SAML: Subject Confirmation Methods and Trust Models

Originally published as: SSTC/SAML FTF #4 Workitems
Available at: http://www.oasis-open.org/committees/security/minutes/SSTC-F2F-4-Blakley-Workitems.ppt

By: Bob Blakley, 27-29 Aug 2001

Conversion to document format by Jeff Hodges, 23-Apr-2003

Table of Contents

Figures

Section 1: SAML "SenderVouches" SubjectConfirmation Method:

A Proposed Alternative to Approaches in Bindings 0.5 Draft

Proposals in the Bindings 0.5 Draft:

Two proposals:

–1) Sender public key in assertion about subject

•Later revised: sender = subject

•Therefore "subject public key in assertion about subject"

–2) Hash of message in assertion about subject

•This case isn't discussed further here

•It requires issuance of a new assertion for every message

•I believe this isn't feasible

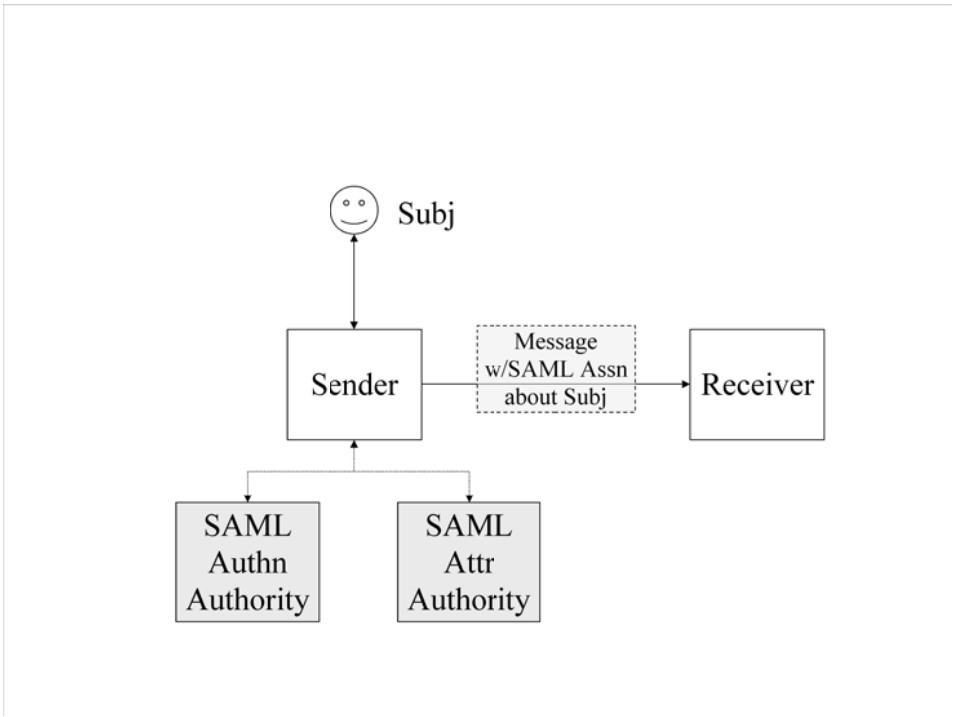

Figure 1: Bindings 0.5/1 (Original): Context

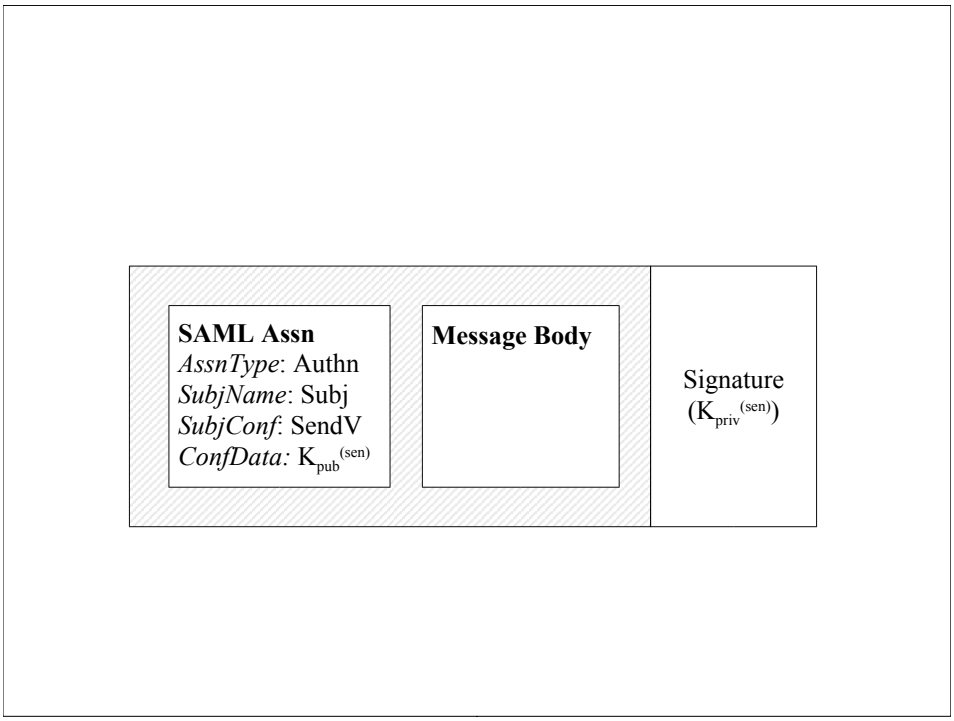

Figure 2: Bindings 0.5/1 (Original): Message Format

Bindings 0.5/1 (Original): MITM Attack Analysis

•Assumption: Sender ≠ Subject

–Subject public key CANNOT be used by Receiver to confirm subject (Sender does not have access to Subject private key; thus cannot apply signature to message)

•Receiver trusts Sender to assert the correct subject

–Another way of saying this: Sender can assert incorrect subjects

–No other party can forge messages "from" Sender

•because Receiver validates Attr. Authority's assertion of Sender's public key, and Receiver validates Sender's assertion of binding of Authn Assertion to message body.

•Net: No intermediary between Sender and Receiver can execute MITM cut-and-paste attack.

