Security and Privacy Considerations for the OASIS Security Assertion Markup Language (SAML) V2.0

Committee Draft 01, 18 August 2004

Document identifier:
sstc-saml-sec-consider-2.0-cd-01

Location:
http://www.oasis-open.org/committees/documents.php?wg_abbrev=security

Editors:
Frederick Hirsch, Nokia
Rob Philpott, RSA Security
Eve Maler, Sun Microsystems

SAML V2.0 Contributors:
Conor P. Cahill, AOL
Hal Lockhart, BEA Systems
Michael Beach, Boeing
Rick Randall, Booze, Allen, Hamilton
Tim Alsop, Cybersafe
Nick Ragouzis, Enosis
John Hughes, Entegrity Solutions
Paul Madsen, Entrust
Irving Reid, Hewlett-Packard
Paula Austel, IBM
Maryann Hondo, IBM
Michael McIntosh, IBM
Tony Nadalin, IBM
Scott Cantor, Internet2
RL 'Bob' Morgan, Internet2
Rebekah Metz, NASA
Prateek Mishra, Netegrity
Peter C Davis, Neustar
Frederick Hirsch, Nokia
John Kemp, Nokia
Charles Knouse, Oblix
Steve Anderson, OpenNetwork
John Linn, RSA Security
Rob Philpott, RSA Security
Jahan Moreh, Sigaba
Anne Anderson, Sun Microsystems
Jeff Hodges, Sun Microsystems
Eve Maler, Sun Microsystems
Ron Monzillo, Sun Microsystems
Greg Whitehead, Trustgenix

Abstract:

This non-normative specification describes and analyzes the security and privacy properties of SAML.

Status:

This is a **Committee Draft** approved by the Security Services Technical Committee on 17 August 2004.

Committee members should submit comments and potential errata to the security-services@lists.oasis-open.org list. Others should submit them by filling out the web form located at http://www.oasis-open.org/committees/comments/form.php?wg_abbrev=security. The committee will publish on its web page (http://www.oasis-open.org/committees/security) a catalog of any changes made to this document.

For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights web page for the Security Services TC (http://www.oasis-open.org/committees/security/ipr.php).

Table of Contents

1 Introduction

This non-normative document describes and analyzes the security and privacy properties of the OASIS Security Assertion Markup Language (SAML) defined in the core SAML specification [SAMLCore] and the SAML bindings [SAMLBind] and profiles [SAMLProf] specifications. The intent in this document is to provide information to architects, implementors, and reviewers of SAML-based systems about the following:

- The privacy issues to be considered and how SAML architecture addresses these issues
- The threats, and thus security risks, to which a SAML-based system is subject
- The security risks the SAML architecture addresses, and how it does so
- The security risks it does not address
- Recommendations for countermeasures that mitigate those security risks

Terms used in this document are as defined in the SAML glossary [SAMLGloss] unless otherwise noted.

The rest of this section describes the background and assumptions underlying the analysis in this document. Section 4 provides a high-level view of security techniques and technologies that should be used with SAML. The following sections analyze the risks associated with the SAML assertions and protocol as well as specific risks associated with SAML bindings and profiles.

2 Privacy

SAML includes the ability to make statements about the attributes and authorizations of authenticated entities. There are very many common situations in which the information carried in these statements is something that one or more of the parties to a communication would desire to keep accessible to as restricted as possible a set of entities. Statements of medical or financial attributes are simple examples of such cases.

Many countries and jurisdictions have laws and regulations regarding privacy and these should be considered when deploying a SAML based system. A more extensive discussion of the legal issues related to privacy and best practices related to privacy may be found in the Liberty Privacy and Security Best Practices document [LibBestPractices].

Parties making statements, issuing assertions, conveying assertions, and consuming assertions must be aware of these potential privacy concerns and should attempt to address them in their implementations of SAML-aware systems.

2.1 Ensuring Confidentiality

Perhaps the most important aspect of ensuring privacy to parties in a SAML-enabled transaction is the ability to carry out the transaction with a guarantee of confidentiality. In other words, can the information in an assertion be conveyed from the issuer to the intended audience, and only the intended audience, without making it accessible to any other parties?

It is technically possible to convey information confidentially (a discussion of common methods for providing confidentiality occurs in the Security portion of the document in Section 4.2). All parties to SAML-enabled transactions should analyze each of their steps in the interaction (and any subsequent uses of data obtained from the transactions) to ensure that information that should be kept confidential is actually being kept so.

It should also be noted that simply obscuring the contents of assertions may not be adequate protection of privacy. There are many cases where just the availability of the information that a given user (or IP address) was accessing a given service may constitute a breach of privacy (for example, an the information that a user accessed a medical testing facility for an assertion may be enough to breach privacy without knowing the contents of the assertion). Partial solutions to these problems can be provided by various techniques for anonymous interaction, outlined below.

2.2 Notes on Anonymity

The following sections discuss the concept of anonymity.

2.2.1 Definitions That Relate to Anonymity

There are no definitions of anonymity that are satisfying for all cases. Many definitions [Anonymity] deal with the simple case of a sender and a message, and discuss "anonymity" in terms of not being able to link a given sender to a sent message, or a message back to a sender.

And while that definition is adequate for the "one off" case, it ignores the aggregation of information that is possible over time based on behavior rather than an identifier.

Two notions that may be generally useful, and that relate to each other, can help define anonymity.

The first notion is to think about anonymity as being "within a set", as in this comment from "Anonymity, Unobservability, and Pseudonymity" [Anonymity]:

> *To enable anonymity of a subject, there always has to be an appropriate set of subjects with potentially the same attributes....*
>
> *...Anonymity is the stronger, the larger the respective anonymity set is and the more evenly distributed the sending or receiving, respectively, of the subjects within that set is.*

This notion is relevant to SAML because of the use of authorities. Even if a Subject is “anonymous”, that subject is still identifiable as a member of the set of Subjects within the domain of the relevant authority.

In the case where aggregating attributes of the user are provided, the set can become much smaller – for example, if the user is “anonymous” but has the attribute of “student in Course 6@mit.edu”. Certainly, the number of Course 6 students is less than the number of MIT-affiliated persons which is less than the number of users everywhere.

Why does this matter? Non-anonymity leads to the ability of an adversary to harm, as expressed in Dingledine, Freedman, and Molnar’s Freehaven document [FreeHaven]:

> *Both anonymity and pseudonymity protect the privacy of the user's location and true name. Location refers to the actual physical connection to the system. The term “true name”' was introduced by Vinge and popularized by May to refer to the legal identity of an individual. Knowing someone's true name or location allows you to hurt him or her.*

This leads to a unification of the notion of anonymity within a set and ability to harm, from the same source [FreeHaven]:

> *We might say that a system is partially anonymous if an adversary can only narrow down a search for a user to one of a ‘set of suspects.’ If the set is large enough, then it is impractical for an adversary to act as if any single suspect were guilty. On the other hand, when the set of suspects is small, mere suspicion may cause an adversary to take action against all of them.*

SAML-enabled systems are limited to "partial anonymity" at best because of the use of authorities. An entity about whom an assertion is made is already identifiable as one of the pool of entities in a relationship with the issuing authority.

The limitations on anonymity can be much worse than simple authority association, depending on how identifiers are employed, as reuse of pseudonymous identifiers allows accretion of potentially identifying information (see Section 2.2.2). Additionally, users of SAML-enabled systems can also make the breach of anonymity worse by their actions (see Section 2.2.3).

2.2.2 Pseudonymity and Anonymity

Apart from legal identity, any identifier for a Subject can be considered a pseudonym. And even notions like “holder of key” can be considered as serving as the equivalent of a pseudonym in linking an action (or set of actions) to a Subject. Even a description such as “the user that just requested access to object XYZ at time 23:34” can serve as an equivalent of a pseudonym.

Thus, that with respect to “ability to harm,” it makes no difference whether the user is described with an identifier or described by behavior (for example, use of a key or performance of an action).

What does make a difference is how often the particular equivalent of a pseudonym is used.

