Federation for a Secure Enterprise[1]

William R. Simpson and Kevin E. Foltz
Institute for Defense Analyses, 4850 Mark Center Dr., Alexandria, Virginia 22311

[1] The publication of this paper does not indicate endorsement by the Department of Defense or IDA, nor should the contents be construed as reflecting the official position of these organizations.

Abstract:

Federated activity presents a challenge for enterprises with high-level security architectures. Federation involves information sharing among the services and with working partners, coalition partners, first responders, and other organizations. Federation may be unilateral or bilateral and with similar or dissimilar information sharing goals. Strong internal security controls often do not extend cleanly across enterprise boundaries, potentially leading to insecure shortcuts and workarounds that can become the rule instead of the exception. This paper presents methods for an enterprise to extend its strong security policies to include federation partners. It applies to federation partners that support the same security policies with compatible standards and services, and also to partners that provide a similar but incompatible security framework, a subset of required security services, or no security services. The partner organization may be fully trusted, partially trusted, or untrusted. Even in trusted partners the services may not meet required security standards. The solution presented combines selected partner security services, internal services, derived credentials, delegated authorities, and supplemental services to form the federation security architecture. This paper uses the Enterprise Level Security (ELS) architecture as the starting point for a secure enterprise and addresses the challenge of extending this model to federate with different types of partners. We review the security approach, the security properties, and several options for an enterprise to maintain the ELS security properties while enabling federated sharing with other enterprises that have different capabilities and levels of trust.

Keywords: federation, coalition, security, trust, sharing, first responders

Introduction

The Enterprise Level Security (ELS) framework is a way to provide strong security guarantees within an enterprise. It provides rules for secure authentication and authorization that are distributed, scalable, and centrally managed. Entities in the enterprise need only authentication credentials and attributes in data stores, and then access is automatically granted to appropriate applications and services throughout the enterprise. This system works within an enterprise, but often there is a requirement to share resources with different enterprises, which poses a challenge to the ELS framework.

This work discusses options for federation across enterprises in which one enterprise uses ELS. [14]-[18] The partner enterprise may use ELS, another similar security framework, or limited or no security services. The partner enterprise may be trusted, partially trusted, or untrusted. This paper provides options for an ELS enterprise to enable federated sharing with other enterprises that have different capabilities and levels of trust. This enables more functionality within the ELS framework and fewer workarounds, which provides stronger and more uniform security measures.

Throughout this paper the term “partner” will refer to the other side in a federation agreement. This could be a single individual, an entire enterprise, or anything in between.

The ELS Architecture

The Enterprise Level Security (ELS) architecture has evolved from a fortress approach, in which the threat is assumed to be stopped at the front door, to a distributed security system that eliminates or mitigates many of the primary vulnerability points inherent with that system, as shown in Figure 1.

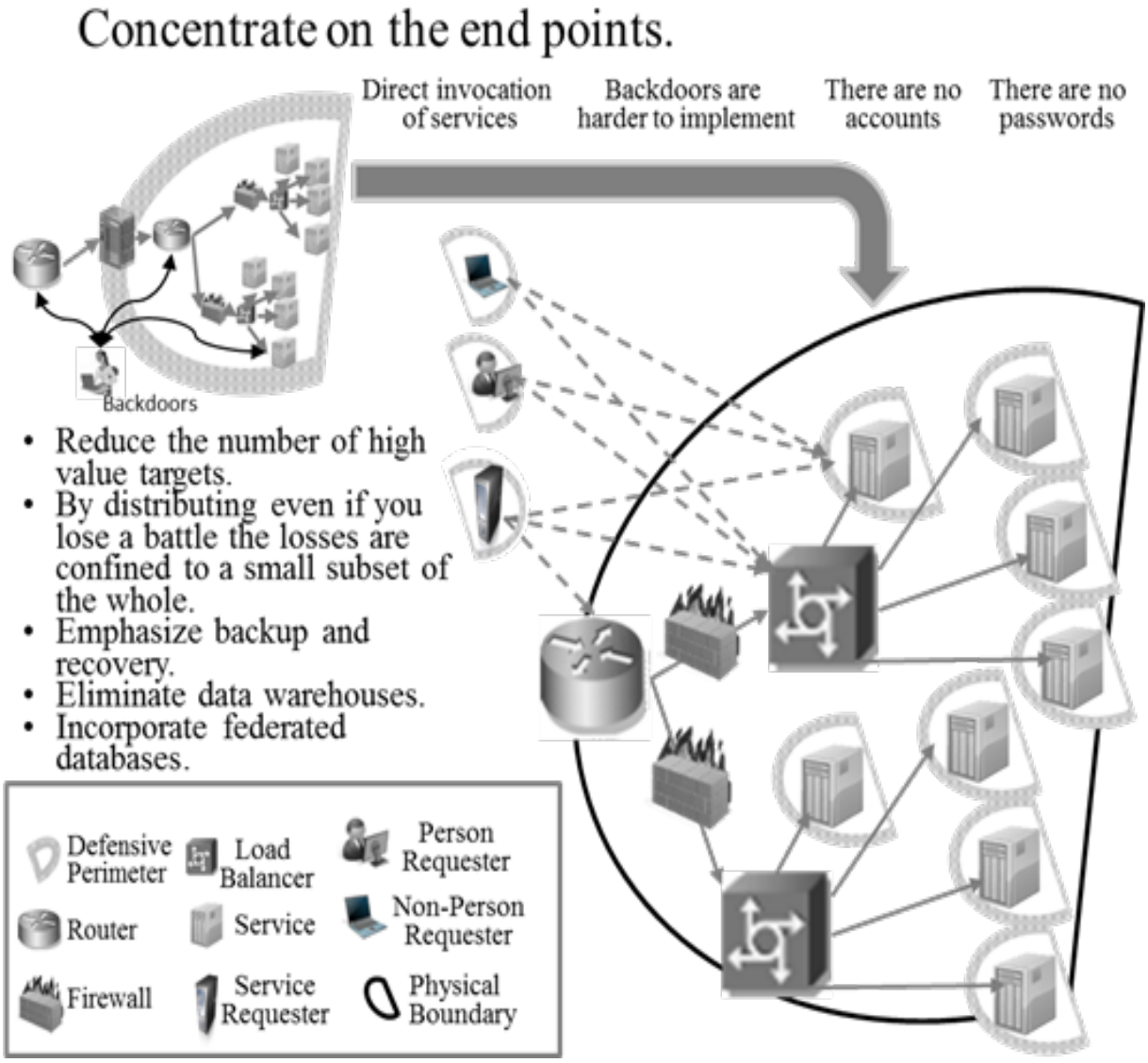

Figure 1 Distributed Security Architecture

The basic process of identification involves a two-way contract between two entities that are initiating a communication. Each entity needs to have some assurance that the party they are engaged with is a known entity and specifically the one to whom the communication should be allowed. This is done by the presentation of claims by each party that are verifiable and may be validated. These claims are often in the form of credentials.

Entities may be active or passive. Passive entities include storage elements, routers, wireless access points, some firewalls, and other entities that do not themselves initiate or respond to web service or web application requests. Active entities are those entities that request or provide services according to Enterprise Level Security (ELS). Active entities include users, applications, and services. All active entities have PKI certificates, and their private keys are stored in tamper proof, threat mitigating storage. Communication between active entities requires bi-lateral, PKI, end-to-end authentication. Active entities must be named in accordance with enterprise naming rules that assure uniqueness over space and time. Authentication is implemented by a verifiable identity claims-based process.