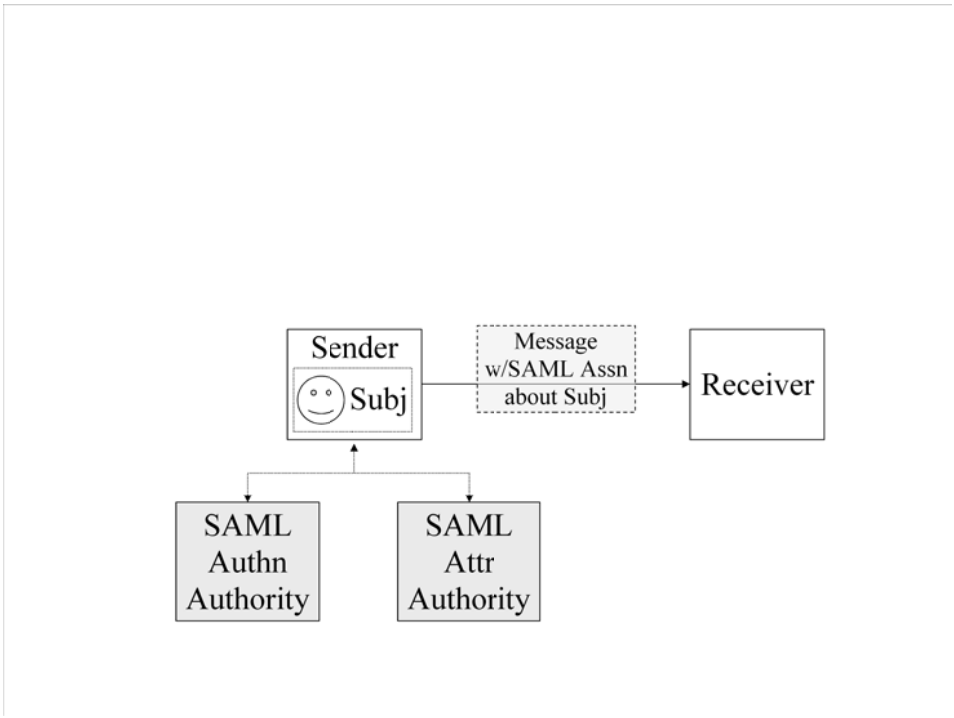

Figure 3: Bindings 0.5/1 (Revised): Context

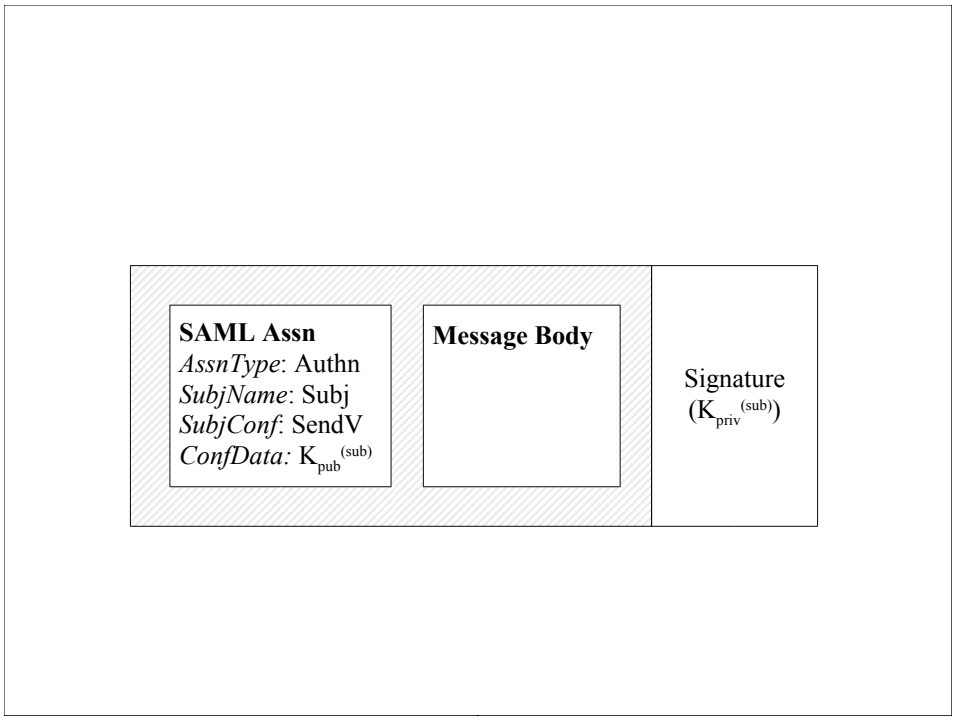

Figure 4: Bindings 0.5/1 (Revised): Message Format

Bindings 0.5/1 (Revised): MITM Attack Analysis

•Sender = Subject, so presumably subject does not attempt to forge own messages.

•No other party can forge subject's signature.

•However, correspondence of subject identity from Authn Assn and public key in SubjectConfirmation/ConfData element is NOT checked by Receiver.

–Therefore, Receiver trusts Authn Authority to never generate an assertion with SubjConf = SenV and ConfData ≠ public-key(assertion subject)

Bindings 0.5/1 (Revised) = HOK

•Since sender = subject, the key in ConfData is always the subject's public key.

•The same effect is achieved by setting SubjConf = HOK and ConfData = $K_{pub}^{(sub)}$!

•So there's no need for another SubjectConfirmation method if this is all we want to do.

Alternate Proposal: SenderVouches

SenderVouches: Goals

•Permit sender to vouch that Authn assertion sent with message designates the correct subject

•Enable use of a single subject assertion in multiple messages from the same sender

•Keep information relating to sender out of assertions about subject

•Keep information relating to the message out of assertions about subject (i.e. message digest in assertion)