[Anonymity] gives a taxonomy of pseudonyms starting from personal pseudonyms (like nicknames) that are used all the time, through various types of role pseudonyms (such as Secretary of Defense), on to “one-time-use” pseudonyms.

Only one-time-use pseudonyms can give you anonymity (within SAML, consider this as "anonymity within a set").

The more often you use a given pseudonym, the more you reduce your anonymity and the more likely it is that you can be harmed. In other words, reuse of a pseudonym allows additional potentially identifying information to be associated with the pseudonym. Over time, this will lead to an accretion that can uniquely identify the identity associated with a pseudonym.

2.2.3 Behavior and Anonymity

As Joe Klein can attest, anonymity isn't all it is cracked up to be.

Klein is the "Anonymous" who authored Primary Colors. Despite his denials he was unmasked as the author by Don Foster, a Vassar professor who did a forensic analysis of the text of Primary Colors. Foster compared that text with texts from a list of suspects that he devised based on their knowledge bases and writing proclivities.

It was Klein's idiosyncratic usages that did him in (though apparently all authors have them).

The relevant point for SAML is that an "anonymous" user (even one that is never named) can be identified enough to be harmed by repeated unusual behavior. Here are some examples:

- A user who each Tuesday at 21:00 access a database that correlates finger lengths and life span starts to be non-anonymous. Depending on that user's other behavior, she or he may become "traceable" [Pooling] in that other "identifying" information may be able to be collected.
- A user who routinely buys a usual set of products from a networked vending machine certainly opens themselves to harm (by virtue of booby-trapping the products).

2.2.4 Implications for Privacy

Origin site authorities (such as authentication authorities and attribute authorities) can provide a degree of "partial anonymity" by employing one-time-use identifiers or keys (for the “holder of key” case).

This anonymity is "partial" at best because the Subject is necessarily confined to the set of Subjects in a relationship with the Authority.

This set may be further reduced (thus further reducing anonymity) when aggregating attributes are used that further subset the user community at the origin site.

Users who truly care about anonymity must take care to disguise or avoid unusual patterns of behavior that could serve to “de-anonymize” them over time.

3 Security

The following sections discuss security considerations.

3.1 Background

Communication between computer-based systems is subject to a variety of threats, and these threats carry some level of associated risk. The nature of the risk depends on a host of factors, including the nature of the communications, the nature of the communicating systems, the communication mediums, the communication environment, the end-system environments, and so on. Section 3 of the IETF guidelines on writing security considerations for RFCs [Rescorla-Sec] provides an overview of threats inherent in the Internet (and, by implication, intranets).

SAML is intended to aid deployers in establishing security contexts for application-level computer-based communications within or between security domains. In this role, SAML transfers authentication data, supporting end systems' ability to protect against unauthorized usage. Communications security is directly applicable to the design of SAML. Systems security is of interest mostly in the context of SAML's threat models. Section 2 of the IETF guidelines gives an overview of communications security and systems security.

3.2 Scope

Some areas that impact broadly on the overall security of a system that uses SAML are explicitly outside the scope of SAML. While this document does not address these areas, they should always be considered when reviewing the security of a system. In particular, these issues are important, but currently beyond the scope of SAML:

- Initial authentication: SAML allows statements to be made about acts of authentication that have occurred, but includes no requirements or specifications for these acts of authentication. Consumers of authentication assertions should be wary of blindly trusting these assertions unless and until they know the basis on which they were made. Confidence in the assertions must never exceed the confidence that the asserting party has correctly arrived at the conclusions asserted.
- Trust Model: In many cases, the security of a SAML conversation will depend on the underlying trust model, which is typically based on a key management infrastructure (for example, PKI or secret key). For example, SOAP messages secured by means of XML Signature [XMLSig] are secured only insofar as the keys used in the exchange can be trusted. Undetected compromised keys or revoked certificates, for example, could allow a breach of security. Even failure to require a certificate opens the door for impersonation attacks. PKI setup is not trivial and must be implemented correctly in order for layers built on top of it (such as parts of SAML) to be secure.
- Suitable implementations of security protocols is necessary to maintain the security of a system, including secure random or pseudo-random number generation and secure key storage.

3.3 SAML Threat Model

The general Internet threat model described in the IETF guidelines for security considerations [Rescorla-Sec] is the basis for the SAML threat model. We assume here that the two or more endpoints of a SAML transaction are uncompromised, but that the attacker has complete control over the communications channel.

Additionally, due to the nature of SAML as a multi-party authentication and authorization statement protocol, cases must be considered where one or more of the parties in a legitimate SAML transaction—who operate legitimately within their role for that transaction—attempt to use information gained from a previous transaction maliciously in a subsequent transaction.

The following scenarios describe possible attacks:

- Collusion: The secret cooperation between two or more system entities to launch an attack, for example:

  Collusion between Principal and service provider

  Collusion between Principal and identity provider

  Collusion between identity provider and service provider

  Collusion among two or more Principals

  Collusion between two or more service providers

  Collusion between two or more identity providers

- Denial-of-Service Attacks: The prevention of authorized access to a system resource or the delaying of system operations and functions.
- Man-in-the-Middle Attacks: A form of active wiretapping attack in which the attacker intercepts and selectively modifies communicated data to masquerade as one or more of the entities involved in a communication association.
- Replay Attacks: An attack in which a valid data transmission is maliciously or fraudulently repeated, either by the originator or by an adversary who intercepts the data and retransmits it, possibly as part of a masquerade attack.
- Session Hijacking: A form of active wiretapping in which the attacker seizes control of a previously established communication association.

In all cases, the local mechanisms that systems will use to decide whether or not to generate assertions are out of scope. Thus, threats arising from the details of the original login at an authentication authority, for example, are out of scope as well. If an authority issues a false assertion, then the threats arising from the consumption of that assertion by downstream systems are explicitly out of scope.

The direct consequence of such a scoping is that the security of a system based on assertions as inputs is only as good as the security of the system used to generate those assertions, and of the correctness of the data and processing on which the generated assertions are based. When determining what issuers to trust, particularly in cases where the assertions will be used as inputs to authentication or authorization decisions, the risk of security compromises arising from the consumption of false but validly issued assertions is a large one. Trust policies between asserting and relying parties should always be written to include significant consideration of liability and implementations should provide an appropriate audit trail.

4 Security Techniques

The following sections describe security techniques and various stock technologies available for their implementation in SAML deployments.

4.1 Authentication

Authentication here means the ability of a party to a transaction to determine the identity of the other party in the transaction. This authentication may be in one direction or it may be bilateral.

4.1.1 Active Session

Non-persistent authentication is provided by the communications channel used to transport a SAML message. This authentication may be unilateral—from the session initiator to the receiver—or bilateral. The specific method will be determined by the communications protocol used. For instance, the use of a secure network protocol, such as TLS [RFC2246] or the IP Security Protocol [IPsec], provides the SAML message sender with the ability to authenticate the destination for the TCP/IP environment.

4.1.2 Message-Level

XML Signature [XMLSig] and the OASIS Web Services Security specifications [WSS] provide methods of creating a persistent "authentication" that is tightly coupled to a document. This method does not independently guarantee that the sender of the message is in fact that signer (and indeed, in many cases where intermediaries are involved, this is explicitly not the case).

Any method that allows the persistent confirmation of the involvement of a uniquely resolvable entity with a given subset of an XML message is sufficient to meet this requirement.

4.2 Confidentiality

Confidentiality means that the contents of a message can be read only by the desired recipients and not anyone else who encounters the message.

4.2.1 In Transit

Use of a secure network protocol such as TLS [RFC2246] or the IP Security Protocol [IPsec] provides transient confidentiality of a message as it is transferred between two nodes.

4.2.2 Message-Level

XML Encryption [XMLEnc] provides for the selective encryption of XML documents. This encryption method provides persistent, selective confidentiality of elements within an XML message.

4.3 Data Integrity

Data integrity is the ability to confirm that a given message as received is unaltered from the version of the message that was sent.

4.3.1 In Transit

Use of a secure network protocol such as TLS [RFC2246] or the IP Security Protocol [IPsec] may be configured to provide integrity protection for the packets transmitted via the network connection.