Figure 2 illustrates a combination of active and passive entities in a typical request flow. The requester and provider are both active entities using ELS. To complete the request, the provider calls on a passive data store using non-ELS methods. To mitigate the lower security of the provider to data store link, the data store interface is often locked down to only communicate with the provider.

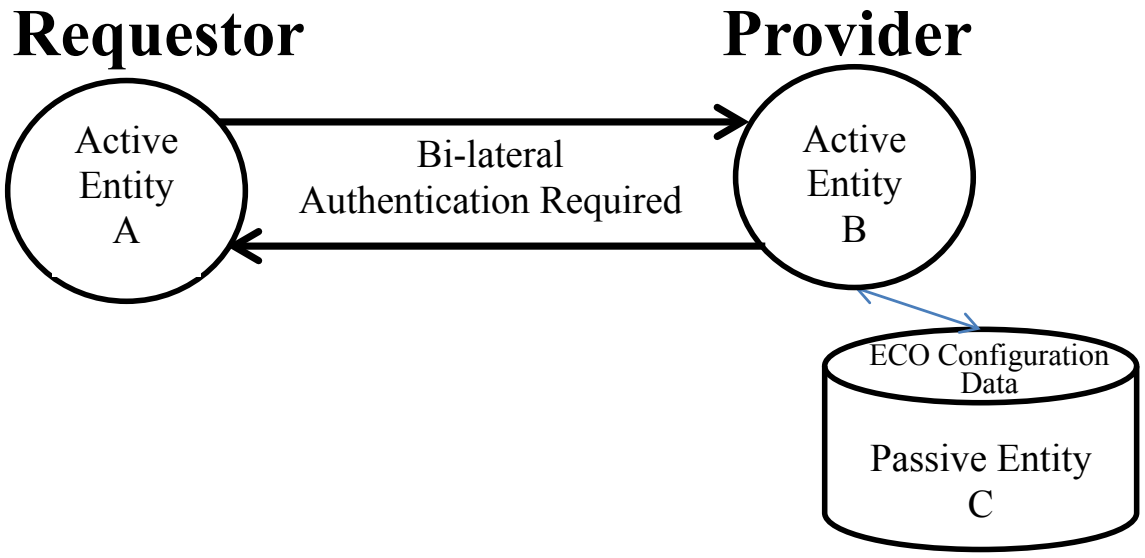

Figure 2 Communication between entities

ELS is designed for a high-assurance environment. For ELS, the primary concerns are the following five security principles:

- Know the Players,
- Maintain Confidentiality,
- Separate Access and Privilege from Identity,
- Maintain Integrity,
- Provide Accountability.

The following sections elaborate on how these are incorporated into the ELS framework.

Know the Players

All communication in ELS is conducted between known entities. Requesters know the entity that provides the data, and providers know who is requesting it. Entity identification and authentication is performed using identity credentials, as shown in Figure 3.

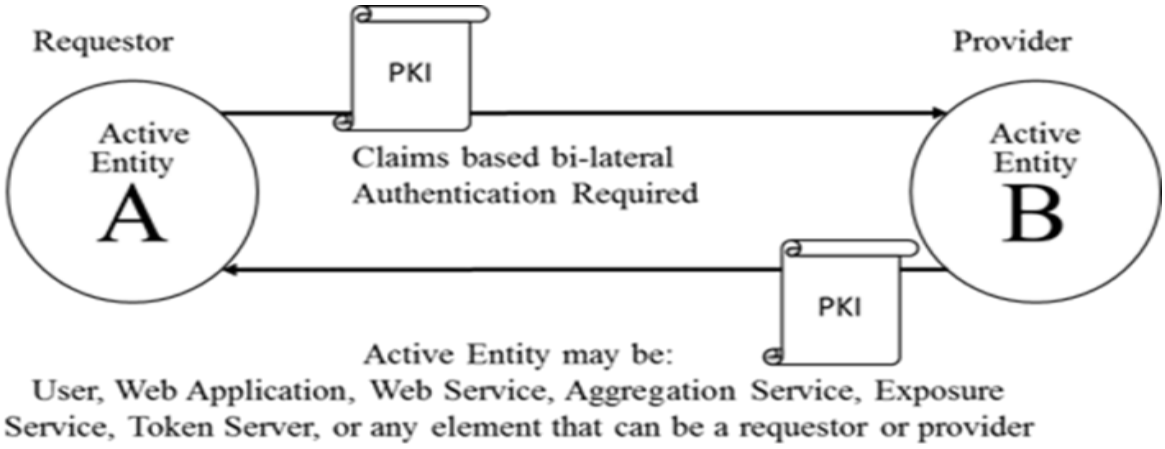

Figure 3 Bi-lateral Authentication

In ELS, the identity credential is an X.509 Public Key Infrastructure (PKI) certificate [1]. PKI certificates are verified and validated. Ownership is verified by a holder-of-key (HOK) check.

Maintain Confidentiality

After establishing the identity of the communication partner, the authenticated entities negotiate cryptographic algorithms and keys that enable confidential communication. As shown in Figure 4, this confidentiality is established through an end-to-end encrypted connection.

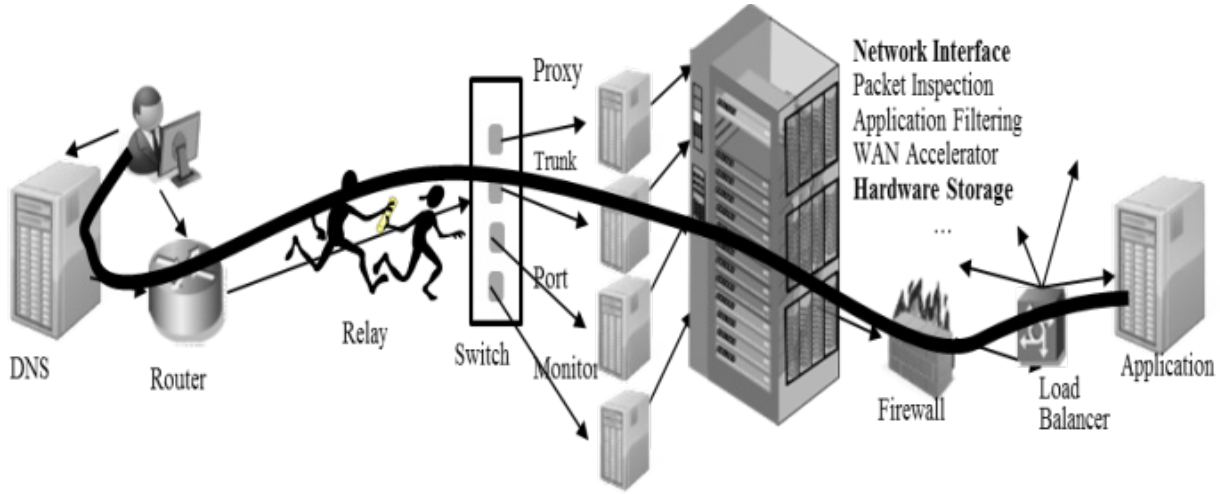

Figure 4 End-to-End Encryption

ELS establishes end-to-end confidentiality using Transport Layer Security (TLS) encryption, in which keys are held only by the endpoints [2].

