

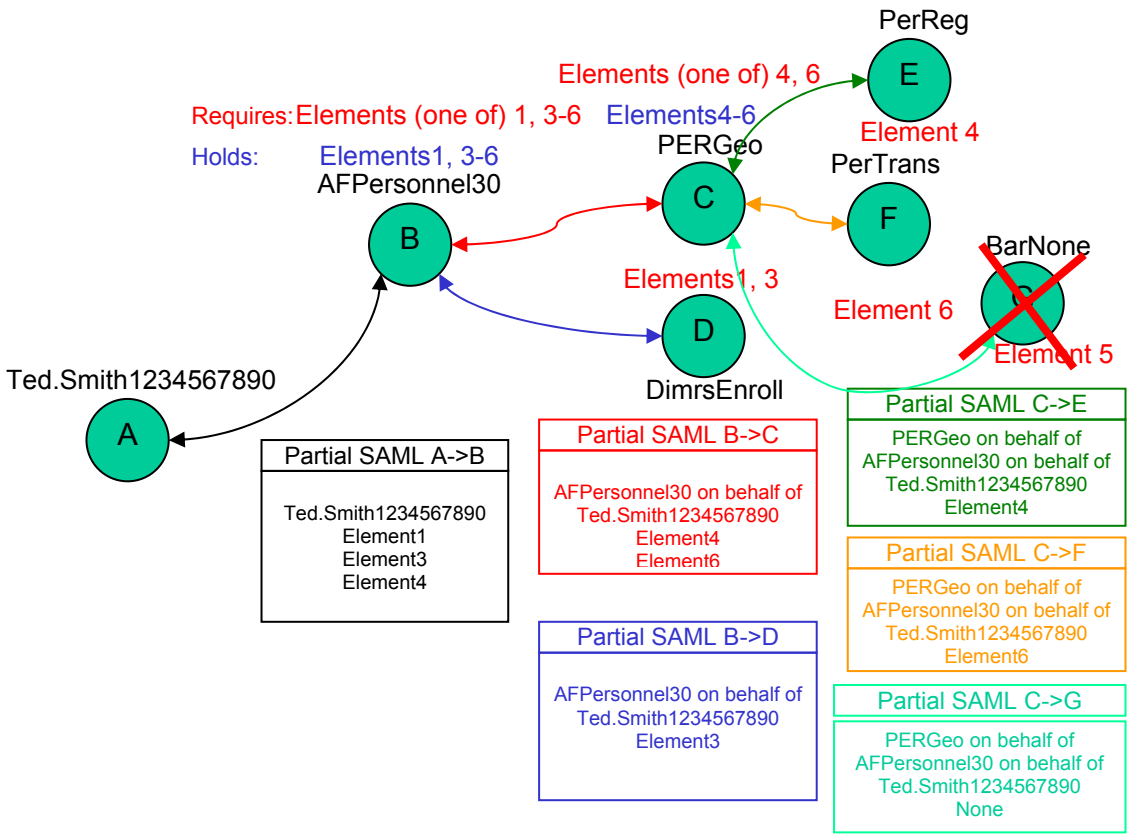

Figure 8 SAML Attribute Assertion for the Calling Tree

[7] 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)

[8] Element 6 is a required escalation element.

Note that the calls to BarNone fails access and while being stealth to the calling routine (which will return with no data after timeout) 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 red annotations) is presented in Figure 13. Note that Element 6 privilege was provided by service escalation.

