OASIS SSTC: SAML Security Considerations

draft-sstc-sec-consider-01

2001-11-14

Jeff Hodges
Chris McLaren
Prateek Mishra
Bob Morgan
Tim Moses
Evan Prodromou

Revision History

Revision	Date	Author	What
00	xx-Aug-2001	Jeff Hodges	Created.
01	2001-11-14	Chris McLaren	First substantive draft presented to TC

1 Introduction

This document describes and analyzes the security properties of the Security Assertions Markup Language. The intent is to provide architects, implementors, and reviewers of SAML-based systems information about..

- what threats, thus security risks, a SAML-based system is subject to,
- what security risks the SAML architecture addresses, and how it does so,
- those it does not address,
- recommendations on mitigating those risks

2 Background and Motivation

Communication between computer-based systems is subject to a variety of threats, and thus have associated risk, depending upon a host of factors including the nature of the communications, the nature of the communicating systems, the communication medium(s), the communication environment, the end-system environments, etc. See section 3 of [sec-cons-03] for an overview of threats inherent in the Internet (and intranets, by implication).

SAML is intended to aid deployers in establishing security contexts for application-level computer-based communications within and/or between security domains. This document comprises and in-depth analysis and assessment of the security afforded by SAML.

See section 2 of [sec-cons-03] for an overview of Communications Security and Systems Security. The former is directly applicable to the design of SAML. The latter is of interest mostly in the context of SAML's threat models. It is worthwhile to note that SAML itself is intended to address the "endpoint authentication" (in part, at least) aspect of Communications Security, and also the "unauthorized usage" aspect of Systems Security.

3 Overview

This document attempts to outline what threats and risks were considered during the design of SAML, and what counter-measures are available to attenuate those risks, in so far as it is possible to do so. This document should also provide guidance for implementers and deployers with regards to "best practices" for security decisions in the SAML context.

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 beyond the scope of SAML:

- initial authentication: SAML allows statements to be made about authentications that have occurred, but includes no requirements or specifications for this these authentications. Consumers of authentication assertions should be wary of blindly trusting these assertions unless/until they know the basis on which they were made. Confidence in the assertions can never exceed the confidence that the asserting party has correctly arrived at the conclusions asserted.
- PKI issues: In many cases the security of a SAML conversation will depend on the underlying PKI. For example, SOAP messages secured via XML-DSIG signatures are only secured in so far 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 set-up is not trivial, but must be done correctly in order for layers built on top of it (such as parts of SAML) to be secure.

3.1 SAML Threat Model

The general Internet threat model described in section 3 of [sec-cons-03] is the basis for the SAML Threat model. Our general assumptions are that the various endpoints of a SAML transaction (and there may be more than two) are uncompromised, but that the attacker has complete control over the communications channel.

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

In all cases the local mechanisms that systems will use to decide whether or not to generate assertions is an out-of-scope step. This means that 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 factually incorrect assertion then the threats arising from the consumption of that assertion by downstream systems are explicitly out-of-scope.

The direct consequence of this is that the security of a system that uses assertions as inputs is only as good as the security of the system used to generate those assertions. When determining what assertion 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 factually incorrect but validly issued assertions is a large one. Trust policies for assertion consumers should never be written without significant consideration of the extent to which issuers of assertions that a system will consume can actually be trusted to make those assertions correctly.

4 Security Techniques

4.1 Authentication

Authentication 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 the SAML Message. This authentication MAY be either in one direction—from the session initiator to the receiver—or bi-directional. The specific method will be determined by the communications protocol used. For instance, the use of a secure network protocol, such as [RFC2246] or [IPSEC] provides ability for the sender of an SAML Message to authenticate the destination for the TCP/IP environment.

4.1.2 Message-Level

XML Digital Signature provides a method of creating a persistent “authentication” that is tightly coupled to a document. This 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 recipient(s) and not anyone else who encounters the message while it is in transit.

4.2.1 In Transit

Use of a secure network protocol such as [RFC2246] or [IPSEC] provides transient confidentiality of a message as it is transferred between two nodes.

4.2.2 Message-Level

XML Encryption is a W3C/IETF joint activity that is actively engaged in the drafting of a specification for the selective encryption of an XML document(s). It is anticipated that this specification will be completed within the next year. This has been identified as a viable means of providing persistent, selective confidentiality of elements within an XML Message.