Separate Access and Privilege from Identity

After the connection has been established from requester to provider, the provider knows the requester's identity. In ELS, this is not always enough to provide access, and an additional authorization credential that is tied to the identity must be provided for access. Figure 5 illustrates the transmission of this access credential.

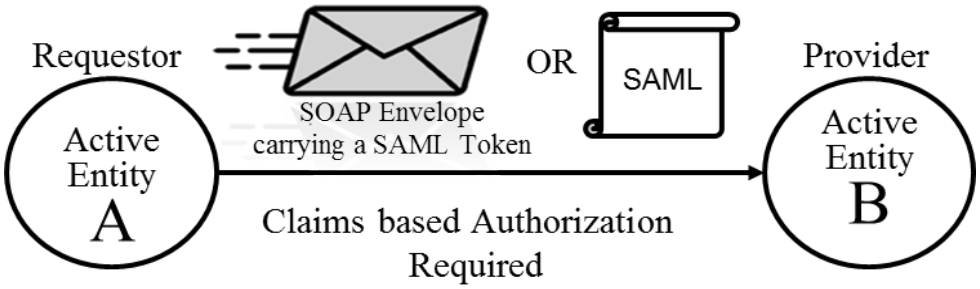

Figure 5 Claims-Based Authorization

In ELS the access credential is sent using the Security Assertion Markup Language (SAML) [3]. SAML tokens are issued and signed by a trusted Security Token Server (STS) that uses a backend Enterprise Attribute System (EAS) to manage attributes and compute access claims.

Maintain Integrity

It is important to know not just that others cannot view transmitted information but also that they cannot modify it. Integrity includes web data exchanged over an end-to-end connection as well as the SAML token and other messages that are exchanged. End-to-end data integrity is illustrated in Figure 6.

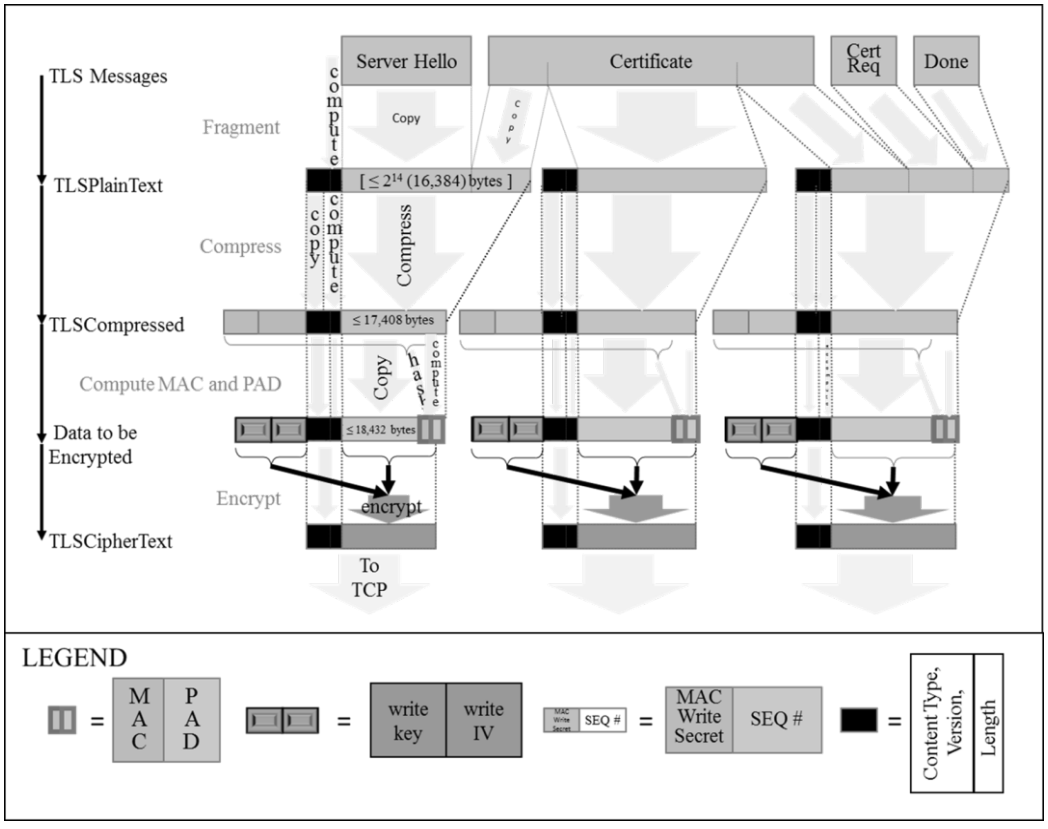

Figure 6 TLS Integrity Measures

In ELS, data integrity for web requests is implemented by TLS message authentication codes (MACs). Messages and SAML tokens include XML digital signatures, which are verified and validated [4].

Provide Accountability

In a high-assurance environment, accountability is important to allow rapid identification of problems and trace them to their source. It also allows better visibility into what is happening and how to allocate the enterprise resources based on who is using which services. Figure 7 illustrates the system of collecting and accumulating information about active entity activity.

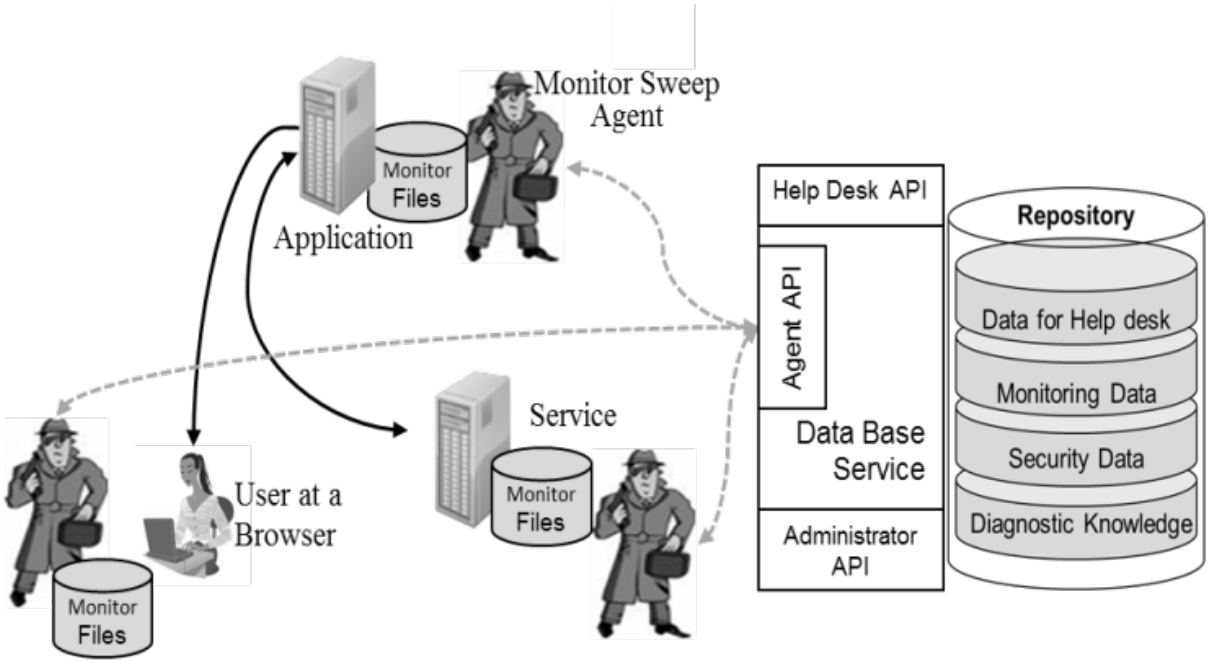

Figure 7 Accountability through Centralized Monitoring

All active entities must monitor specified activities. For enterprise files, a monitor sweep agent reads, translates, cleans, and submits to a data base for recording log records periodically or on demand. The details of this activity are provided in [5] and [6].