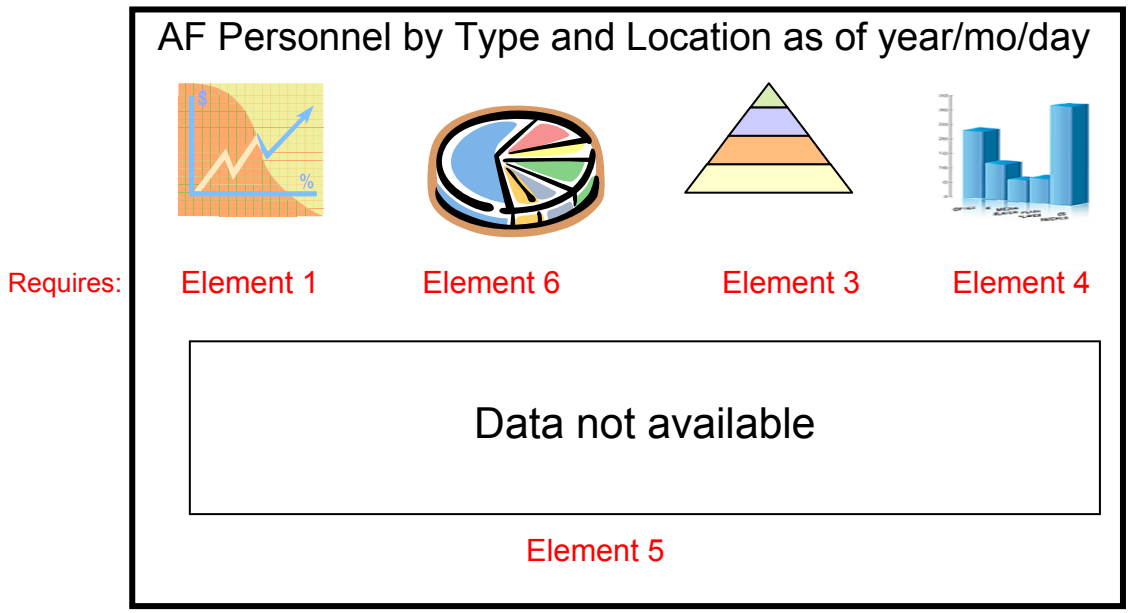

Figure 9 Dashboard Service AFPersonnel30 Case Result (with Annotation)

Additional Requirements on the STS and Services

The STS requirements are given below:

Table 11 STS Additional Requirements

Item	Requirement	Data Structure Required
Element Pruning by individual service call	Least Privilege reduction of Attributes, Groups and Roles in SAML Assertion	Yes, table of service attribute, group and role requirements for access. Must be synchronized with access managers.
Receive prior SAML Assertion	Need subject, attributes, groups and roles for further attribution and group definition	Internal only no external store required.
Apply prior SAML assertion to SAML	Includes modification of subject line in assertion as well as further pruning of elements	Internal only no external store required.

The additional requirements on the Services are given below:

Table 12 Service Additional Requirements

Item	Requirement	Data Structure Required
Hold SAML Assertion	Required only when subsequent service calls are to be performed on behalf of the requestor	Internal only no external store required, but must be held on a per thread basis
Send Prior SAML Assertion	When subsequent service calls are made.	Internal only no external store required, but must be transmitted on a per thread basis
Use Subject of SAML Assertion in Logs	Attribution Requirement	Log files in existence
Purge held SAML Assertion	When thread is complete.	none

Service Use Case Summary

The process of using SAML token modification for tracking of delegation, attribution and least privileges has both advantages and disadvantages.

- Advantages
  - Use of SAML standard without extension or violation
  - Full attribution for data analyses and forensics.
  - Least privilege is invoked on service to service calls
  - Aggregation service does not need to filter response to user based on access credentials
  - Federation works exactly the same way
  - Peron-to-Person delegation compatible
- Disadvantages
  - Use of SAML standard in an way that SAML standard writers did not anticipate
  - Service must store and covey SAML assertion invoking the thread
  - STS currently does not process this data
  -

Summary

This paper has presented two areas where delegation, attribution and least privilege are required. The first case involves human interactions where agents are appointed to perform certain tasks on behalf of a principle. The second is for the delegation and attribution that occurs in service aggregation. Both are approached differently, but fit within the AF IA Architecture. In fact, they dove-tail nicely with each other. The process began with Ted.Smith1234567890, but could have just as easily begun with Jack.Jones1234565432 OnBehalfOf Ted.Smith1234567890 (where Jack has been delegated to do the Air Force Personnel work for Ted). Further, the process works unchanged when applied to federation[9] with remapped identities and groups.

[9] Air Force Information Assurance Strategy Team, *Federation,* Version 0.5, SAF/XC, 5 August 2008.