Until such time as XML Encryption is an accepted standard confidentiality may be implemented in transit (and not end-to-end) by reliance on transports that provide in transit confidentiality (as described in 4.2.1 above).

4.3 Data Integrity

Data integrity is provided by a system when there is a method of confirming 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 [RFC2246] or [IPSEC] MAY be configured so as to provide for integrity check CRCs of the packets transmitted via the network connection.

4.3.2 Message-Level

XML Digital Signature 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 TLS/SSL Cipher Suites

The use of TLS/SSL over HTTP is recommended at many places in this document. However TLS/SSL can be configured to use many different cipher suites, not all of which are adequate to provide “best practices” security. A brief description of what exactly constitutes a “cipher suite” follows, and is in turn followed by recommendations for cipher suite selection.

4.4.1 What Is A Cipher Suite

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:

1. What kind of key exchange algorithm is 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 is currently the most interoperable one. Another important key exchange algorithm is the authenticated Diffie-Hellman “DHE_DSS” key exchange, which has no patent-related implementation constraints.

2. Whether it is freely exportable from the U.S. due to using short (512 bits) public keys for key exchange and short symmetric keys (40 bits) for encryption. Those are currently subject to breaking in an afternoon by a moderately well equipped adversary.

3. What encryption algorithm is used. The fastest option is the RC4 stream cipher; DES and variants (DES40, 3DES-EDE) are also supported in "cipher block chaining" (CBC) mode, as

is (in some suites) null encryption. *(Null encryption does nothing; in such cases SSL is used only to authenticate and provide integrity protection. Cipher suites with null encryption do not provide confidentiality, and should not be used in cases where it is a requirement.)*

4. What digest algorithm is used for the Message Authentication Code, either MD5 or SHA1.

So for example the cipher suite named SSL_DHE_DSS_EXPORT_WITH_DES40_CBC_SHA uses SSL, 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 can't change the cipher suites which are available.

4.4.2 Recommendations regarding cipher suites

The following cipher suites adequately meet the requirements for confidentiality and message integrity, and can be configured to meet the authentication requirement as well (by forcing the presence of X.509V3 certificates). They are also well supported in many client applications. Support of these suites is recommended:

- TLS_RSA_WITH_3DES_EDE_CBC_SHA (when using TLS)
- SSL_RSA_WITH_3DES_EDE_CBC_SHA (when using SSL)

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 to also implement support for the AES cipher suites, such as:

- TLS_RSA_WITH_AES_128_CBC_SHA

5 Analyses of SAML Specifics

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 via one of the bindings.

However, there is one issue at the assertion level that bears analysis: **An assertion, once issued, is out of the control of the issuer.**

This 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 and the issuer has no control over with whom the consumer will share the information contained in the assertion (or the assertion itself). This isn’t even mentioning our malicious attacker who can see the contents of each assertion that passes 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 consider the possible consequences if the information in the assertion is stored on a remote site (where it can be directly mis-used, or exposed to potential hackers, or possibly stored for more creatively fraudulent uses). Consider also 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 threats considered in the design of the SAML request/response protocol, the risks arising from these threats, and the appropriate counter-measures, depend to a large extent on the particular protocol-binding that is used. The bindings described in [ref-bindings] are each considered separately below.

5.2.1 Denial of Service Attack

The SAML Protocol itself opens the door to one specific threat—the denial of service attack. Since handling a SAML Request is potentially a very expensive operation (parse of the request message—typically a DOM construction, database/assertion store lookup—potentially on an unindexed key, construction of a response message, and potentially one or more digital signature operations) there is a particularly high asymmetry between the effort required by an attacker generating requests and the effort needed to handle those requests.

Counter-measures against this attack are various and are each considered separately below.

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 CA at their root) will provide traceability in the case of a denial-of-service 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 denial-of-service 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 can easily be spoofed.)

5.2.1.2 Requiring signed requests

In addition to the benefits gained from client authentication (per 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 huge amount of work for the responder. Narrowing this asymmetry decreases the risk associated with this attack.

Note however that an attacker can theoretically capture a signed message and then replay it continually, getting around this requirement. This may be avoided by requiring the use of the <SignatureProperties> element containing a timestamp which 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 high security you have against replay DOS attacks. Sadly the use of <SignatureProperties> to define a timestamp is not part of the XML-DSIG specification and could lead to interoperability issues.