4.3.2 Message-Level

XML Signature [XMLSig] provides a method of creating a persistent guarantee of the unaltered nature of a message that is tightly coupled to that message.

Any method that allows the persistent confirmation of the unaltered nature of a given subset of an XML message is sufficient to meet this requirement.

4.4 Notes on Key Management

Many points in this document will refer to the ability of systems to provide authentication, data integrity, and confidentiality via various schemes involving digital signature and encryption. For all these schemes the security provided by the scheme is limited based on the key management systems that are in place. Some specific limitations are detailed below.

4.4.1 Access to the Key

It is assumed that, if key-based systems are going to be used for authentication, data integrity, and non-repudiation, security is in place to guarantee that access to a private or secret key representing a principal is not available to inappropriate parties. For example, a digital signature created with Bob's private key is only proof of Bob's involvement to the extent that Bob is the only one with access to the key.

In general, access to keys should be kept to the minimum set of entities possible (particularly important for corporate or organizational keys) and should be protected with passphrases and other means. Standard security precautions (don't write down the passphrase, when you're away from a computer don't leave a window with the key accessed open, and so on) apply.

4.4.2 Binding of Identity to Key

For a key-based system to be used for authentication there must be some trusted binding of identity to key. Verifying a digital signature on a document can determine if the document is unaltered since it was signed, and that it was actually signed by a given key. However, this does not confirm that the key used is actually the key of a specific individual appropriate for the time and purpose. Verifying the binding of a key to a party requires additional validation.

This key-to-individual binding must be established. Common solutions include local directories that store both identifiers and key—which is simple to understand but difficult to maintain—or the use of certificates. Using certificates can provide a scalable means to associate a key with an identity, but requires mechanisms to manage the certificate lifecycle and changes to the status of the binding (e.g. An employee leaves and no longer has a corporate identity). One common approach is to use a Public Key Infrastructure (PKI).

In this case a set of trusted root Certifying Authorities (CAs) are identified for each consumer of signatures—answering the question "Whom do I trust to make statements of identity-to-key binding?" Verification of a signature then becomes a process of first verifying the signature (to determine that the signature was done by the key in question and that the message has not changed) and then validating the certificate chain (to determine that the key is bound to the right identity) and validating that the binding is still appropriate. Validating the binding requires steps to be taken to ensure that the binding is currently valid—a certificate typically has a "lifetime" built into it, but if a key is compromised during the life of the certificate then the key-to-identity binding contained in the certificate becomes invalid while the certificate is still valid on its face. Also, certificates often depend on associations that may end before their lifetime expires (for example, certificates that should become invalid when someone changes employers, etc.) Different mechanisms may be used to validate key and certificate validity, such as Certificate Revocation Lists (CRLs), the Online Certificate Status Protocol [OCSP], or the XML Key Management Specification (XKMS) [XKMS], but these mechanisms are out of scope of the SSTC work.

A proper key management system is thus quite strong but very complex. Verifying a signature ends up being a process of verifying the document-to-key binding, then verifying the key-to-identity binding, as well as the current validity of the key and certificate.

4.5 SSL/TLS Cipher Suites

The use of HTTP over SSL 3.0 or TLS 1.0 [RFC2246] , or use of URLs with the HTTPS URL scheme, is strongly recommended at many places in this document.

Unless otherwise specified, in any SAML binding's use of SSL 3.0 [SSL3] or TLS 1.0 [RFC2246], servers MUST authenticate to clients using a X.509 v3 certificate. The client MUST establish server identity based on contents of the certificate (typically through examination of the certificate's subject DN field).

SSL/TLS can be configured to use many different cipher suites, not all of which are adequate to provide "best practices" security. The following sections provide a brief description of cipher suites and recommendations for cipher suite selection.

4.5.1 SSL/TLS Cipher Suites

> **Note:** While references to the US Export restrictions are now obsolete, the constants naming the cipher suites have not changed. Thus, SSL_DHE_DSS_EPORT_WITH_DES40_CBC_SHA is still a valid cipher suite identifier, and the explanation of the historical reasons for the inclusion of "EXPORT" has been left in place in the following summary.

A cipher suite combines four kinds of security features, and is given a name in the SSL protocol specification. Before data flows over a SSL connection, both ends attempt to negotiate a cipher suite. This lets them establish an appropriate quality of protection for their communications, within the constraints of the particular mechanism combinations which are available. The features associated with a cipher suite are:

- The protocol, SSL or TLS.
- The type of key exchange algorithm used. SSL defines many; the ones that provide server authentication are the most important ones, but anonymous key exchange is supported. (Note that anonymous key exchange algorithms are subject to "man in the middle" attacks, and are **not recommended** in the SAML context.) The "RSA" authenticated key exchange algorithm is currently the most interoperable algorithm. Another important key exchange algorithm is the authenticated Diffie-Hellman "DHE_DSS" key exchange, which has no patent-related implementation constraints.[1]
- Whether the key exchange algorithm is freely exportable from the United States of America. Exportable algorithms must use short (512-bit) public keys for key exchange and short (40-bit) symmetric keys for encryption. Keys of these lengths have been successfully attacked, and their use is not recommended.
- The encryption algorithm used. The fastest option is the RC4 stream cipher; DES and variants (DES40, 3DES-EDE) as well as AES are also supported in "cipher block chaining" (CBC) mode. Other modes are also supported, refer to the TLS documentation [RFC2246].
- Null encryption is also an option in some cipher suites. Note that null encryption performs **no** encryption; in such cases SSL/TLS is used only to authenticate and provide integrity protection. Cipher suites with null encryption do not provide confidentiality, and **must not be used** in cases where confidentiality is a requirement and is not obtained by means other than SSL/TLS.
- The digest algorithm used for the Message Authentication Code. The recommended choice is SHA1.
- For example, the cipher suite named SSL_DHE_DSS_EXPORT_WITH_DES40_CBC_SHA uses SSL, uses an authenticated Diffie-Hellman key exchange (DHE_DSS), is export grade (EXPORT), uses an exportable variant of the DES cipher (DES40_CBC), and uses the SHA1 digest algorithm in its MAC (SHA).

A given implementation of SSL will support a particular set of cipher suites, and some subset of those will be enabled by default. Applications have a limited degree of control over the cipher suites that are used on their connections; they can enable or disable any of the supported cipher suites, but cannot change the cipher suites that are available.

1 [1] The RSA algorithm patent has expired; hence this issue is mostly historical.

4.5.2 SSL/TLS Recommendations

SSL 2.0 must not be used due to known security weaknesses. TLS is preferred, SSL 3.0 may also be used.

The SAML 2.0 Bindings specification outlines which cipher suites are required and recommended, making normative statements. This section repeats this information for completeness, but that specification is considered normative in case of inconsistency.

TLS-capable implementations MUST implement the TLS_RSA_WITH_3DES_EDE_CBC_SHA cipher suite and MAY implement the TLS_RSA_WITH_AES_128_CBC_SHA cipher suite.

FIPS TLS-capable implementations MUST implement the corresponding TLS_RSA_FIPS_WITH_3DES_EDE_CBC_SHA cipher suite and MAY implement the corresponding TLS_RSA_FIPS_AES_128_CBC_SHA cipher suite [FIPS].

SSL-capable implementations MUST implement the SSL_RSA_WITH_3DES_EDE_CBC_SHA cipher suite.

FIPS SSL-capable implementations MUST implement the FIPS ciphersuite corresponding to the SSL SSL_RSA_WITH_3DES_EDE_CBC_SHA cipher suite [FIPS].

However, the IETF is moving rapidly towards mandating the use of AES, which has both speed and strength advantages. Forward-looking systems would be wise as well to implement support for the AES cipher suites, such as:

- TLS_RSA_WITH_AES_128_CBC_SHA

5 General SAML Security Considerations

The following sections analyze the security risks in using and implementing SAML and describe countermeasures to mitigate the risks.

5.1 SAML Assertions

At the level of the SAML assertion itself, there is little to be said about security concerns—most concerns arise during communications in the request/response protocol, or during the attempt to use SAML by means of one of the bindings. The consumer is, of course, always expected to honor the validity interval of the assertion and any `<OneTimeUse>` elements that are present in the assertion.