References

1. Air Force Information Assurance Strategy Team, *Air Force Information Assurance Enterprise Architecture*, Version 1.25, SAF/XC, 11 April 2008.
2. Air Force Information Assurance Strategy Team, *Federation,* Version 0.5, SAF/XC, 5 August 2008.
3. AFPD 33-3 Information Management, AF Portal Community of Practice: AF Information & Data Management Strategy – Implementation (Policy) https://wwwd.my.af.mil/afknprod/ASPs/CoP/OpenCoP.asp?Filter=OO-SC-AF-DM or http://www.e-publishing.af.mil/
4. COI Coordination Panel Charter, AF Portal Community of Practice: AF Information & Data Management Strategy – Implementation (COI Primer) https://wwwd.my.af.mil/afknprod/ASPs/CoP/OpenCoP.asp?Filter=OO-SC-AF-DM
5. COI Primer, AF Portal Community of Practice: AF Information & Data Management Strategy – Implementation (COI Primer) https://wwwd.my.af.mil/afknprod/ASPs/CoP/OpenCoP.asp?Filter=OO-SC-AF-DM
6. DoD Directive 8320.2 "Data Sharing in a Net-Centric Department of Defense" and DOD Guidance 8320.2-G "Guidance for Implementing Net-Centric Data Sharing", AF Portal Community of Practice: AF Information & Data Management Strategy – Implementation (Policy) https://wwwd.my.af.mil/afknprod/ASPs/CoP/OpenCoP.asp?Filter=OO-SC-AF-DM
7. Metadata Concept, AF Portal Community of Practice: AF Information & Data Management Strategy – Implementation (Metadata) https://wwwd.my.af.mil/afknprod/ASPs/CoP/OpenCoP.asp?Filter=OO-SC-AF-DM
8. Transparency Integrated Product Team (TIPT) information and proceedings AF Portal Community of Practice https://wwwd.my.af.mil/afknprod/ASPs/CoP/OpenCoP.asp?Filter=OO-TR-AF-39
9. Air Force Instruction (AFI) 31-501, Personnel Security Program Management
10. AFI 33-115, Network Management and Licensing Network Users and Certifying Network Professionals
11. AFI 33-119, Electronic Mail (E-mail) Management and Use
12. AFI 33-202, Computer Security
13. AFI 33-204, Information Protection Security Awareness, Training, and Education (SATE)
14. AFMAN 33-223, Identification and Authentication
15. AFMC Supplement 1, AFMAN 33-223, Identification and Authentication
16. CJCSI 3170.01E, Joint Capabilities Integration and Development System
17. CJCSI 6212.01D, Interoperability and Supportability of Information Technology and National Security Systems
18. DoDD 5000.1, The Defense Acquisition System
19. DoDD 4630.5, Interoperability and Supportability of Information Technology and National Security Systems
20. DoDD 8000.1, Management of DoD Information Resources and Information Technology

21. DoDD 8115.01, “DoD Information Technology Portfolio Management,” October 10, 2005
22. DoDD 8115.1, Information Technology Portfolio Management
23. DoDD 8500.1, lnformation Assurance (IA), 24 OCT 02
24. DoDD 8530.1, Computer Network Defense (CND), 8 Jan 2001
25. DoDI 4630.8, Procedures for Interoperability and Supportability of Information Technology and National Security Systems
26. DoDI 5000.2, Operation of the Defense Acquisition System
27. DoDI 8500.2, Information Assurance Implementation, 6 FEB 03
28. DoDI 8520.2, Public Key Infrastructure (PKD and Public Key (PK) Enabling, 1 APR 04
29. DoDI 8115.02, "Information Technology Portfolio Management Implementation", October 30, 2006
30. JTF-GNO CTO 06-02, Tasks for Phase I of PKl Implementation, 17 JAN 06
31. DoD/CIO Memo, Approval of the Alternate Logon Token, 14 AUG 06
32. JTF-GNO WARNORD 07-37, Public Key Infrastructure Implementation, Phase 2, August 2007
33. The National Defense Strategy of the United States of America, March 2005
34. Department of Defense Net-Centric Data Strategy, May 9, 2003
35. Joint Concept of Operations for Global Information Grid NetOps, Version 3, August 4, 2006
36. OASIS open set of Standards (see Endnote)
37. “Guide to Secure Web Services: Recommendations of the National Institute of Standards and Technology”, NIST-US Department of Commerce Publication, August 2007.
38. “Web Service Security: Scenarios, Patterns, and Implementation Guidance for Web Services Enhancements (WSE) 3.0”, Microsoft Corporation, 2005
39. “WS-ReliableMessaging Specification”, OASIS, June 2007
40. “WS-SecureConversation Specification”, OASIS, March 2007
41. “WSE 3.0 and WS-ReliableMessaging”, Microsoft White Paper, June 2005, http://msdn2.microsoft.com/en-us/library/ms996942(d=printer).aspx
42. FIPS PUB 196, Federal Information Processing Standards Publication. “Entity Authentication Using Public Key Cryptography”, February 18, 1997