5.2.1.3 Restricting access to the interaction URL

If the ability to issue a request to the SAML processor is limited at a very low level to a set of known parties, this drastically reduces the risk of a denial of service attack. In this case only attacks originating from within the finite set of known parties are possible; this both greatly decreases exposure to potentially malicious client and greatly decreases exposure to DDOS attacks using compromised machines as zombies.

5.3 SAML Protocol Bindings

5.3.1 SOAP 1.1

Since this 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 below. Note that particular instantiations of this binding (such as the SOAP over HTTP case) may have additional requirements, and must be considered separately.

5.3.1.1 Eavesdropping

Since there is no in transit confidentiality requirement it is entire possible that an eavesdropping party could acquire both the SOAP message containing the request and the SOAP message containing the response.

This 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 be 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 site X.

Similarly, eavesdropping on a series of authorization queries could create a “map” of resources which 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 an query and response may expose that a given user is active on the querying site, which could easily be information that should not be divulged in cases such as medicial information sites, political sites, etc. Also the details of the assertion(s) carried in the response may be information that should be kept confidential. This is particularly true for the attribute case where the response typically carries information about attributes of the subject; if these attributes represent information that should not be available to entities not party to the transaction(financial information like credit ratings, medical attributes, etc.) then the risk from eavesdropping is high.

In cases where any of these risks is a concern the counter-measure for eavesdropping attacks is, naturally, to provide some form of in transit message confidentiality. For SOAP messages this confidentiality can be enforced 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 should be the XML Encryption standard [reference] when it is finalized. This standard should allow encryption of the SOAP message itself, which eliminates the risk of eavesdropping unless the key used in the encryption has been compromised (reference for this?)

Until such time as the XML Encryption standard becomes available deployers will need to depend on the SOAP transport layer, or a layer beneath it, to provide in transit confidentiality.

The details of how to do this depend on the specific SOAP transport chosen. Using HTTP over TLS/SSL is one example of a method of providing in transit confidentiality (and is considered in detail in section 5.3.1.7). Other transports will necessitate other in transit confidentiality techniques (for example an SMTP transport might use S/MIME).

Additionally, it is possible that a layer beneath the SOAP transport might, in some cases, provide the in transit confidentiality required. 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.

5.3.1.2 Replay

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.

In general the best way to prevent replay is prevent the message capture in the first place. Some of the transport level schemes used to provide in transit confidentiality will accomplish this. 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 requestor and encrypted to the responder, then the attacker can replay that request at any time without needing to be able to undo the encryption. This is a particular issue since the SAML Request does not include information about the issue time of the request, thus making it difficult to determine if replay is occuring.

In general the only recourse is to design systems that use the unique key of the request (its ID) 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.

Fixed-use tokens & ticketing model.

5.3.1.3 Message Insertion

The message insertion attack for the SOAP binding amounts to the creation of a request (for information on replacing all or part of a response see 5.3.1.5 and 5.3.1.6 below). The ability to make a request is not a threat at the SOAP binding level.

5.3.1.4 Message Deletion

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

In either case the SAML protocol binding for SOAP does not address this threat. The SOAP protocol itself, and the transports beneath it, may provide some information depending on how the message deletion is accomplished.

Reliable RPC DCE UDP Variant Secure Mode

5.3.1.5 Message Modification

Message modification is a threat to this 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 an attribute query’s <CompletenessSpecifier> could produce results leading to compromise or denial of service, as could altering the <AttributeDesignator>s themselves.

Modification of the request to alter 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 the result of an authorization decision could lead to very serious security breaches, In order to address these potential threats a system must be introduced to provide a guarantee of in transit message integrity. The SAML Protocol, and the SOAP binding, neither requires nor forbids the deployment of systems that guarantee in transit message integrity, but due to this large threat it is HIGHLY RECOMMENDED that such a system is

At the SOAP binding level this can be accomplished by digitally signing requests and responses. (CORE Allows Reference). If messages are digitally signed (with a sensible PKI setup reference), then the recipient has a guarantee that the message has not be 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.

5.3.1.6 Man-In-The-Middle.

The SOAP binding is susceptible to man-in-the-middle 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) some sort of bilateral authentication is required.

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 could also be accomplished by digitally signing both requests and responses (with all the caveats discussed in section 5.3.1.5 above). This doesn't 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 depend on asynchronous messaging (i.e. no 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 author are the same party in order to prevent this weaker form of "man-in-the-middle 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 man-in-the-middle attacks. The shorter the valid window of the assertion, the less damage can be done if it is intercepted

5.3.1.7 Specifics of SOAP over HTTP

Since the SOAP over HTTP sub-binding requires that conformant applications support HTTP over TLS/SSL with bilateral certificate-backed authentication this system is 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 full certificate support is mandated. If an acceptable level of protection from the various risks can be arrived at through other means (for example, via 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.