However, one issue at the assertion level bears analysis: an assertion, once issued, is out of the control of the issuer. This fact has a number of ramifications. For example, the issuer has no control over how long the assertion will be persisted in the systems of the consumer; nor does the issuer have control over the parties with whom the consumer will share the assertion information. These concerns are over and above concerns about a malicious attacker who can see the contents of assertions that pass over the wire unencrypted (or insufficiently encrypted).

While efforts have been made to address many of these issues within the SAML specification, nothing contained in the specification will erase the requirement for careful consideration of what to put in an assertion. At all times, issuers should consider the possible consequences if the information in the assertion is stored on a remote site, where it can be directly misused, or exposed to potential hackers, or possibly stored for more creatively fraudulent uses. Issuers should also consider the possibility that the information in the assertion could be shared with other parties, or even made public, either intentionally or inadvertently.

5.2 SAML Protocol

The following sections describe security considerations for the SAML request-response protocol itself, apart from any threats arising from use of a particular protocol binding.

5.2.1 Denial of Service

The SAML protocol is susceptible to a denial of service (DOS) attack. Handling a SAML request is potentially a very expensive operation, including parsing the request message (typically involving construction of a DOM tree), database/assertion store lookup (potentially on an unindexed key), construction of a response message, and potentially one or more digital signature operations. Thus, the effort required by an attacker generating requests is much lower than the effort needed to handle those requests.

5.2.1.1 Requiring Client Authentication at a Lower Level

Requiring clients to authenticate at some level below the SAML protocol level (for example, using the SOAP over HTTP binding, with HTTP over TLS/SSL, and with a requirement for client-side certificates that have a trusted Certificate Authority at their root) will provide traceability in the case of a DOS attack.

If the authentication is used only to provide traceability, then this does not in itself prevent the attack from occurring, but does function as a deterrent.

If the authentication is coupled with some access control system, then DOS attacks from non-insiders is effectively blocked. (Note that it is possible that overloading the client-authentication scheme could still function as a denial-of-service attack on the SAML service, but that this attack needs to be dealt with in the context of the client authentication scheme chosen.)

Whatever system of client authentication is used, it should provide the ability to resolve a unique originator for each request, and should not be subject to forgery. (For example, in the traceability-only case, logging the IP address is insufficient since this information can easily be spoofed.)

5.2.1.2 Requiring Signed Requests

In addition to the benefits gained from client authentication discussed in Section 5.2.1.1, requiring a signed request also lessens the order of the asymmetry between the work done by requester and responder. The additional work required of the responder to verify the signature is a relatively small percentage of the total work required of the responder, while the process of calculating the digital signature represents a relatively large amount of work for the requester. Narrowing this asymmetry decreases the risk associated with a DOS attack.

Note, however, that an attacker can theoretically capture a signed message and then replay it continually, getting around this requirement. This situation can be avoided by requiring the use of the XML Signature element `<ds:SignatureProperties>` containing a timestamp; the timestamp can then be used to determine if the signature is recent. In this case, the narrower the window of time after issue that a signature is treated as valid, the higher security you have against replay denial of service attacks.

5.2.1.3 Restricting Access to the Interaction URL

Limiting the ability to issue a request to a SAML service at a very low level to a set of known parties drastically reduces the risk of a DOS attack. In this case, only attacks originating from within the finite set of known parties are possible, greatly decreasing exposure both to potentially malicious clients and to DOS attacks using compromised machines as zombies.

There are many possible methods of limiting access, such as placing the SAML responder inside a secured intranet and implementing access rules at the router level.

6 SAML Bindings Security Considerations

The security considerations in the design of the SAML request-response protocol depend to a large extent on the particular protocol binding (as defined in the SAML bindings specification [SAMLBind]) that is used. The bindings sanctioned by the OASIS Security Services Technical Committee are the SOAP binding, Reverse SOAP Binding (PAOS), HTTP Redirect binding, HTTP Redirect/POST binding and HTTP Artifact binding and SAML URI bindings.

6.1 SAML SOAP Binding

Since the SAML SOAP binding requires no authentication and has no requirements for either in-transit confidentiality or message integrity, it is open to a wide variety of common attacks, which are detailed in the following sections. General considerations are discussed separately from considerations related to the SOAP-over-HTTP case.

6.1.1 Eavesdropping

Threat: Since there is no in-transit confidentiality requirement, it is possible that an eavesdropping party could acquire both the SOAP message containing a request and the SOAP message containing the corresponding response. This acquisition exposes both the nature of the request and the details of the response, possibly including one or more assertions.

Exposure of the details of the request will in some cases weaken the security of the requesting party by revealing details of what kinds of assertions it requires, or from whom those assertions are requested. For example, if an eavesdropper can determine that site *X* is frequently requesting authentication assertions with a given confirmation method from site *Y*, he may be able to use this information to aid in the compromise of site *X*.

Similarly, eavesdropping on a series of authorization queries could create a “map” of resources that are under the control of a given authorization authority.

Additionally, in some cases exposure of the request itself could constitute a violation of privacy. For example, eavesdropping on a query and its response may expose that a given user is active on the querying site, which could be information that should not be divulged in cases such as medical information sites, political sites, and so on. Also the details of any assertions carried in the response may be information that should be kept confidential. This is particularly true for responses containing attribute assertions; if these attributes represent information that should not be available to entities not party to the transaction (credit ratings, medical attributes, and so on), then the risk from eavesdropping is high.

Countermeasures: In cases where any of these risks is a concern, the countermeasure for eavesdropping attacks is to provide some form of in-transit message confidentiality. For SOAP messages, this confidentiality can be enforced either at the SOAP level or at the SOAP transport level (or some level below it).

Adding in-transit confidentiality at the SOAP level means constructing the SOAP message such that, regardless of SOAP transport, no one but the intended party will be able to access the message. The general solution to this problem is likely to be XML Encryption [XMLEnc]. This specification allows encryption of the SOAP message itself, which eliminates the risk of eavesdropping unless the key used in the encryption has been compromised. Alternatively, deployers can depend on the SOAP transport layer, or a layer beneath it, to provide in-transit confidentiality.

The details of how to provide this confidentiality depend on the specific SOAP transport chosen. Using HTTP over TLS/SSL (described further in Section 6.1.7) is one method. Other transports will necessitate other in-transit confidentiality techniques; for example, an SMTP transport might use S/MIME.

In some cases, a layer beneath the SOAP transport might provide the required in-transit confidentiality. For example, if the request-response interaction is carried out over an IPsec tunnel, then adequate in-transit confidentiality may be provided by the tunnel itself.

6.1.2 Replay

Threat: There is little vulnerability to replay attacks at the level of the SOAP binding. Replay is more of an issue in the various profiles. The primary concern about replay at the SOAP binding level is the potential for use of replay as a denial-of-service attack method.

Countermeasures: In general, the best way to prevent replay attacks is to prevent the message capture in the first place. Some of the transport-level schemes used to provide in-transit confidentiality will accomplish this goal. For example, if the SAML request-response conversation occurs over SOAP on HTTP/TLS, third parties are prevented from capturing the messages.

Note that since the potential replayer does not need to understand the message to replay it, schemes such as XML Encryption do not provide protection against replay. If an attacker can capture a SAML request that has been signed by the requester and encrypted to the responder, then the attacker can replay that request at any time without needing to be able to undo the encryption. The SAML request includes information about the issue time of the request, allowing a determination about whether replay is occurring. Alternatively, the unique key of the request (its `ID`) can be used to determine if this is a replay request or not.

Additional threats from the replay attack include cases where a “charge per request” model is in place. Replay could be used to run up large charges on a given account.

Similarly, models where a client is allocated (or purchases) a fixed number of interactions with a system, the replay attack could exhaust these uses unless the issuer is careful to keep track of the unique key of each request.

6.1.3 Message Insertion

Threat: A fabricated request or response is inserted into the message stream. A false response such as a spurious “yes” reply to an authorization decision query or the return of false attribute information in response to an attribute query may result in inappropriate receiver action.