Federation Technical Considerations

The previous section describes some of the technologies and principles of ELS. This section discusses different ways to extend the principles of ELS to a federation partner based on the partner's technical capabilities. Options Include:

- ELS Federation,
- ELS-like Federation,
- Identity Credential Federation,
- Weak Identity Federation,
- Ad Hoc Federation,
- Person-to-person Federation.

The federation options are listed from most to least technologically compatible. Each option on the list also implicitly includes the options lower on the list. For example, an ELS-like partner provides all the options available to a Weak Identity Federation partner, as well as ELS-like Federation and Identity Credential Federation. This section discusses the technical means to accomplish federation for each of the options listed above.

ELS Federation

ELS federation is an agreement to accept identity and access claims from another enterprise. Federation is a long-term substantial agreement that is made at the enterprise level. To resolve federation issues, the federation STS relies on the following information:

- Certificates of Federated STSs for validating signatures in SAML tokens, and chain to trusted root CA;
- A set of identity-mapping pairs with the form (Identity1, Identity2), in which Identity1 in a SAML issued by the federated STS is to be mapped to Identity2 in the local enterprise;
- A set of mapping pairs of the form (Claim A, Claim B) where Claim A in a SAML issued by the federated STS is to be mapped to Claim B in the local enterprise;
- Additional attribute mappings associated with claim mappings.

An example of data captured in federation agreements is shown in Table 1. This shows the data for two separate federation agreements.

Table 1 Federation Data Requirements

Federation Partner 1 Information		
Certificate	Federation Partner 1 certificate and chain to root CA	
Identity Mappings	Identity 1	Identity 2
	Identity A	Identity B
	Identity r	<no change>
	…	…
Claim and Attribute Mappings	Claim A	<null>
	Claim n	Claim z
	Claim y and Attribute q	Claim y and Attribute r
	…	…
Federation Partner 2 Information		
Certificate	Federation Partner 2 certificate and chain to root CA	
Identity Mappings	Identity x	Identity y
	Identity Q	Identity R
	…	…
Claim and Attribute Mappings	Claim n	<no change>
	Claim p with Attributes x, y, z	Claim p with Attribute k
	Claim A	<no change>
	…	…

Each web service in the enterprise has a limited number of trusted root CAs for authentication credentials and trusted STS certificates for SAML signatures stored in its trust store. With ELS

federation a list of trusted partner CAs and STSs is established. Trusted partner CAs for identity credentials are distributed to applications and services. Trusted partner STS credentials are distributed to federation STSs within the enterprise. The federation STS is called by a service when an unknown authorization credential is encountered, and the federation STS checks against known federation partners to validate the credentials, creates a new SAML token with its own signature, and returns this to the application or service for processing.

For identity and claim mappings, the special cases of "null" and "no change" are acceptable in addition to explicit values. "Null" removes the claim or identity, while "no change" leaves the original claim or identity. The claims to be mapped must match claims from sources on both sides. Claims in the federation partner SAML must match the federation agreement exactly. Claims in the re-issued SAML must match claims for the target application or service. Identity and claim mappings are added to the federation store after an amendment to the federation agreement. Revocation of a federation agreement is accomplished by removing the federation partner from the trusted STS data store.

When a federation STS recognizes and validates a partner authorization credential, it uses its mapping list to map the received credential into a new credential with possibly different identity, claims, and attributes. This new credential is signed by the federation STS and returned to the requesting application or service. The application or service then processes this new SAML token as though it had received it from a valid requester within the enterprise. Failure to validate an incoming SAML token by the federation STS results in an error message response to the application or service, which leads to an authorization failure at the application or service.

ELS-like Federation

If the federation partner is not using ELS but does have a way to provide the security functions that ELS provides, then a more complicated federation agreement may be needed with some translation algorithms. This is similar to ELS Federation, but it accounts for different semantics, formats, and data encodings. The mappings may not be exact, in which case there may be some loss of information in the translation. This may hinder automation and scalability because mismatches in data resolution could lead to either denial when access should be allowed or access when it should be denied. This is generally unavoidable when data is represented in different ways.

These differences also apply to tokens and credentials. If the strength of credentials in the partner enterprise is variable, this can weaken the entire authentication process. For example, if the partner provides software authentication credentials for some users instead of hardware credentials for all users, then unless the source of the credential is explicitly provided in the signed portion of the certificate, it is not possible to provide access to hardware-credentialed entities and deny access to software-credentialed entities. The secure choice for an enterprise that requires hardware certificates in such a situation is to deny access even though the entities using hardware credentials should be given access.

For authorization credentials, if the partner uses something other than the approved SAML 2.0 format that is used in ELS, potential problems exist again because certain guarantees provided by the SAML format, such as validity windows, signatures, or encryption, may be weaker or nonexistent in the partner authorization token. Also, even if the token is equally strong, the local

STS needs to understand the partner token format in order to parse, validate, and translate it into a locally comprehensible SAML for the local applications and services. Token translators exist to transform one token format into another, but they address only the format, not the meaning or security properties.

When the partner is "ELS-like," most of the work is already done and the process of federating is similar to that with an ELS partner. The challenge is working out the small differences and making sure that these small differences do not open up large security holes or prevent large groups of entities from receiving proper access. Shortcuts and workarounds may be necessary to provide a seamless experience to the end users; however, the majority of cases should work with standard translation schemes as long as the protocols, standards, and data formats are commonly used and available.

In the event that formats and data are incompatible, additional work to make needed conversions at the requesting enterprise may solve the problem. For example, many STSs can be configured to issue tokens according to different protocols and formats, so configuring a partner STS to use the proper SAML format could resolve token format differences. Authentication credentials could be addressed by indicating whether a credential is stored in hardware or software so that receiving entities can decide whether to allow access. This could be accomplished by dividing the issuing CAs into those that issue only hardware credentials and those that issue only software credentials. These solutions are intended as quick fixes, and they do not require any new technology. In some cases these can move an ELS-like partner to an ELS partner, but in most cases they remain ELS-like but with more streamlined operations.

Identity Credential Federation

A federation partner may provide identity credentials but not authorization credentials, such as an account-based system in which the ID is used to log in. In this situation a basic function of ELS is missing, so it is no longer ELS-like, but the identity credentials can still be used as a starting point for federation.

One solution is to include a mapping from partner identity to local identity that also includes associated authorization claims. This would be performed at the STS as part of the federation mappings. This is not the typical use for these mappings, and it requires modifications to Table 1 to combine the identity, claim, and attribute mappings. This method would work if the identity credential is passed for authentication and the authorization information is exchanged between enterprises and incorporated into the federation agreement. This should generally only be used for small sets of requesters, since these federation mappings are intended to be for claim and attribute equivalences and generic identity transform rules, not explicit per-entity claim and attribute information.