Note however that the use of transport level security (such as the SSL or TLS protocols on top of HTTP) only provides confidentiality/integrity/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.

5.4 Profiles of SAML

In order to use SAML security assertions in practice, they are embedded in or combined with other objects by an originating party. These combined objects are then communicated from the originating site to a destination, and subsequently processed at the destination. A set of rules describing how to embed and extract SAML assertions into a framework or protocol is termed a **profile** for SAML. A set of rules for embedding and extracting SAML assertions into a specific class of <FOO> objects is termed a <FOO> profile of SAML. This specification defines two different profiles for SAML, each of which have two different "sub-cases". The profiles defined are: Web Browser Single Sign-on (with Artefact and Form Post sub-cases) and SOAP (with HolderOfKey and SenderVouches sub-cases). Each profile is considered from a security perspective below.

5.4.1 Web Browser Single Sign-On (General concerns)

User authentication at the source site is still explicitly out of scope, as are all issues that arise from it. The key notion is that the source system entity MUST be able to ascertain that it is the same authenticated client system entity that it is interacting with 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).

5.4.1.1 Eavesdropping

In all web-browser cases the possibility of eavesdropping exists. 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. SSL/TLS over HTTP ([RFC2246]) meets this requirements, as does [IPSEC].

5.4.1.1.1 Eavesdropping: Theft of the user authentication information

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

In order to avoid this issue 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, trusted, source site, prior to revealing the authentication information. Using HTTP over TLS can be used to address this concern.

5.4.1.1.2 Eavesdropping: Theft of the bearer token

In the case where the authentication assertion contains the assertion bearer authentication protocol identifier, theft of the artefact will enable an adversary to impersonate the subject.

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 AudienceRestrictionConditionType 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, trusted, source site.

5.4.1.2 Replay

The possibility of a replay attack, used either to attempt to deny service or to retrieve information fraudulently, exists for this profile. The specific counter-measures used depend on the sub-case and are discussed below.

5.4.1.3 Message Insertion

Message Insertion attacks are not a threat to this profile.

5.4.1.4 Message Deletion

Deleting a message during any step of the interactions between the browser, SAML producer, and SAML consumer will cause the interaction to fail.

In each case this results in a denial of some service, but does not increase the exposure of any information.

The SAML specification provides no counter-measures for message deletion.

5.4.1.5 Message Modification

The possibility of alteration of the messages in the stream exists for the Web Browser Single Sign-on case. Some potential undesirable results:

- Alteration of the initial request can result in rejection at the SAML Issuer, or creation of an artefact targeted at a different resource than the one requested

- Alteration of the artefact 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)

- Etc.

In order to avoid the possibility of these problems, traffic needs to occur via a system that guarantees message integrity from endpoint to endpoint.

For the Web Browser Single Sign-on profile the recommended method of providing message integrity in transit is the use of TLS/SSL over HTTP with a cipher suite that provides data integrity checking.

5.4.1.6 Man-In-The-Middle.

Man-In-The-Middle attacks are particularly pernicious for this profile. The MITM can relay requests, capture the returned assertion (or artefact) and relay back a false one. Then the original user can't access the resource in question, but the MITM can using the captured resource.

Preventing this requires a number of counter-measures to be in place. Firstly, 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 bi-directional 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 requestor). To prevent the eavesdropping a confidentiality system should be put in place. To prevent alteration of the message during port forwarding, a data integrity system should by put in place.

For this profile 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.

5.4.2 SAML Artefact

The specific threats and counter-measures for the SAML Artefact profile are outlined below.

5.4.2.1 Replay

The threat of replay as a re-use of an artefact has been addressed by the requirement that each artefact is a one-time use item. Systems should track cases where multiple requests are made referencing the same artefact as this may represent intrusion attempts.

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

5.4.2.2 Threats Specific to this profile

This section should included detailed discussion of the threats outlined in the bindings docs…

4.1.3.3.1 Stolen artifact

Threat:

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

Counter-Measure:

As indicated in Steps 1, 2, 5 and 6, confidentiality must be provided whenever an artifact is communicated between a site and the user's browser. This provides protection against an Eve gaining access to a real user's SAML artifact.