Countermeasures: The ability to insert a request is not a threat at the SOAP binding level. The threat of inserting a false response can be a denial of service attack, for example returning SOAP Faults for responses, but this attack would become quickly obvious. The more subtle attack of returning fabricated responses is addressed in the SAML protocol, appropriate since according to the SOAP Binding definition each SOAP response must contain a single SAML protocol response unless it contains a fault. The SAML Protocol addresses this with two mechanisms, correlation of responses to requests using the required InResponseTo attribute, making an attack harder since requests must be intercepted to generate responses, and through the support origin authentication, either via signed SAML responses or through a secured transport connection such as SSL/TLS.

6.1.4 Message Deletion

Threat: The message deletion attack would either prevent a request from reaching a responder, or would prevent the response from reaching the requester.

Countermeasures: In either case, the SOAP binding does not address this threat. In general, correlation of request and response messages may deter such an attack, for example use of the InResponseTo attribute in the SAMLResponseType.

6.1.5 Message Modification

Threat: Message modification is a threat to the SOAP binding in both directions.

Modification of the request to alter the details of the request can result in significantly different results being returned, which in turn can be used by a clever attacker to compromise systems depending on the assertions returned. For example, altering the list of requested attributes in the `<Attribute>` elements could produce results leading to compromise or rejection of the request by the responder.

Modification of the request to alter the apparent issuer of the request could result in denial of service or incorrect routing of the response. This alteration would need to occur below the SAML level and is thus out of scope.

Modification of the response to alter the details of the assertions therein could result in vast degrees of compromise. The simple examples of altering details of an authentication or an authorization decision could lead to very serious security breaches.

Countermeasures: In order to address these potential threats, a system that guarantees in-transit message integrity must be used. The SAML protocol and the SOAP binding neither require nor forbid the deployment of systems that guarantee in-transit message integrity, but due to this large threat, it is **highly recommended** that such a system be used. At the SOAP binding level, this can be accomplished by digitally signing requests and responses with a system such as XML Signature [XMLSig]. The SAML specification allows for such signatures; see the SAML assertion and protocol specification [SAMLCore] for further information.

If messages are digitally signed (with a sensible key management infrastructure, see Section 4.4) then the recipient has a guarantee that the message has not been altered in transit, unless the key used has been compromised.

The goal of in-transit message integrity can also be accomplished at a lower level by using a SOAP transport that provides the property of guaranteed integrity, or is based on a protocol that provides such a property. SOAP over HTTP over TLS/SSL is a transport that would provide such a guarantee.

Encryption alone does not provide this protection, as even if the intercepted message could not be altered per se, it could be replaced with a newly created one.

6.1.6 Man-in-the-Middle

Threat: The SOAP binding is susceptible to man-in-the-middle (MITM) attacks. In order to prevent malicious entities from operating as a man in the middle (with all the perils discussed in both the eavesdropping and message modification sections), some sort of bilateral authentication is required.

Countermeasures: A bilateral authentication system would allow both parties to determine that what they are seeing in a conversation actually came from the other party to the conversation.

At the SOAP binding level, this goal could also be accomplished by digitally signing both requests and responses (with all the caveats discussed in Section 6.1.5 above). This method does not prevent an eavesdropper from sitting in the middle and forwarding both ways, but he is prevented from altering the conversation in any way without being detected.

Since many applications of SOAP do not use sessions, this sort of authentication of author (as opposed to authentication of sender) may need to be combined with information from the transport layer to confirm that the sender and the author are the same party in order to prevent a weaker form of “MITM as eavesdropper”.

Another implementation would depend on a SOAP transport that provides, or is implemented on a lower layer that provides, bilateral authentication. The example of this is again SOAP over HTTP over TLS/SSL with both server- and client-side certificates required.

Additionally, the validity interval of the assertions returned functions as an adjustment on the degree of risk from MITM attacks. The shorter the valid window of the assertion, the less damage can be done if it is intercepted.

6.1.7 Use of SOAP over HTTP

Since the SOAP binding requires that conformant applications support HTTP over TLS/SSL with a number of different bilateral authentication methods such as Basic over server-side SSL and certificate-backed authentication over server-side SSL, these methods are always available to mitigate threats in cases where other lower-level systems are not available and the above listed attacks are considered significant threats.

This does not mean that use of HTTP over TLS with some form of bilateral authentication is mandatory. If an acceptable level of protection from the various risks can be arrived at through other means (for example, by an IPsec tunnel), full TLS with certificates is not required. However, in the majority of cases for SOAP over HTTP, using HTTP over TLS with bilateral authentication will be the appropriate choice.

The HTTP Authentication RFC [RFC2617]describes possible attacks in the HTTP environment when basic or message-digest authentication schemes are used.

Note, however, that the use of transport-level security (such as the SSL or TLS protocols under HTTP) only provides confidentiality and/or integrity and/or authentication for “one hop”. For models where there may be intermediaries, or the assertions in question need to live over more than one hop, the use of HTTP with TLS/SSL does not provide adequate security.

6.2 Reverse SOAP (PAOS) Binding

6.2.1 Denial of Service

Threat: Remove HTTP accept header field and/or the PAOS HTTP header field causing HTTP responder to ignore PAOS processing possibility.

Countermeasures: Integrity protect the HTTP message, using SSL/TLS integrity protection or other adequate transport layer security mechanism.

6.3 HTTP Redirect binding

6.3.1 Denial of Service

Threat: Malicious redirects into identity or service provider targets

Description: A spurious entity could issue a redirect to a user agent so that the user agent would access a resource that disrupts single sign-on. For example, an attacker could redirect the user agent to a logout resource of a service provider causing the Principal to be logged out of all existing authentication sessions.

Countermeasures: Access to resources that produce side effects could be specified with a transient qualifier that must correspond to the current authentication session. Alternatively, a confirmation dialog could be interposed that relies on a transient qualifier with similar semantics.

6.4 HTTP Redirect/POST binding

This section utilizes materials from [ShibMarlena and [Rescorla-Sec] and is derived from material in the SAML 1.1 Bindings and Profiles specification [SAML11Bind].

6.4.1 Stolen Assertion

Threat: If an eavesdropper can copy the real user’s SAML response and included assertions, then the eavesdropper could construct an appropriate POST body and be able to impersonate the user at the destination site.

Countermeasures: Confidentiality MUST be provided whenever a response is communicated between a site and the user’s browser. This provides protection against an eavesdropper obtaining a real user’s SAML response and assertions.

If an eavesdropper defeats the measures used to ensure confidentiality, additional countermeasures are available:

- The Identity Provider and Service Provider sites SHOULD make some reasonable effort to ensure that clock settings at both sites differ by at most a few minutes. Many forms of time synchronization service are available, both over the Internet and from proprietary sources.
- When a non-SSO SAML profile uses the POST binding it must ensure that the receiver can perform timely subject confirmation. To this end, a SAML authentication assertion for the principal MUST be included in the POSTed form response.
- Values for `NotBefore` and `NotOnOrAfter` attributes of SSO assertions SHOULD have the shortest possible validity period consistent with successful communication of the assertion from Identity Provider to Service Provider site. This is typically on the order of a few minutes. This ensures that a stolen assertion can only be used successfully within a small time window.
- The Service Provider site MUST check the validity period of all assertions obtained from the

Identity Provider site and reject expired assertions. A Service Provider site MAY choose to implement a stricter test of validity for SSO assertions, such as requiring the assertion's `IssueInstant` or `AuthenticationInstant` attribute value to be within a few minutes of the time at which the assertion is received at the Service Provider site.

- If a received authentication statement includes a `<saml:SubjectLocality>` element with the IP address of the user, the Service Provider site MAY check the browser IP address against the IP address contained in the authentication statement.

6.4.2 Man In the Middle Attack

Threat: Since the Service Provider site obtains bearer SAML assertions from the user by means of an HTML form, a malicious site could impersonate the user at some new Service Provider site. The new Service Provider site would believe the malicious site to be the subject of the assertion.

Countermeasures: The Service Provider site MUST check the Recipient attribute of the SAML response to ensure that its value matches the https://`<assertion consumer host name and path>`. As the response is digitally signed, the `Recipient` value cannot be altered by the malicious site.