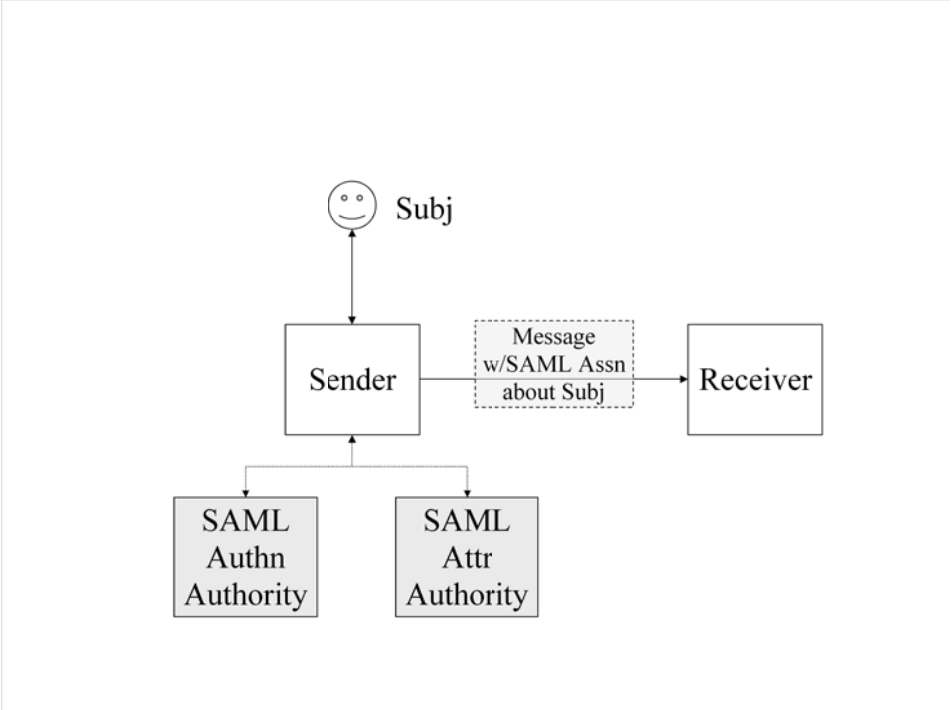

Figure 5: SenderVouches: Context

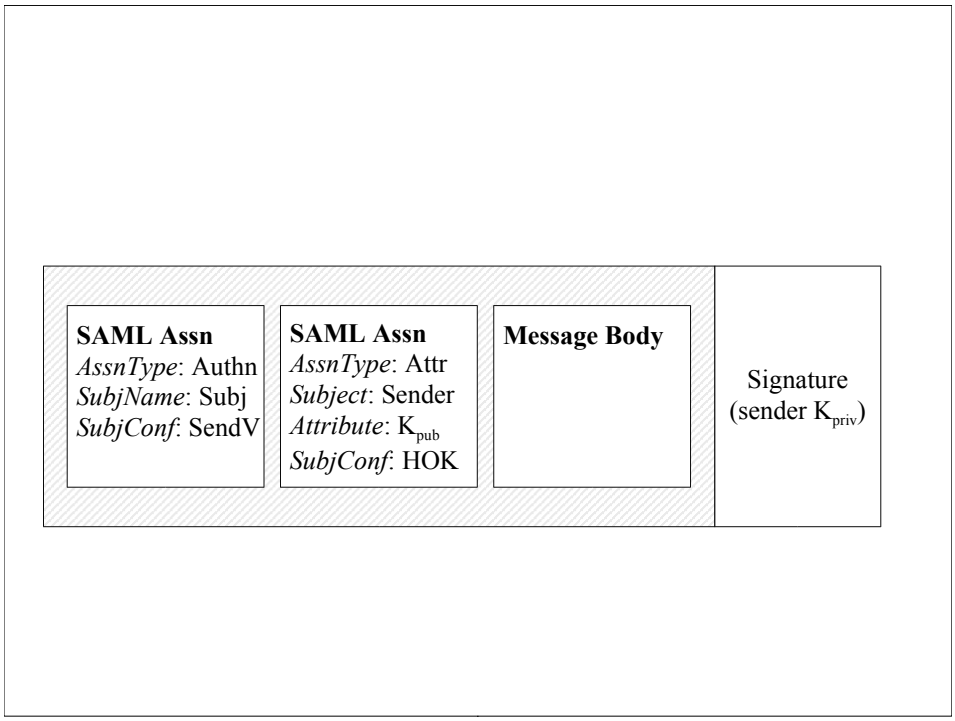

Figure 6: SenderVouches: Message Format

SenderVouches: Processing (1)

•Extract Authn assertion

•Extract Authn assertion's SubjConf method

–It is "SenderVouches"

–"SenderVouches" method requres Receiver to look for Sender's claim that the Authn assertion designates the correct subject

–In the store-and-forward case, this claim consists of the signature which Sender applied to the entire message (which incorporates both assertions and the message body)

SenderVouches: Processing (2)

•Validate SubjectConf of Authn Assertion

–Retrieve Sender public key from Attr Assertion

–Extract SubjConf from Attr Assertion (= "HOK")

–use Sender public key to verify signature on message

•This validates the fact that "Sender" is the subject of the Attr assertion, because the signature proves the sender's possession of the private key corresponding to the public key asserted by the Attr Assertion.

•It also binds the Authn and Attr assertions to the message body

•… which in turn validates the fact that "Subj" is the subject of the Authn assertion, because Sender's signature represents Sender's claim that the Authn assertion designates the correct subject

SenderVouches: Trust Model

•Receiver trusts Sender to vouch that the Authentication Assertion designates the correct subject

•Receiver trusts SAML Attr Authority to vouch for the Sender's public key

•Sender trusts SAML Authn Authority to vouch for the Subject's name

•Open question: does Receiver need to trust SAML Authn Authority to vouch for the Subject's name? Or can it simply rely transitively on the Sender's trust in this?