Should Eve defeat the measures used to ensure confidentiality, additional counter-measures are available. Recall that SAML assertions communicated through Step 5 must always include an SSO assertion. SSO assertions SHOULD have short validity periods (values for NotBefore and NotOnOrAfter attributes) consistent with successful functioning of the profile. This ensures that a stolen artifact can only be used successfully within a small time window.

Source and destination sites SHOULD make some reasonable effort to ensure that clock settings are 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.

RECOMMENDATIONS for the Source Site:

(a) Source sites SHOULD track the time difference between when a SAML artifact is generated and placed on a URL line and when the destination site "calls back" for an assertion. A maximum time limit of a few minutes is recommended. Should an assertion be requested by a destination site query beyond this time limit, a SAML error should be returned by the source site.

(b) SSO assertions MAY BE created by the source site either when the corresponding SAML artifact is created or when the destination site “calls back” for an assertion. In each of these cases, the validity period of the assertion should be set appropriately (longer in the former case, shorter for the latter).

(c) values for NotBefore and NotOnOrAfter attributes of SSO assertions SHOULD have the shortest possible validity period consistent with successfully communication of the assertion from source to destination site. This is typically on the order of a few minutes.

RECOMMENDATIONS for Destination Site:

(a) 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 for example, requiring the IssueInstant attribute value or AuthenticationInstant attribute value of the assertion to be within a few minutes of the time at which the assertion is received at the destination site.

(b) Authentication statements MAY include an <AuthenticationLocality> 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.

4.1.3.3.2 Attacks on Steps 4 and 5

Threat: The message exchange on steps 4 and 5 may be attacked in a variety of ways, including: artifact or assertion theft, replay, message insertion or modification, MITM (man-in-the-middle attack).

Counter-Measure: The requirement for the use of a SAML protocol binding with the properties of bilateral authentication, message integrity and confidentiality obviates these attacks.

4.1.3.3.3 Malicious Destination Site

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

Counter-Measure:

The new destination site will need to authenticate itself to the source site so as to obtain the SAML assertions corresponding to the SAML artifacts. There are two cases:

(a) If the new destination site has no relationship with the source site, it will be unable to authenticate and this step will fail.

(b) If the new destination site has an existing relationship with the source site, the source site will determine that artifacts are being queried against from a site other than the one to which the artifacts were issued. In such a case, the source site will not provide the assertions to the new destination site.

4.1.3.3.4 Forged SAML artifact

Threat: A MAL (malicious user) could forge a SAML artifact.

Counter-Measure:

A SAML artifact must be constructed in such a way that it is very hard to guess and Section 4.1.3 provides specific recommendations in this space. A MAL could attempt to repeatedly “guess” a valid SAML artifact value (one that corresponds to an existing assertion at a source site) but given the size of the value space would likely require a very large number of failed attempts. A source site SHOULD implement measures to ensure that repeated attempts at querying against non-existent artifacts are monitored.

4.1.3.3.5 Browser State Exposure

Threat: The SAML artifact profile involves “upload” of SAML artifacts to the web browser from a source 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 “re-used” at some later point in time.

Counter-Measure: The “one-use” property of SAML artifacts ensures that they may not be re-used from a browser. Due to the recommended short life-times of artifacts and mandatory SSO assertions, it is difficult to steal an artifact and re-use it from some other browser at a later time.

5.4.3 Form POST

The specific threats and counter-measures for the Form POST profile are outlined below.

5.4.3.1 Replay

Replay attacks amounts to resubmission of the form in order to access a protected resource fraudulently. The required one-time use property of the assertions transferred (mandated by the profile) prevents this from succeeding.

5.4.3.2 Threats Specific to this profile

This section should included detailed discussion of the threats outlined in the bindings docs…

4.1.4.2.1 Stolen assertion

Threat: If an eavesdropper (Eve) can copy the real user’s SAML assertion (Form POST), then the Eve could construct an appropriate POST body and be able to impersonate the user at the destination site.

Counter-Measure: As indicated in Steps 1, 2, 3 and 4, confidentiality must be provided whenever an assertion is communicated between a site and the user’s browser. This provides protection against an Eve gaining access to a user’s SAML assertion.