6.4.3 Forged Assertion

Threat: A malicious user, or the browser user, could forge or alter a SAML assertion.

Countermeasures: The browser/POST profile requires the SAML response carrying SAML assertions to be signed, thus providing both message integrity and authentication. The Service Provider site MUST verify the signature and authenticate the issuer.

6.4.4 Browser State Exposure

Threat: The browser/POST profile involves uploading of assertions from the web browser to a Service Provider site. This information is available as part of the web browser state and is usually stored in persistent storage on the user system in a completely unsecured fashion. The threat here is that the assertion may be "reused" at some later point in time.

Countermeasures: Assertions communicated using this profile must always have short lifetimes and should have a <OneTimeUse> SAML assertion <Conditions> element. Service Provider sites are expected to ensure that the assertions are not re-used.

6.4.5 Replay

Threat: Replay attacks amount to resubmission of the form in order to access a protected resource fraudulently.

Countermeasures: The profile mandates that the assertions transferred have the one-use property at the Service Provider site, preventing replay attacks from succeeding.

6.4.6 Modification or Exposure of state information

Threat: Relay state tampering or fabrication

Some of the messages may carry a <RelayState> element, which is recommended to be integrity-protected by the producer and optionally confidentiality- protected. If these practices are not followed, an adversary could trigger unwanted side effects. In addition, by not confidentiality-protecting the value of this element, a legitimate system entity could inadvertently expose information to the identity provider or a passive attacker.

Countermeasure: Follow the recommended practice of confidentiality- and integrity- protecting the RelayState data. Note: Because the value of this element is both produced and consumed by the same system entity, symmetric cryptographic primitives could be utilized

6.5 HTTP Artifact Binding

This section utilizes materials from [ShibMarlena and [Rescorla-Sec] and is derived from material in the SAML 1.1 Bindings and Profiles specification [SAML11Bind].

6.5.1 Stolen Artifact

Threat: If an eavesdropper can copy the real user's SAML artifact, then the eavesdropper could construct a URL with the real user's SAML artifact and be able to impersonate the user at the destination site.

Countermeasures: Confidentiality MUST be provided whenever an artifact is communicated between a site and the user's browser. This provides protection against an eavesdropper gaining access to a real user's SAML artifact.

If an eavesdropper defeats the measures used to ensure confidentiality, additional countermeasures are available:

- The source and destination sites SHOULD make some reasonable effort to ensure that clock settings at both sites differ by at most a few minutes. Many forms of time synchronization service are available, both over the Internet and from proprietary sources.
- The source site SHOULD track the time difference between when a SAML artifact is generated and placed on a URL line and when a `<samlp:Request>` message carrying the artifact is received from the destination. A maximum time limit of a few minutes is recommended. Should an assertion be requested by a destination site query beyond this time limit, the source site MUST not provide the assertions to the destination site.
- It is possible for the source site to create SSO assertions either when the corresponding SAML artifact is created or when a `<samlp:Request>` message carrying the artifact is received from the destination. The validity period of the assertion SHOULD be set appropriately in each case: longer for the former, shorter for the latter.
- Values for `NotBefore` and `NotOnOrAfter` attributes of SSO assertions SHOULD have the shortest possible validity period consistent with successful communication of the assertion from source to destination site. This is typically on the order of a few minutes. This ensures that a stolen artifact can only be used successfully within a small time window.
- The destination site MUST check the validity period of all assertions obtained from the source site and reject expired assertions. A destination site MAY choose to implement a stricter test of validity for SSO assertions, such as requiring the assertion's `IssueInstant` or `AuthenticationInstant` attribute value to be within a few minutes of the time at which the assertion is received at the destination site.
- If a received authentication statement includes a `<saml:SubjectLocality>` element with the IP address of the user, the destination site MAY check the browser IP address against the IP address contained in the authentication statement.

6.5.2 Attacks on the SAML Protocol Message Exchange

Threat: The message exchange used by the Service Provider to obtain an assertion from the Identity Provider could be attacked in a variety of ways, including artifact or assertion theft, replay, message insertion or modification, and MITM (man-in-the-middle attack).

Countermeasures: The requirement for the use of a SAML protocol binding with the properties of bilateral authentication, message integrity, and confidentiality defends against these attacks.

6.5.3 Malicious Destination Site

Threat: Since the Service Provider obtains artifacts from the user, a malicious site could impersonate the user at some new Service Provider site. The new Service Provider site would obtain assertions from the Identity Provider site and believe the malicious site to be the user.

Countermeasures: The new Service Provider site will need to authenticate itself to the Identity Provider site so as to obtain the SAML assertions corresponding to the SAML artifacts. There are two cases to consider:

1. If the new Service Provider site has no relationship with the Identity Provider site, it will be unable to authenticate and this step will fail.
2. If the new Service Provider site has an existing relationship with the Identity Provider site, the Identity Provider site will determine that assertions are being requested by a site other than that to which the artifacts were originally sent. In such a case, the Identity Provider site MUST not provide the assertions to the new Service Provider site.

6.5.4 Forged SAML Artifact

Threat: A malicious user could forge a SAML artifact.

Countermeasures: The Bindings specification provides specific recommendations regarding the construction of a SAML artifact such that it is infeasible to guess or construct the value of a current, valid, and outstanding assertion handle. A malicious user could attempt to repeatedly "guess" a valid SAML artifact value (one that corresponds to an existing assertion at a Identity Provider site), but given the size of the value space, this action would likely require a very large number of failed attempts. An Identity Provider site SHOULD implement measures to ensure that repeated attempts at querying against non-existent artifacts result in an alarm.

6.5.5 Browser State Exposure

Threat: The SAML browser/artifact profile involves "downloading" of SAML artifacts to the web browser from an Identity Provider site. This information is available as part of the web browser state and is usually stored in persistent storage on the user system in a completely unsecured fashion. The threat here is that the artifact may be "reused" at some later point in time.

Countermeasures: The "one-use" property of SAML artifacts ensures that they cannot be reused from a browser. Due to the recommended short lifetimes of artifacts and mandatory SSO assertions, it is difficult to steal an artifact and reuse it from some other browser at a later time.

6.5.6 Replay

Threat: Reuse of an artifact by repeating protocol messages

Countermeasures: The threat of replay as a reuse of an artifact is addressed by the requirement that each artifact is a one-time-use item. Systems should track cases where multiple requests are made referencing the same artifact, as this situation may represent intrusion attempts.

The threat of replay on the original request that results in the assertion generation is not addressed by SAML, but should be mitigated by the original authentication process.

6.6 SAML URI Binding

6.6.1 Substitution

Threat: Substitution of assertion with another by substitution of URI reference. Given that a URI is opaque to the receiver it is hard to validate the integrity.

Countermeasures: Where this is a concern, transport layer integrity protection such as with SSL/.TLS is required.

7 SAML Profile Security Considerations

The SAML profiles specification [SAMLProf] defines profiles of SAML, which are sets of rules describing how to embed SAML assertions into and extract them from a framework or protocol.

7.1 Web Browser Single Sign-On (SSO) Profiles

Note that user authentication at the source site is explicitly out of scope, as are issues related to this source site authentication. The key notion is that the source system entity must be able to ascertain that the authenticated client system entity that it is interacting with is the same as the one in the next interaction step. One way to accomplish this is for these initial steps to be performed using TLS as a session layer underneath the protocol being used for this initial interaction (likely HTTP).

7.1.1 SSO Profile

7.1.1.1 Eavesdropping

Threat: The possibility of eavesdropping exists in all web browser cases.

Countermeasures: In cases where confidentiality is required (bearing in mind that any assertion that is not sent securely, along with the requests associated with it, is available to the malicious eavesdropper), HTTP traffic needs to take place over a transport that ensures confidentiality. HTTP over TLS/SSL [RFC2246] and the IP Security Protocol [IPsec] meet this requirement.

The following sections provide more detail on the eavesdropping threat.