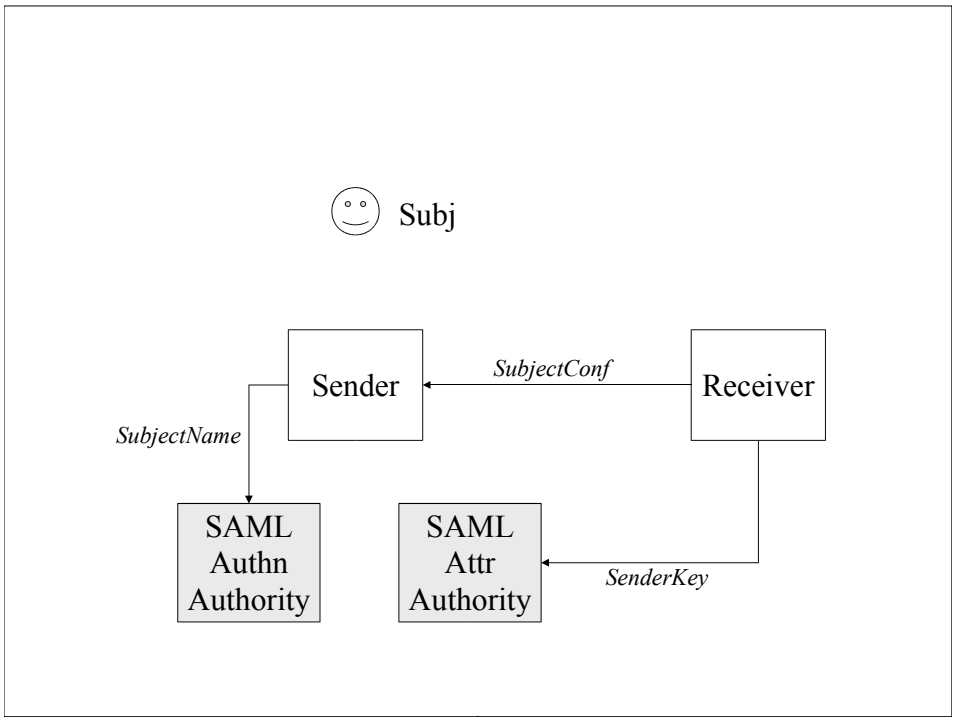

Figure 7: SenderVouches: Trust Relationships

SenderVouches: MITM Attack Analysis

•Assumption: Sender ≠ Subject

–Subject public key CANNOT be used by Receiver to confirm subject (Sender does not have access to Subject private key; thus cannot apply signature to message)

•Receiver trusts Sender to assert the correct subject

–Another way of saying this: Sender can assert incorrect subjects

–No other party can forge messages "from" Sender

•because Receiver validates Attr. Authority's assertion of Sender's public key, and Receiver validates Sender's assertion of binding of Authn Assertion to message body.

•Net: No intermediary between Sender and Receiver can execute MITM cut-and-paste attack.

•Exactly the same as Bindings 0.5 (Original)

Advantages Over Other Proposals

•Supports SubjConf via Sender vouching

•No requirement for information about message in any assertion

•No requirement for information about Sender in Subj Authn assertion

•Thus both assertions can be re-used independently

•Single signature verification validates SubjConf for both assertions

Use in Session-Oriented Environments

•SubjConf = "SenderVouches" should also be usable in session-oriented environments

–Could be done analogously to store-and-forward: simply send a message within the session which consists of the two assertions and the original message body

–Could be done by depending on the session security infrastructure (Receiver verifies Sender identity at session setup time; any message containing Authn assertion within session context is treated as if Sender implicitly asserted that Authn assertion designates the correct subject

Section 1 Conclusions

•To accomplish goals of Bindings 0.5/1 (Original) and Bindings 0.5/2: ***use SenderVouches***

•To accomplish goals of Bindings 0.5/1 (Revised): ***use HolderOfKey***

Section 2: SubjectConfirmation for Authentication Assertions - Trust Models

What is a Trust Model?

•Initial Conditions

–Initial knowledge state of each entity

•e.g. RP knows AuthnAuthority PublicKey

•Assumptions

–Conditions which are necessary for security but which we can't verify

•e.g. Signatures can't be forged

•e.g. AuthnAuthority private-key not compromised

•Rules

–Describe "who trusts whom for what if why". Form of rules is:

•*"Entity-X* **trusts** *Entity-Y* **for** *Z* **if** *Condition-C"*

•e.g. "*RP* **trusts** *AuthnAuthority* **for** *name-assertion* **if** *AuthnAuthority-Signature-Verifies"*

What Is SubjectConfirmation?