Should Eve defeat the measures used to ensure confidentiality, additional counter-measures are available. Recall, that SAML assertions communicated through Step 3 must always include an SSO assertion. SSO assertions SHOULD have short validity periods (values for NotBefore and NotOnOrAfter attributes) consistent with successful functioning of the profile. This ensures that a stolen assertion can only be used successfully within a small time window.

Source and destination sites SHOULD make some reasonable effort to ensure that clock settings are 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.

RECOMMENDATIONS for the Source Site:

(a) values for NotBefore and NotOnOrAfter attributes of SSO assertions SHOULD have the shortest possible validity period consistent with successfully communicating the assertion from source to destination site. This is typically of the order of a few minutes.

RECOMMENDATIONS for Destination Site:

(a) 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 for example, requiring the IssueInstant attribute value or AuthenticationInstant attribute value of the assertion to be within a few minutes of the time at which the assertion is received at the destination site.

(b) Authentication statements MAY include an <AuthenticationLocality> 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.

4.1.4.2.2 MITM Attack

Threat: Since the destination site obtains bearer SAML assertions from the user via a Form post, a malicious site could impersonate the user at some new destination site. The new destination site would believe the malicious site to be the user.

Counter-Measure:

The destination site MUST check the <saml:Target> elements of the SSO assertion to ensure that at least one of their values matches the <assertion consumer host name and path>. As the assertion is digitally signed, the <saml:Target> value cannot be altered by the malicious site.

4.1.4.2.3 Forged Assertion

Threat: A MAL or the browser user could forge or alter a SAML assertion (form POST).

Counter-Measure: The POST browser profile requires SAML assertions to be signed, thus providing both message integrity and authentication. The destination site MUST verify the signature and authenticate the issuer.

4.1.4.2.4 Browser State Exposure

Threat: The POST browser profile involve upload of assertions to the web browser from a source 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 “re-used” at some later point in time.

Counter-Measure: Assertions communicated using FORM post must always include a SSO assertion. It is recommended that SSO assertions have short life-times and that destination sites must ensure that they may be used only once.

5.4.4 SOAP Profile

This profile defines methods for securely attaching security assertions to a SOAP document. SOAP documents are used in multiple contexts specifically including cases where the message is transported asynchronously (i.e. no session is active, message can be persisted) and is routed through a number of intermediaries. This introduces additional issues and possible threats that are not possible in cases based on a current session

[Reference 4.2.4 in bindings]

5.4.4.1 Holder of Key

General information on the security model of this profile

5.4.4.1.1 Eavesdropping

Eavesdropping continues to be a threat in the same manner outlined in section 5.3.1.1. The routable nature of SOAP adds the potential for a much greater number of steps and actors in the course of a message's lifetime, which means all the potentials for eavesdropping are increased as the number of possible times a message is in transit increases.

In addition the persistent nature of the SOAP messages add an additional possibility of eavesdropping: items that are stored can be read from their store.

To provide maximum protection from eavesdropping assertions should be encrypted such that only the intended audiences can view the material. This removes threats of eavesdropping in transit, but does not remove risks associated with storage by the receiver, or poor handling of the clear text by the receiver.

5.4.4.1.2 Replay

Binding of assertions to a document opens the door broadly for replay attacks by a malicious user. Issuing a "HolderOfKey" assertion amounts to "blessing the user's key" for the purpose of binding assertions to documents. Once a HolderOfKey assertion has been issued to a user, that user can bind it to any document or documents he chooses.

While each assertion is signed, and bound by a second signature into a document, there is nothing preventing a malicious user from detaching a (signed) assertion from the document it arrived in and rebinding it to another document.

There are two lines of defence against this type of attack. The first, obvious, one is to carefully consider to whom you issue HolderOfKey assertions (can they be trusted with the right to attach the assertion to any document?) and what kind of assertions you issue as HolderOfKey assertions (do you want to give up control over the binding of this particular statement to a given document?). The second is a short lifetime on the assertion, to narrow the window of opportunity for this attack.

Also the capture and resubmission of the total message is a potential issue, but one that is beyond the scope of the SAML specification.

5.4.4.1.3 Message Insertion

There is no message insertion attack at the level of the HolderOfKey profile.

5.4.4.1.4 Message Deletion

There is no message deletion attack at the level of the HolderOfKey profile.

5.4.4.1.5 Message Modification

The double signing of this profile prevents most message modification attacks. The receiver is always able to verify the signature on the assertion itself (and should be able to verify that the key used in that signing act is associated with the putative signer, via X509V3 certificate and CRL checks, etc.) which provides a guarantee that the assertion is unaltered.