7.1.1.2 Theft of the User Authentication Information

Threat: In the case where the subject authenticates to the source site by revealing reusable authentication information, for example, in the form of a password, theft of the authentication information will enable an adversary to impersonate the subject.

Countermeasures: In order to avoid this problem, the connection between the subject's browser and the source site must implement a confidentiality safeguard. In addition, steps must be taken by either the subject or the destination site to ensure that the source site is genuinely the expected and trusted source site before revealing the authentication information. Using HTTP over TLS can be used to address this concern.

7.1.1.3 Theft of the Bearer Token

Threat: In the case where the authentication assertion contains the assertion bearer's authentication protocol identifier, theft of the artifact will enable an adversary to impersonate the subject.

Countermeasures: Each of the following methods decreases the likelihood of this happening:

- The destination site implements a confidentiality safeguard on its connection with the subject's browser.
- The subject or destination site ensures (out of band) that the source site implements a confidentiality safeguard on its connection with the subject's browser.
- The destination site verifies that the subject's browser was directly redirected by a source site that directly authenticated the subject.
- The source site refuses to respond to more than one request for an assertion corresponding to the same assertion ID.
- If the assertion contains a condition element of type **AudienceRestrictionType** that identifies a specific domain, then the destination site verifies that it is a member of that domain.

- The connection between the destination site and the source site, over which the assertion ID is passed, is implemented with a confidentiality safeguard.
- The destination site, in its communication with the source site, over which the assertion ID is passed, must verify that the source site is genuinely the expected and trusted source site.

7.1.1.4 Replay

The possibility of a replay attack exists for this set of profiles. A replay attack can be used either to attempt to deny service or to retrieve information fraudulently. The specific countermeasures depend on which specific binding is used and are discussed above

7.1.1.5 Message Insertion

Message insertion attacks are discussed in the section on bindings.

7.1.1.6 Message Deletion

Threat: Deleting a message during any step of the interactions between the browser, SAML assertion issuer, and SAML assertion consumer will cause the interaction to fail. It results in a denial of some service but does not increase the exposure of any information.

Countermeasures: Use of an integrity protected transport channel addresses the threat of message deletion when no intermediaries are present.

7.1.1.7 Message Modification

Threat: The possibility of alteration of the messages in the stream exists for this set of profiles. Some potential undesirable results are as follows:

- Alteration of the initial request can result in rejection at the SAML issuer, or creation of an artifact targeted at a different resource than the one requested
- Alteration of the artifact can result in denial of service at the SAML consumer.
- Alteration of the assertions themselves while in transit could result in all kinds of bad results (if they are unsigned) or denial of service (if they are signed and the consumer rejects them).

Countermeasures:

To avoid message modification, the traffic needs to be transported by means of a system that guarantees message integrity from endpoint to endpoint.

For the web browser-based profiles, the recommended method of providing message integrity in transit is the use of HTTP over TLS/SSL with a cipher suite that provides data integrity checking.

7.1.1.8 Man-in-the-Middle

Threat: Man-in-the-middle attacks are particularly pernicious for this set of profiles. The MITM can relay requests, capture the returned assertion (or artifact), and relay back a false one. Then the original user cannot access the resource in question, but the MITM can do so using the captured resource.

Countermeasures: Preventing this threat requires a number of countermeasures. First, using a system that provides strong bilateral authentication will make it much more difficult for a MITM to insert himself into the conversation.

However the possibility still exists of a MITM who is purely acting as a bidirectional port forwarder, and eavesdropping on the information with the intent to capture the returned assertion or handler (and possibly alter the final return to the requester). Putting a confidentiality system in place will prevent eavesdropping. Putting a data integrity system in place will prevent alteration of the message during port forwarding.

For this set of profiles, all the requirements of strong bilateral session authentication, confidentiality, and data integrity can be met by the use of HTTP over TLS/SSL if the TLS/SSL layer uses an appropriate cipher suite (strong enough encryption to provide confidentiality, and supporting data integrity) and requires X509v3 certificates for authentication.

7.1.1.9 Impersonation without Reauthentication

Threat: Rogue user attempts to impersonate currently logged-in legitimate Principal and thereby gain access to protected resources.

Once a Principal is successfully logged into an identity provider, subsequent <AuthnRequest> messages from different service providers concerning that Principal will not necessarily cause the Principal to be reauthenticated. Principals must, however, be authenticated unless the identity provider can determine that an <AuthnRequest> is associated not only with the Principal's identity, but also with a validly authenticated identity provider session for that Principal.

Countermeasures: In implementations where this threat is a concern, identity providers MUST maintain state information concerning active sessions, and MUST validate the correspondence between an `<AuthnRequest>` and an active session before issuing an `<AuthnResponse>` without first authenticating the Principal. Cookies posted by identity providers MAY be used to support this validation process, though Liberty does not mandate a cookie-based approach.

7.1.2 Enhanced Client and Proxy Profile

7.1.2.1 Man in the Middle

Threat: Intercept AuthnRequest and AuthnResponse SOAP messages, allowing subsequent Principal impersonation.

A spurious system entity can interject itself as a man-in-the-middle (MITM) between the enhanced client and a legitimate service provider, where it acts in the service provider role in interactions with the enhanced client and in the enhanced client role in interactions with the legitimate service provider. In this way, as a first step, the MITM is able to intercept the service provider's AuthnRequest and substitute any URL of its choosing for the responseConsumerServiceURL value in the PAOS header block before forwarding the AuthnRequest on to the enhanced client. Typically, the MITM will insert a URL value that points back to itself. Then, if the enhanced client subsequently receives an AuthnResponse from the identity provider and subsequently sends the contained AuthnResponse to the responseConsumerServiceURL received from the MITM, the MITM will be able to masquerade as the Principal at the legitimate service provider.

Countermeasure: The identity provider specifies to the enhanced client the address to which the enhanced client must send the :AuthnResponse. The responseConsumerServiceURL in the PAOS header is only used for error responses from the enhanced client – as specified in the profile.

7.1.2.2 Denial of Service

Threat: Change an AuthenRequest SOAP request so that it cannot be processed, such as by changing the PAOS header block service attribute value to an unknown value or by changing the ECP header block ProviderID or IDPList to cause the request to fail.

Countermeasures: Provide integrity protection for the SOAP message, by using SOAP Message Security or SSL/TLS.

7.1.3 Identity Provider Discovery Profile

Threat: Cookie poisoning attack, where parameters within the cookie are modified, to cause discovery of an fraudulent identity provider for example.

Countermeasures: The specific mechanism of using a common domain limiits the feaibility of this threat.

7.1.4 Single Logout Profile

Threat: Passive attacker can collect a Principal's name identifier

During the initial steps, a passive attacker can collect the <LogoutRequest> information when it is issued in the redirect. Exposing these data poses a privacy threat.

Countermeasures: All exchanges should be conducted over a secure transport such as SSL or TLS.

Threat: Unsigned `<LogoutRequest>` message

An Unsigned `<LogoutRequest>` could be injected by a spurious system entity thus denying service to the Principal. Assuming that the NameIdentifier can be deduced or derived then it is conceivable that the user agent could be directed to deliver a fabricated `<LogoutRequest>` message.

Countermeasures: Sign the `<LogoutRequest>` message. The identity provider can also verify the identity of a Principal in the absence of a signed request.

7.2 Name Identifier Management Profiles

Threat: Allow system entities to correlate information or otherwise inappropriately expose identity information, harming privacy.

Countermeasures: IDP must take care to use different name identifiers with different service providers for same principal. The IDP SHOULD encrypt the name identifier it returns to the service provider, allowing subsequent interactions to use an opaque identifier.

7.3 Attribute Profiles

Threats related to bindings associated with attribute profiles are discussed above. No additional profile-specific threats are known.

8 Summary

Security and privacy must be addressed in a systemic manner, considering human issues such as social engineering attacks, policy issues, key management and trust management, secure implementation and other factors outside the scope of this document. Security technical solutions have a cost, so requirements and policy alternatives must also be considered, as must legal and regulatory requirements.

This non-normative document summarizes general security issues and approaches as well as specific threats and countermeasures for the use of SAML assertions, protocols, bindings and profiles in a secure manner that maintains privacy. Normative requirements are specified in the normative SAML specifications.