For larger-scale federation, the data owner or some other entity with access to the data can delegate the appropriate claims to the appropriate individuals in the partner enterprise. The delegation framework within ELS is designed to allow such short-term access to specific individuals. In this case the data owner, and not the STS, maintains the mappings through the delegation service. This is a more appropriate place to store this information. However, it still requires manually assigning claims to individuals based on their identities.

The basic structure for federated delegation is shown in Figure 8. The dashed lines represent the flow for setting up the delegation. The local delegator uses the Special Delegation Service to assign access claims to the federated partner identity as stored in the authoritative content store for federation partners. The special delegation service is separate from the standard in-enterprise delegation service, and the content store for federation partners is separate from the normal store for in-enterprise entities. This special delegation service uses the claims exposure and editor service to store access claims for the federated identity in the Claims Repository.

The solid lines represent the flow for a federated requester to retrieve a SAML token with delegated claims. The partner interfaces with the STS, just as an in-enterprise user does, and the STS then makes a request, just as for any other requester, to retrieve the claims from the Claims Repository.

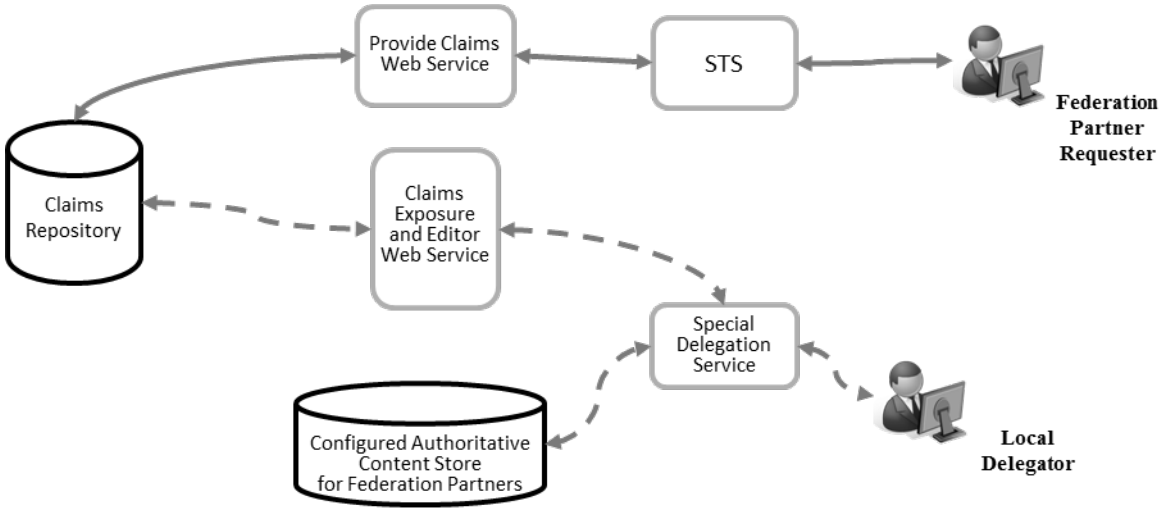

Figure 8 Identity credential federation using delegation

A third solution is to use a separate authoritative data store for the partner entities and tie it into the Enterprise Attribute System that feeds the SAML creation process. This brings the partner entities into the local enterprise and allows those entities to request local SAML tokens that will work just like those from requesters within the enterprise. Maintenance of the authoritative data store could be performed locally or assigned to the partner enterprise. Local maintenance would again require detailed manual attention, while letting the partner maintain it allows them to manage their own entities. This is the best-integrated solution, since the attributes required are brought into the ELS system.

In all cases, a store maintains information about federated identities. For the mapping method, this is the federation agreement at the federation STS, which contains both identity information and claims and attributes used for access. For the delegation method, it is the store used by the special delegation service, which contains only identities, but no claims because these are delegated as needed. With the separate store, this store contains all identities and their attributes.

Weak Identity Federation

If the federation partner does not provide sufficiently strong identity credentials, then identity credential federation is not possible. For example, a federation partner that uses username and

password authentication is not compatible with the PKI-based authentication used for ELS. However, if it is determined that the need to federate outweighs the security risks associated with weak credentials, then there are methods to allow such access while preserving many of the security goals of ELS. The primary challenge is that ELS requires PKI authentication to the STS to receive a SAML token and to the application or service to establish a connection, but the federation partner does not have this capability.

In order to allow creation of the SAML, a separate STS can be set up to receive other forms of authentication, such as username and password, Kerberos over TLS, other non-PKI TLS authentication, tokens, single sign on (SSO), or other methods. The STS can use the authenticated identity to generate a SAML token. If the identity format does not match the ELS standard distinguished name (DN) format, a mapping can be performed, either algorithmically or based on an enterprise-wide database of identity and derived DN pairs. If the federation partner has an STS that already uses non-PKI authentication, it can be leveraged to create authorization credentials.

Application and service endpoints can rely on alternative authentication methods, map the identity to a derived DN if necessary, and then compare this identity to the identity in the SAML token. In the generated SAML token, the authentication method can be indicated, or federation partners with non-PKI authentication at their STSs can be noted in the federation agreement so that receiving applications and services can enforce access accordingly. For example, standard PKI authentication may provide full access while alternative authentication methods provide limited access that is sufficient for federation purposes. Because there is a clear distinction in authentication method and separate endpoints set up for access, this method provides a way to manage federated access independent of normal access. If a security problem is detected at a federated non-PKI endpoint, that endpoint can be shut down without affecting normal ELS operations.

This process is cumbersome and redundant because it requires new instances of applications and services and potentially standing up a new STS. It may require additional logic to enforce different access based on the authentication method. It also weakens enterprise security by allowing non-ELS access to applications and services that otherwise require ELS. The access can be limited, but it may provide an attacker an easier entrance to ELS-protected data, which could be leveraged through additional attacks to gain full access. This should only be attempted for urgent and infrequent federation agreements with the understanding of the risk to all affected applications and services, as well as the risk to the enterprise in general. This is not an ELS-compliant solution, because it fails to satisfy authentication and accountability requirements. However, it is a way that an enterprise can share all or part of ELS-protected resources while leveraging existing ELS infrastructure. As a result, this may be a better solution than other options available.

Ad Hoc Federation

If there is no easy way in advance to identify what is to be shared with whom, then an ad hoc approach can address federation requirements. Delegation may be used to explicitly share resources with individuals just as needed. Delegation was mentioned above as an option for the case in which strong identity credentials are available. It is best suited for when the delegation decisions are distributed. A large federation agreement can be implemented by dividing the

resources to be shared and the partner entities to be given access into smaller, manageable groups. A large federation agreement is broken into a large number of locally administered smaller agreements. Central control is maintained by the hierarchical structure of the delegations. This may or may not be possible for the structure of the sharing agreement between enterprises.

This approach requires some form of authentication for federated individuals. This is not necessarily a separate solution from those listed above, but instead an option for generating access claims for these methods. This is in contrast to explicitly assigning attributes or claims to these identities or mapping federated identities and claims through a federation STS.

This option is listed after weak identity federation because of the ad hoc and decentralized nature of the sharing. With the options above, a central source determines who can access which resources. In this case individuals make these decisions rather than the enterprises forming federation agreements. As a result, from the enterprise perspective, this is less secure, especially as the scope and scale of sharing increase.