The receiver can also verify the binding signature to ensure that the message to which the assertion is attached is unaltered.

The profile is secure against modification within the limits of the PKI setup in place. The remaining threats are outside the scope of SAML (compromised keys, revoked certificates being used, etc.)

Note that the threat of message modification by the holder of the key exists as discussed in the Replay section above.

5.4.4.1.6 *Man-In-The-Middle.*

MITM is impossible for this profile, since the assertion specifies the key that must be used for the binding signature, and the assertion itself is protected against tampering by a signature.

The MITM can eavesdrop (if communication is not protected by some confidentiality scheme) but cannot alter the document without detection.

Does DSIG prevent me from altering the signer info? Can I remove the key from the signature element (possibly forcing XKMS lookup or other binding that I can pervert to my malicious ends?)

5.4.4.2 Sender Vouches

5.4.4.2.1 *Eavesdropping*

Eavesdropping continues to be a threat in the same manner outlined in section 5.3.1.1. The routable nature of SOAP adds the potential for a much greater number of steps and actors in the course of a message's lifetime, which means all the potentials for eavesdropping are increased as the number of possible times a message is in transit increases.

In addition the persistent nature of the SOAP messages add an additional possibility of eavesdropping: items that are stored can be read from their store.

To provide maximum protection from eavesdropping assertions should be encrypted such that only the intended audiences can view the material. This removes threats of eavesdropping in transit, but does not remove risks associated with storage by the receiver, or poor handling of the clear text by the receiver.

5.4.4.2.2 *Replay*

The fact that the sender does all binding prevents a variety of replay attacks that relate to reusing the assertion with different documents. In this case the assertions are directly signed into the document so separating them from the document for reuse would not benefit a malicious user.

However, the capture and resubmission of the total message is still a potential issue, albeit one that is beyond the scope of the SAML specification.

5.4.4.2.3 *Message Insertion*

There is no message insertion attack at the level of the SenderVouches profile.

5.4.4.2.4 *Message Deletion*

There is no message insertion attack at the level of the SenderVouches profile.

5.4.4.2.5 *Message Modification*

The binding signature should prevent any message modification attacks. Selection of what parts of the document to sign should be made carefully with the possibility of this attack in mind.

Receivers should consider only the portions of the document actually bound by signature to the assertions as valid with respect to the assertions.

5.4.4.2.6 Man-In-The-Middle.

The requirement for a signature here should also prevent MITM attacks. Note that the verifiability of the signature is key to this step: not only must a receiver be able to verify that a document was signed with a key, he needs to be able to verify the binding of key to identity. Typically this is accomplished by including an X509V3 certificate with the digital signature which the receiver verifies with respect to some set of trusted Certifying Authorities.

If this step is skipped then MITM becomes a possibility where the MITM captures the original document, alters it, and passes along this new document signed with a key that purports to be from the original sender (but which is actually held by the MITM).

The MITM can eavesdrop (if communication is not protected by some confidentiality scheme) but cannot alter the document without detection.

6 References

6.1 SAML Specification Documents

[core] Security Assertions Markup Language: Core Assertion Architecture
http://www.oasis-open.org/committees/security/docs/…WHATEVER

[bindings] Oasis Security Services Bindings Model
http://www.oasis-open.org/committees/security/docs/… WHATEVER

[saml-reqs-01] Oasis Security Services Use Cases And Requirements
http://www.oasis-open.org/committees/security/docs/draft-sstc-saml-reqs-01.pdf

6.2 Normative References

[xml-dsig] XML-Signature Syntax and Processing, W3C Candidate Recommendation 19-April-2001.
http://www.w3.org/TR/2001/CR-xmldsig-core-20010419/

[saml-glossary] OASIS Security Services TC Glossary
http://www.oasis-open.org/committees/security/docs/draft-sstc-glossary-01.pdf

6.3 Supplementary Documents

[xml-encyption] XML Encryption Syntax and Processing, WG Working Draft 26 June 2001.
http://www.w3.org/TR/xmlenc-core/

[sec-cons-03] Guidelines for Writing RFC Text on Security Considerations
http://www.ietf.org/internet-drafts/draft-rescorla-sec-cons-03.txt

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

[ebXML-MSS] Message Service Specification: ebXML Transport, Routing & Packaging Version 1.0 (chapter 12 specifically)
http://www.ebxml.org/specs/ebMS.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