9 References

The following are cited in the text of this document:

[Anonymity] Anonymity, Unobservability, and Pseudonymity – A Proposal for Terminology Andreas Pfitzmann, Marit Köhntopp, http://www.realname-diskussion.info/anon_terminology.pdf.

[FreeHaven] The Free Haven Project: Distributed Anonymous Storage Service Roger Dingledine & Michael J. Freedman & David Molnar http://www.freehaven.net/paper/node6.html http://www.freehaven.net/paper/node7.html

[IPsec] IETF IP Security Protocol Working Group, http://www.ietf.org/html.charters/ipsec-charter.html.

[LibBestPractices] C. Varney et al, Privacy and Security Best Practices, Version 2.0, November 12, 2003, http://www.projectliberty.org/specs/final_privacy_security_best_practices.pdf

[OCSP] "X.509 Internet Public Key Infrastructure – Online Certificate Status Protocol - OCSP," M. Myers, et al., IETF RFC 2560, June 1999, http://ietf.org/rfc/rfc2560.txt

[Pooling] Pooling Intellectual Capital: Thoughts on Anonymity, Pseudonymity, and Limited Liability in Cyberspace David G. Post http://www.cli.org/DPost/paper8.htm

[Rescorla-Sec] E. Rescorla et al., *Guidelines for Writing RFC Text on Security Considerations*, Best Current Practice RFC 3552, July 2003, http://www.ietf.org/rfc/rfc3552.txt?number=3552

[RFC2246] The TLS Protocol Version 1.0, http://www.ietf.org/rfcs/rfc2246.html.

[RFC2617] J. Franks et al, HTTP Authentication: Basic and Digest Access Authentication, RFC 2617, http://www.ietf.org/rfc/rfc2617.txt

[SAML11Bind] Bindings and Profiles for the OASIS Security Assertion Markup Language (SAML) V1.1, OASIS Standard, 2 September 2003 http://www.oasis-open.org/committees/download.php/3405/oasis-%20sstc-saml-bindings-1.1.pdf

[SAMLBind] S. Cantor et al., *Bindings for the OASIS Security Assertion Markup Language (SAML) V2.0*. OASIS SSTC, August 2004. Document ID sstc-saml-bindings-2.0-cd-01. See http://www.oasis-open.org/committees/security/.

[SAMLCore] S. Cantor et al., *Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0*. OASIS SSTC, August 2004. Document ID sstc-saml-core-2.0-cd-01. See http://www.oasis-open.org/committees/security/.

[SAMLGloss] J. Hodges et al., *Glossary for the OASIS Security Assertion Markup Language (SAML) V2.0*. OASIS SSTC, August 2004. Document ID sstc-saml-glossary-2.0-cd-01. See http://www.oasis-open.org/committees/security/.

[SAMLProf] S. Cantor et al., *Profiles for the OASIS Security Assertion Markup Language (SAML) V2.0*. OASIS SSTC, August 2004. Document ID sstc-saml-profiles-2.0-cd-01. See http://www.oasis-open.org/committees/security/.

[ShibMarlena] Marlena Erdos, Shibboleth Architecture DRAFT v1.1, http://shibboleth.internet2.edu/draft-internet2-shibboleth-arch-v05.html.

[SRMPPres] Message Queuing: Messaging Over The Internet Shai Kariv http://www.microsoft.com/israel/events/teched/presentations/EN308.zip

[SSL3] "The SSL Protocol Version 3.0", http://wp.netscape.com/eng/ssl3/draft302.txt

[WSS] Web Services Security specifications (WSS), OASIS. http://www.oasis-open.org/committees/wss.

[WSS-SAML] P. Hallam-Baker et al., *Web Services Security: SAML Token Profile*, OASIS, March 2003, http://www.oasis-open.org/committees/wss.

[XKMS] "XML Key Management Specifications (XKMS 2.0)", W3C Candidate Recommendation, 5 April 2004, http://www.w3.org/TR/xkms2/

[XMLEnc] Donald Eastlake et al., *XML Encryption Syntax and Processing*, http://www.w3.org/TR/xmlenc-core/, World Wide Web Consortium, December 2002.

[XMLSig] Donald Eastlake et al., *XML-Signature Syntax and Processing*, http://www.w3.org/TR/xmldsig-core/, World Wide Web Consortium.

The following additional documents are recommended reading:

[ebXML-MSS] Message Service Specification V2.0, OASIS, April 2002. http://www.oasis-open.org/committees/download.php/272/ebMS_v2_0.pdf. The information about the security module is the material of interest.

[ebXML-Risk] ebXML Technical Architecture Risk Assessment v1.0, http://www.ebxml.org/specs/secRISK.pdf.

[Prudent] Prudent Engineering Practice for Cryptographic Protocols, http://citeseer.nj.nec.com/abadi96prudent.html.

[Robustness] Robustness principles for public key protocols, http://citeseer.nj.nec.com/2927.html.

Appendix A. Acknowledgments

The editors would like to acknowledge the contributions of the OASIS Security Services Technical Committee, whose voting members at the time of publication were:

- Conor Cahill, AOL
- Hal Lockhart, BEA Systems
- Rick Randall, Booz Allen Hamilton
- Ronald Jacobson, Computer Associates
- Gavenraj Sodhi, Computer Associates
- Tim Alsop, CyberSafe Limited
- Paul Madsen, Entrust
- Carolina Canales-Valenzuela, Ericsson
- Dana Kaufman, Forum Systems
- Irving Reid, Hewlett-Packard
- Paula Austel, IBM
- Maryann Hondo, IBM
- Michael McIntosh, IBM
- Anthony Nadalin, IBM
- Nick Ragouzis, Individual
- Scott Cantor, Internet2
- Bob Morgan, Internet2
- Prateek Mishra, Netegrity
- Forest Yin, Netegrity
- Peter Davis, Neustar
- Frederick Hirsch, Nokia
- John Kemp, Nokia
- Senthil Sengodan, Nokia
- Scott Kiester, Novell
- Steve Anderson, OpenNetwork
- Ari Kermaier, Oracle
- Vamsi Motukuru, Oracle
- Darren Platt, Ping Identity
- Jim Lien, RSA Security
- John Linn, RSA Security
- Rob Philpott, RSA Security
- Dipak Chopra, SAP
- Jahan Moreh, Sigaba
- Bhavna Bhatnagar, Sun Microsystems
- Jeff Hodges, Sun Microsystems
- Eve Maler, Sun Microsystems
- Ronald Monzillo, Sun Microsystems
- Emily Xu, Sun Microsystems
- Mike Beach, Boeing

- Greg Whitehead, Trustgenix
- James Vanderbeek, Vodafone

The editors also would like to acknowledge the following people for their contributions to previous versions of the OASIS Security Assertions Markup Language Standard:

- Stephen Farrell, Baltimore Technologies
- David Orchard, BEA Systems
- Krishna Sankar, Cisco Systems
- Zahid Ahmed, CommerceOne
- Carlisle Adams, Entrust
- Tim Moses, Entrust
- Nigel Edwards, Hewlett-Packard
- Joe Pato, Hewlett-Packard
- Bob Blakley, IBM
- Marlena Erdos, IBM
- Marc Chanliau, Netegrity
- Chris McLaren, Netegrity
- Lynne Rosenthal, NIST
- Mark Skall, NIST
- Simon Godik, Overxeer
- Charles Norwood, SAIC
- Evan Prodromou, Securant
- Robert Griffin, RSA Security (former editor)
- Sai Allarvarpu, Sun Microsystems
- Chris Ferris, Sun Microsystems
- Emily Xu, Sun Microsystems
- Mike Myers, Traceroute Security
- Phillip Hallam-Baker, VeriSign (former editor)
- James Vanderbeek, Vodafone
- Mark O'Neill, Vordel
- Tony Palmer, Vordel

Finally, the editors wish to acknowledge the following people for their contributions of material used as input to the OASIS Security Assertions Markup Language specifications:

- Thomas Gross, IBM
- Birgit Pfitzmann, IBM

Appendix B. Notices