Person-to-Person Sharing

If all the solutions above have technological or other issues that make them infeasible, an individual may make the decision to share information with a federation partner out of band on a person-to-person basis. This type of sharing should be properly authorized, and the person sharing the material should have release authority with guidelines. This could involve emailing documents, placing them on a public web-site, providing printed copies, or verbally providing the needed information. This goes around ELS by allowing access without any automated enterprise-level checks of identity or access claims. The last two also lack attribution, because the one sharing and the one receiving are the only entities that know about the transaction. In this case shared information may be fingerprinted or steganographically imprinted for later forensics in case of leaks. It is sufficiently hard to stop such sharing that small amounts of unauthorized sharing are expected throughout the enterprise and with partners. ELS and the federation options listed above attempt to provide a feasible option that is better than going completely out of band.

Although this method essentially bypasses most or all ELS security, it is mentioned because it is better than many alternatives that involve giving full electronic access by credential sharing, workarounds or back doors. This method lacks the formal security of ELS (it may include accountability provisions on the enterprise side), but it provides the important property that only the information that is needed is shared because a human with authority to access and release data is making the decision on sharing each item. To help secure this method of sharing information, policies can guide people on which real-world security checks to make related to such sharing. Person-to person sharing has a long history prior to electronic sharing, so this accumulated knowledge can be used to shape policy.

Evaluating Options

The options presented above describe a spectrum from full ELS sharing to non-ELS sharing. It is tempting to say that ELS is the most secure and person-to-person is the least secure, but in reality scale matters. The security features are reduced from ELS to person-to-person, but the scale also tends to be reduced. Attacks are easier at the non-ELS end of the spectrum, but their smaller scale is likely to result in smaller incidents. At the full ELS end of the spectrum, the security is high and attacks are difficult, but a successful attack could rapidly escalate to a major incident.

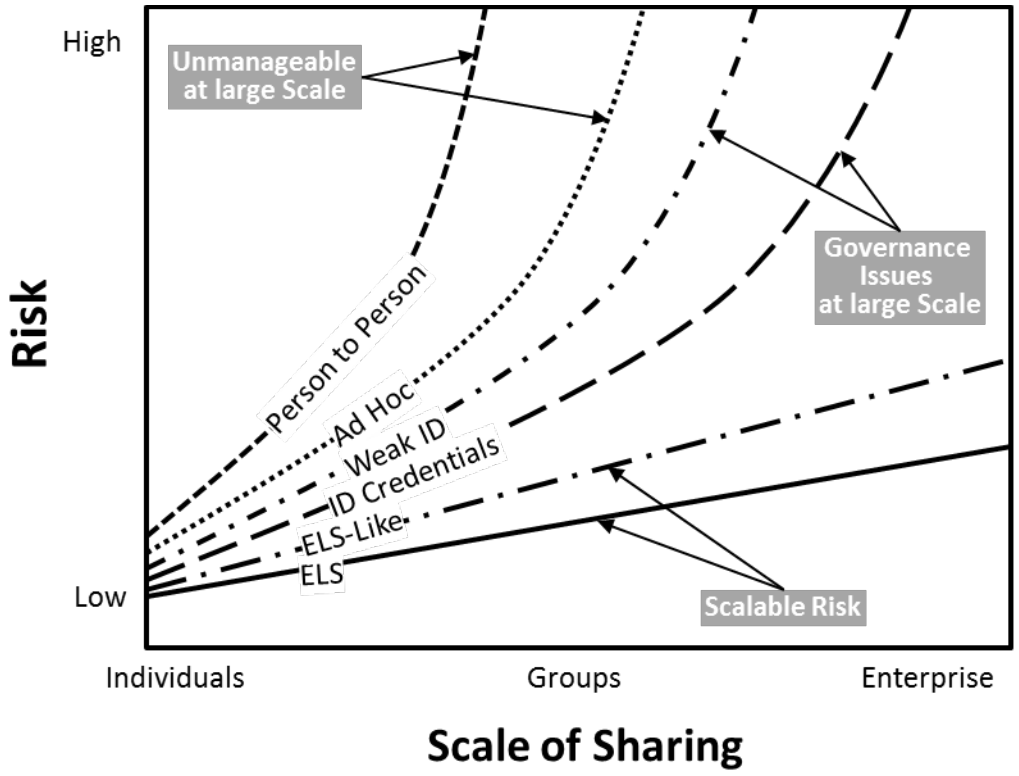

Figure 9 Notional comparison of federation security at different scales of operation

For efficient sharing, ELS provides the most secure solution at scale and should be the preferred choice by the enterprise. Other options provide acceptable security at small scales, but they should not be put to widespread use, because the resulting vulnerabilities quickly escalate with scale-up. Figure 9 shows a notional plot of the vulnerability associated with different federation option as a function of scale. The exact curves vary according to how risk and size are measured, but the key idea is that each federation option will have some maximum scale at which its security is acceptable. At the scale of a few individuals sharing information, the risk remains low for all methods, but as the scale increases, the risk associated with the approaches diverges, with ELS offering the only acceptable solution at full enterprise scale.

Despite the scalability limitations, all approaches can be viable options. For example, "Weak ID" federation, although not desirable compared to ELS, is better than the ad hoc or person-to-person approaches, and making all options available provides a smoother continuum of choices based on resources available. A particular enterprise may wish to limit the options to a subset of these or expand to include more. It may be better to adopt a strict "ELS or nothing" policy to motivate partners to adopt ELS, even if it means very limited sharing. For cases in which the focus is on sharing, providing all options helps to provide the best security available for the given situation. To manage overall enterprise risk, it may be necessary to limit the absolute number of federation partners or their constituent members that use each non-ELS type of federation to maintain the overall vulnerability level below a desired threshold.

Federation Trust Considerations

The technical considerations are important for federation, but having the technology available does not always mean it is a good option. This section takes a different perspective to federation and considers the tradeoffs among federation options based on the level of trust that exists with the federation partner. This is usually important to consider when making information or services available to partners, and less important when accessing partner information and services. It is assumed that some sharing is desired, regardless of the level of trust. It is also assumed that sharing should be limited, since this is true even within a single enterprise using ELS. The challenge is how to use the available technology to build upon the real-world trust relationships to enable as many valid sharing behaviors as possible while preventing behaviors that are not valid or desired.

In this section, trust is meant to be the degree to which one enterprise believes that the other enterprise will not abuse technical capabilities to violate the agreed-upon terms of the federation agreement. It does not apply to weak security methods such as username/password authentication or weak cryptographic algorithms such as RC4 or MD5.

Trust can be a tricky issue. For example, if one enterprise trusts the root certificate authority (CA) of the other enterprise, then PKI authentication can work across the enterprises. Thus, it might be tempting for one enterprise to declare, “Enterprise 1 trusts Enterprise 2’s root CA,” to make security work. This declaration may sound like, “Enterprise trusts Enterprise 2’s root CA and therefore Enterprise 1 will encode that trust in its system to allow both systems to work together.” However, stating trust for the purpose of interoperability is really saying “Enterprise 1 will open up a security vulnerability in its system that Enterprise 2 can exploit and trust Enterprise 2 not to abuse it.” Because such trust opens up vulnerabilities, it is important that statements of trust are truly based on existing or established relationships and not simply put in place for convenience. Since trust is usually not an all-or-nothing proposition, any trust relationship must also be accompanied by monitoring and measures to identify and block malicious behavior by a partner, which can be summarized as “trust but verify.”

For a trusted partner or a partner for whom there is accountability and a long-standing relationship, trusting a partner CA might be a good way to federate. For a situation in which an enterprise wants to share but really does not trust the partner not to abuse additional privilege, there is no longstanding relationship or accountability, or the damage that can be done is greater than any retribution likely to be enacted, this is a risky choice. The key is to first decide what to actually trust and then use that as a basis to build the system – not to build a system first and then declare trust where needed in order to make it work.

This section provides options for what to do with different trust levels. Of course, trust can be increased through external policies and agreements as well, so these options do not exist in isolation from the real world. However, it is often easier to build a system based on existing trust than to work to establish trust, since changing the real world to match a technical system is often slower than changing a technical system to match the real world.

Full Trust

If one enterprise fully trusts another enterprise, federation can be done in any number of ways. This could be the case for two divisions within a larger enterprise, a merger between two companies, two parallel systems within a single enterprise, or a hierarchical arrangement, in which one group is sharing with another that contains it. In this case, any option will work, and the best choice is usually the one that enables the most integration and sharing or the one that is simplest to set up. In this case, the degree of ELS compatibility is likely to be the determining factor for choosing a solution.

Infrastructure Trust

If one enterprise trusts the high-level security functions, such as certificate authorities and other enterprise level functions, then the best option is one that leverages this existing trust and limits access based on these trusted services. For example, if both enterprises use ELS, a federation agreement among STSs is a viable choice, since this leverages trusted root CAs and trusted STSs and attribute stores, which are all high-level security functions.

In some cases only a part of a partner's ELS infrastructure is trusted or the partner's infrastructure supports most but not all ELS requirements. For example, if the root CA is trusted but the STSs or attribute stores are not trusted, identity-based federation with delegation or a local attribute store for those identities might be a better option. This leverages the existing trust in the identity credentials while providing the claims through the local STS and internally maintained attributes. This provides the function of the untrusted partner STS or attributes in a trusted way. The lack of trust is similar to a lack of technology, since a function that exists but is not trusted is similar to one that is missing.

The complementary situation is when a partner STS is trusted while partner identity credentials are not trusted. This case is not common, because weak authentication to an application often correlates with weak authentication to the STS, which means the SAML token cannot be trusted even if the STS and attribute stores are trusted. The best technical solution when identity services are not trusted may be no technical solution, i.e., person-to-person federation, which would limit the scale of sharing to individuals or small groups of known entities.

Individual Trust

In some cases certain individuals of a partner organization are trusted even though the organization's security functions are not. In this case, identity-based federation may be possible for the specific identities who are trusted. For example, a partner with a smart card credential might be technologically acceptable since the public/private key pair can be recorded and used for authentication without the associated certificate. However, this requires no key escrow or other method be in use by the partner enterprise to duplicate the private key associated with the credential. This method relies explicitly on the strength of the keys associated with the credential and their storage. This fails to achieve the efficiency and scalability of PKI, but allows smaller-scale point-to-point trust to be established using specific public/private key pairs. The certificate is replaced by an internal mapping of public keys to identities.

Another option is to provide a local credential to the trusted individual. This brings the individual into the enterprise. This is a well-integrated method, but proper enterprise-level vetting often

cannot be completed on such an individual due to the organization he works for. An individual in an enterprise may trust an individual from a partner enterprise, but the enterprise may not trust them sufficiently to issue a credential.

A local attribute store can record attributes for the trusted individual in order to provide proper access credentials through a local STS. This requires the STSs applications and services to check not just issuers of certificates but public keys for specific individual requesters. Revocation checking must also be supported within the enterprise for such individual keys.

A simpler option for leveraging this trust in the individual and associated credential is delegation. This allows resource owners to assign which applications and services a partner can access instead of providing access based on the partner's attributes. Automatic access based on attributes provides more scalability and automation than delegation, because with delegation all access must be granted manually. This might be an acceptable tradeoff when limited data is to be shared with a small number of individuals in a secure way. In this case, the STS operates as usual and only the applications and services are modified to trust individual keys.

No Trust

In some cases, data owners in an enterprise really do not trust a federation individual or his organization not to abuse their access but they still need to provide some access. Since they cannot trust individuals, the option of individual trust does not apply. The problem is that any access granted is likely to be abused, such as copying and exfiltrating data, or rendering inoperable or untrustworthy any services or devices to which more generous access is granted. In this case, the idea is to limit the individual to only what is needed when it is needed and to heavily monitor activity to detect suspicious behavior where limits cannot be strictly enforced through technology. Shared information may be fingerprinted or steganographically imprinted for later forensics in case of leaks.

One option is to provide a partner individual with a locked-down device that is preconfigured with a credential that has been delegated appropriate access and to let him use this as his only method of access. In this case the issuing enterprise tightly controls the hardware, software, credentials, and access. This method limits the information to the display on the locked-down device, which can be remotely monitored. If the device is not returned within the timeframe of the valid access, then it can be remotely disabled through device management policy. Monitoring of the device and the data accessed can also reveal potential exfiltration and initiate disabling of the machine and termination of delegation privileges.

None of the automated technological solutions are viable, because they are designed to facilitate automatic sharing of information. Without trust, no automation is desired. For all trust levels, the idea of "trust but verify" underlies all communication. What changes as trust levels decrease is not just what is shared but the measures taken to prevent, detect, and limit harm from the federation partner.

Conclusion

This paper examines secure enterprise federation options from the perspectives of both technical capabilities and trust. The paper reviews six federation methods:

1) ELS Federation,
2) ELS-like Federation,
3) Identity credential Federation,
4) Weak identity federation,
5) Ad Hoc Federation,
6) Person-to-Person Sharing.

All solutions are acceptable at small scale, but only those near the top of the list scale to the organizational or enterprise level while maintaining acceptable security. The best choice is generally the one closest to the top of the above list, but these top-end options require the most infrastructure for proper operation. When less-secure options are chosen, it is important as part of an overall risk management process for the enterprise to monitor and limit the extent of their aggregate use across the enterprise. This limits the scale of the lower options in order to maintain the assurance requirements of the enterprise.

Technological solutions also must be considered in the context of existing trust. A common but flawed solution is to choose the best technical solution and simply declare any required trust by formal edict. This masks the underlying trust issues and provides these untrusted partners front-door access to enterprise applications and services. A better approach is to first assess the level of trust in a partner enterprise, its security services, and the individuals involved in the sharing, and then build a federation agreement based on those solid building blocks – and those alone.

This research is part of a body of work for high-assurance enterprise computing using web services. Elements of this work include bi-lateral end-to-end authentication using PKI credentials for all person and non-person entities, a separate SAML credential for claims-based authorization, full encryption at the transport layer, and a defined federation process. Many of the elements of this work are described in [8]-[19].

References

[1]. X.509 Standards

a. DoDI 8520.2, Public Key Infrastructure (PKI) and Public Key (PK) Enabling, 24 May 2011
b. JTF-GNO CTO 06-02, Tasks for Phase I of PKI Implementation, 17 January 2006
c. X.509 Certificate Policy for the United States Department of Defense, Version 9.0, 9 February 2005
d. FPKI-Prof Federal PKI X.509 Certificate and CRL Extensions Profile, Version 6, 12 October 2005
e. RFC Internet X.509 Public Key Infrastructure: Certification Path Building, 2005
f. Public Key Cryptography Standard, PKCS #1 v2.2: RSA Cryptography Standard, RSA Laboratories, October 27, 2012
g. PKCS#12 format PKCS #12 v1.0: Personal Information Exchange Syntax Standard, RSA Laboratories, June 1999 http://www.rsa.com/rsalabs/node.asp?id=2138 PKCS 12 Technical Corrigendum 1, RSA laboratories, February 2000

[2]. TLS family Internet Engineering Task Force (IETF) Standards

In draft for reference only:

a. TLS Renegotiation Support Extension to HTTP/2, 2015-03-24
b. Terminology related to TLS and DTLS, 2015-03-26
c. X.509v3 TLS Feature Extension, 2015-04-06
d. TLS over HTTP, 2015-03-09
e. A TLS ClientHello padding extension, 2015-02-17

f. Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS) Versions 1.2 and Earlier, 2015-03-09
g. Transport Layer Security (TLS) Session Hash and Extended Master Secret Extension, 2015-04-16
h. The Transport Layer Security (TLS) Protocol Version 1.3, 2015-03-09

Standards:

i. RFC 2830 Lightweight Directory Access Protocol (v3): Extension for Transport Layer Security, 2000-05
j. RFC 3749 Transport Layer Security Protocol Compression Methods, 2004-05
k. RFC 4279 Pre-Shared Key Ciphersuites for Transport Layer Security (TLS), 2005-12
l. RFC 5246 The Transport Layer Security (TLS) Protocol Version 1.2, 2008-08
m. RFC 5289 TLS Elliptic Curve Cipher Suites with SHA-256/384 and AES Galois Counter Mode (GCM), 2008-08
n. RFC 5929 Channel Bindings for TLS, 2010-07
o. RFC 6358 Additional Master Secret Inputs for TLS, 2012-01
p. RFC 7251 AES-CCM Elliptic Curve Cryptography (ECC) Cipher Suites for TLS, 2014-06
q. RFC 7301 Transport Layer Security (TLS) Application-Layer Protocol Negotiation Extension, 2014-07
r. RFC 7457 Summarizing Known Attacks on Transport Layer Security (TLS) and Datagram TLS (DTLS), 2015-02
s. RFC 2712 Addition of Kerberos Cipher Suites to Transport Layer Security (TLS), 1999-10

[3]. Organization for the Advancement of Structured Information Standards (OASIS) open set of Standards
a. N. Ragouzis et al., Security Assertion Markup Language (SAML) V2.0 Technical Overview, OASIS Committee Draft, March 2008
b. P. Mishra et al. Conformance Requirements for the OASIS Security Assertion Markup Language (SAML) V2.0. OASIS Standard, March 2005
c. S. Cantor et al. Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0, OASIS Standard, March 2005

[4]. William List and Rob Melville, IFIP Working Group 11.5, Integrity In Information, Computers and Security, Volume 13, Issue 4, pp. 295–301, Elsevier, doi:10.1016/0167-4048(94)90018-3, 1994

[5]. William R. Simpson and Coimbatore Chandersekaran, CCCT2010, Volume II, pp. 84–89, "An Agent Based Monitoring System for Web Services," Orlando, FL, April 2011

[6]. William R. Simpson and Coimbatore Chandersekaran, 1st International Conference on Design, User Experience, and Usability, part of the 14th International Conference on Human-Computer Interaction (HCII 2011), "A Multi-Tiered Approach to Enterprise Support Services," 10 pp. Orlando, FL, July 2011. Also published in: A. Marcus (Ed.): Design, User Experience, and Usability, Pt I, HCII 2011, LNCS 6769, pp. 388–397, 2011.© Springer-Verlag Berlin Heidelberg 2011

[7]. Frank Konieczny, Eric Trias and Nevin Taylor, "SEADE: Countering the Futility of Network Security," Air and Space Power Journal, Sep-Oct 2015, Vol 29, No.5, pg. 4

[8]. Coimbatore Chandersekaran and William R. Simpson, World Wide Web Consortium (W3C) Workshop on Security Models for Device APIs, "The Case for Bi-lateral End-to-End Strong Authentication," 4 pp., London, England, December 2008

[9]. William R. Simpson and Coimbatore Chandersekaran, The 2nd International Multi-Conf.on Engineering and Technological Innovation: IMETI2009, Volume I, pp. 300–305, "Information Sharing and Federation," Orlando, Florida, July 2009

[10]. Coimbatore Chandersekaran and William R. Simpson, The 3rd International Multi-Conf. on Engineering and Technological Innovation: IMETI2010, Volume 2, "A SAML Framework for Delegation, Attribution and Least Privilege," pp. 303–308, Orlando, Florida, July 2010

[11]. William R. Simpson and Coimbatore Chandersekaran, The 16th International Command and Control Research and Technology Symposium: CCT2011, Volume II, pp. 84–89, "An Agent Based Monitoring System for Web Services," Orlando, Florida, April 2011

[12]. William R. Simpson and Coimbatore Chandersekaran, International Journal of Computer Technology and Application (IJCTA), "An Agent-Based Web-Services Monitoring System," Vol. 2, No. 9, September 2011, pp. 675–685

[13]. William R. Simpson, Coimbatore Chandersekaran and Ryan Wagner, Lecture Notes in Engineering and Computer Science, Proceedings World Congress on Engineering and Computer Science 2011, Volume I, "High Assurance Challenges for Cloud Computing," pp. 61–66, San Francisco, October 2011

[14]. Coimbatore Chandersekaran and William R. Simpson, Lecture Notes in Engineering and Computer Science, Proceedings World Congress on Engineering 2012, The 2012 International Conference of Information Security and Internet Engineering, Volume I, "Claims-Based Enterprise-Wide Access Control," pp. 524–529, London, July 2012

[15]. William R. Simpson and Coimbatore Chandersekaran, Lecture Notes in Engineering and Computer Science, Proceedings World Congress on Engineering 2012, The 2012 International Conference of Information Security and Internet Engineering, Volume I, "Assured Content Delivery in the Enterprise," pp. 555–560, London, July 2012

[16]. William R. Simpson and Coimbatore Chandersekaran, Lecture Notes in Engineering and Computer Science, Proceedings World Congress on Engineering and Computer Science 2012, Volume 1, "Enterprise High Assurance Scale-up," pp. 54–59, San Francisco, October 2012

[17]. Coimbatore Chandersekaran and William R. Simpson, International Journal of Scientific Computing, Vol. 6, No. 2, "A Uniform Claims-Based Access Control for the Enterprise," December 2012, ISSN: 0973-578X, pp. 1–23

[18]. William R. Simpson, Kevin Foltz and Coimbatore Chandersekaran, Lecture Notes in Engineering and Computer Science, Proceedings World Congress on Engineering and Computer Science 2014, Volume 1, "Distributed versus Centralized Protection Schema for the Enterprise," pp. 173-184, Berkeley, CA. October 2014

[19]. William R. Simpson and Kevin Foltz, Proceedings of the World Congress on Engineering 2015 Vol I, WCE 2015, July 1-3, 2015, London, U.K., "Wide Area Network Acceleration in a High Assurance Enterprise," pp. 502–507