•Mitigation for threats against assertions in a particular environment of use

•Assertion requester specifies what SubjectConfirmation should be applied to the assertion to counter threats anticipated in environment of intended use

•Issuer applies requested SubjectConfirmation to assertion

•Relying Party uses SubjectConfirmation method specified in assertion to counter

threats

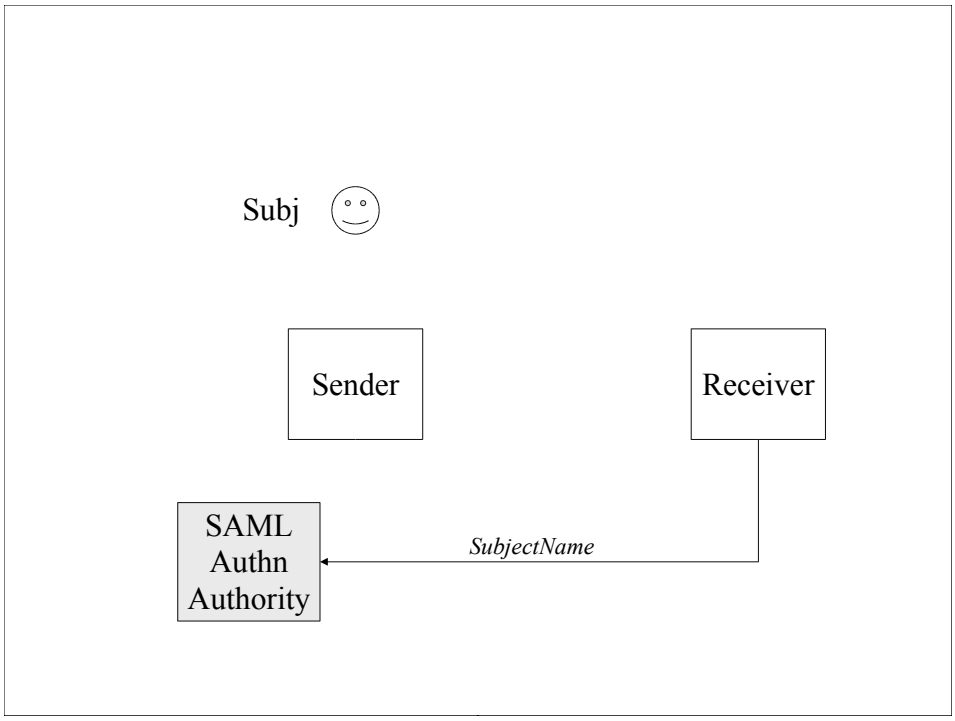

Figure 8: Bearer

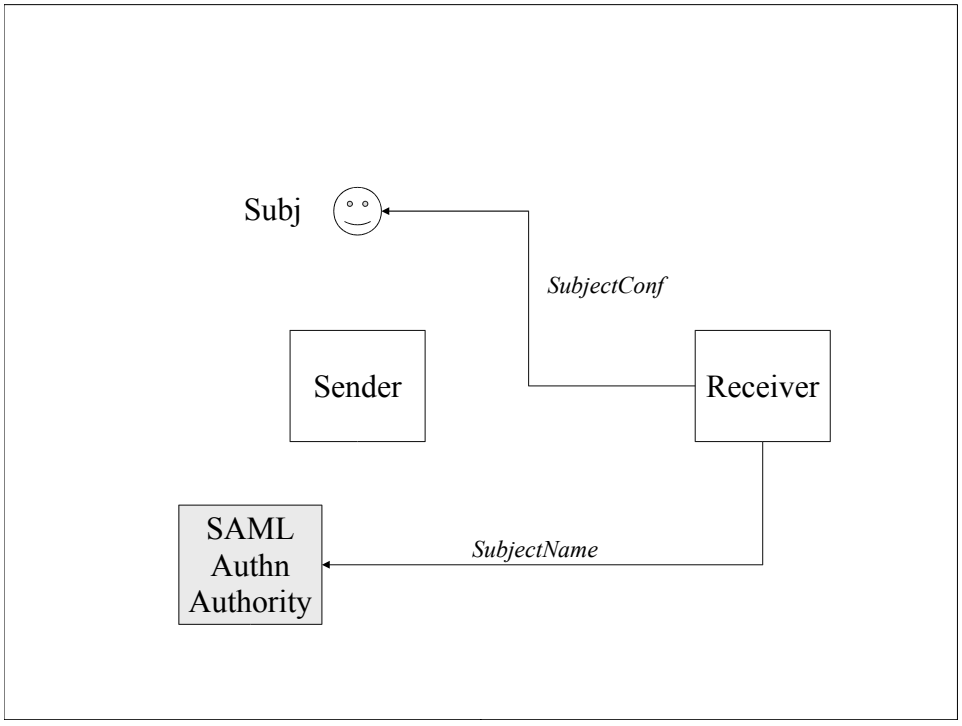

Figure 9: HolderOfKey

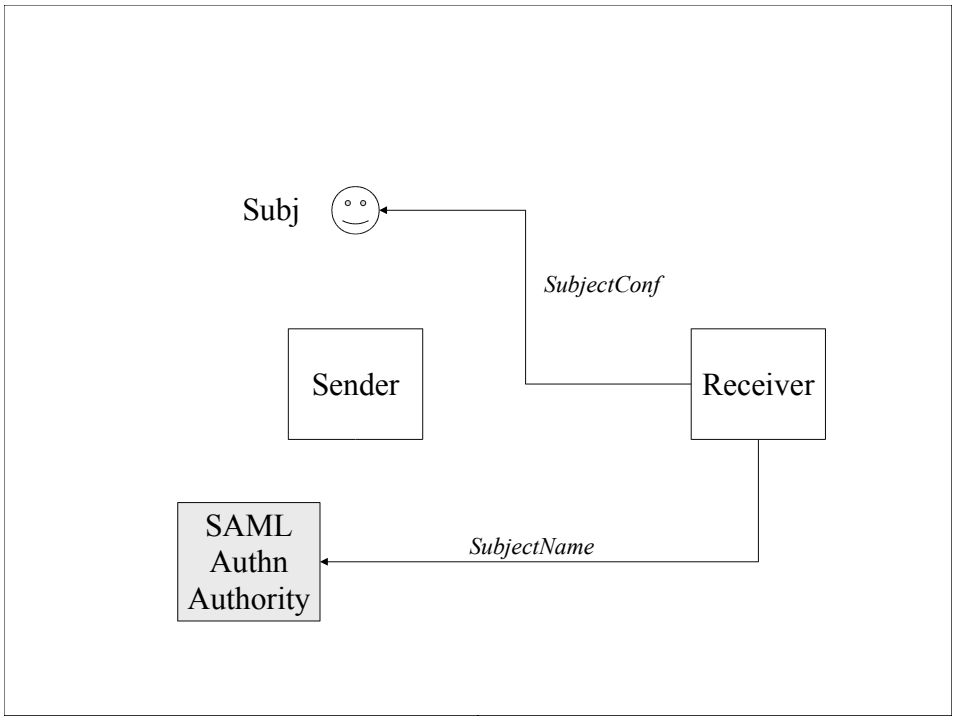

Figure 10: ChallengeProtocol

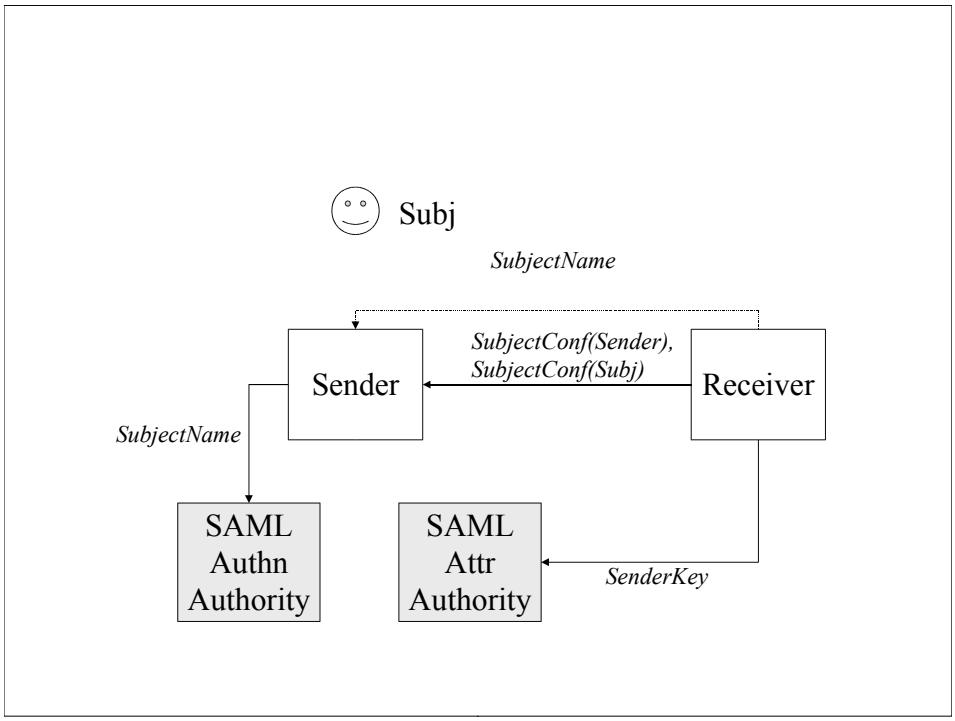

Figure 11: SenderVouches (Option 1)

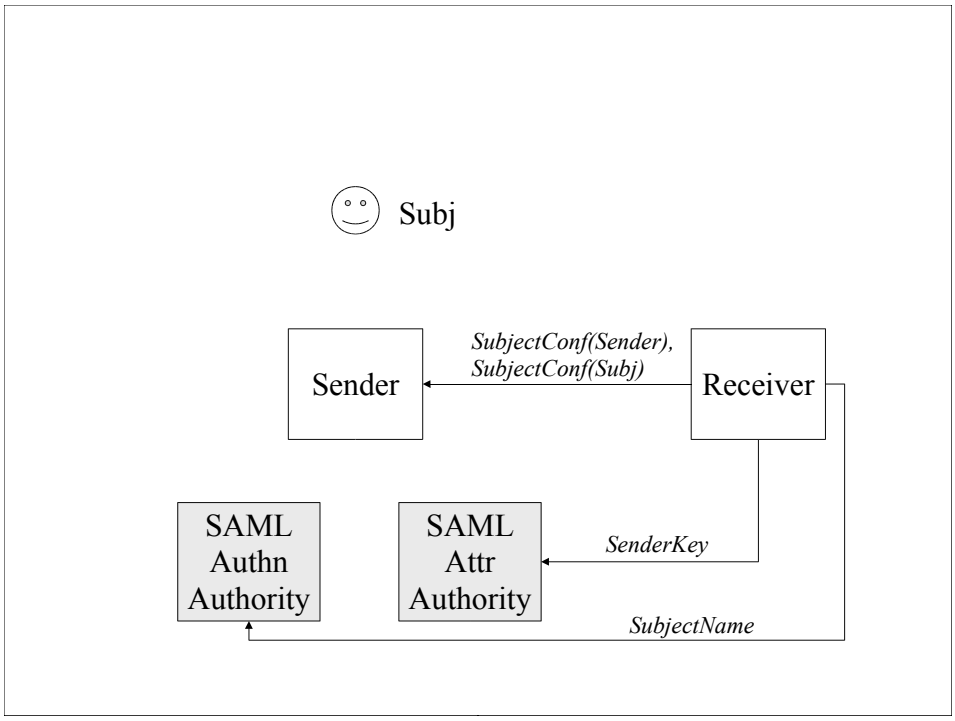

Figure 12: SenderVouches (Option 2)

Section 3: Semantics of SAML Subject Information

Information About Subjects

•Subject Designation

–Designates the subject to whom the statement(s) in an assertion refers.

•Subject Confirmation

–Designates a method which a relying party may use to confirm that an attribute it has received refers to the "correct" subject. This is used to counter various threats in specific bindings.

Semantics of Subject Designation Elements

•Subject Designation Elements used both in Assertions and in Queries:

–Subject/NameIdentifier

•The name of the subject to whom the assertion refers

–Subject/AssertionSpecifier

•The identifier of another assertion which designates the subject to whom the assertion refers

•Subject Designation Elements used in Queries only:

–Artifact

•A value which can be passed to an assertion issuer. An artifact designates a particular subject (or even a particular pre-existing assertion) to an issuer. However, an artifact conveys no information about the subject to whom an assertion refers to any party except the assertion issuer.

Semantics of SubjectConfirmation Element

•SubjectConfirmation

–The mechanism by which a relying party can satisfy itself that the assertion refers to the correct subject.

–This element does NOT designate any subject. It designates a mechanism by which the relying party confirming the correctness of the designation in the Subject Designator.

–This element should NOT be part of the Subject element

•It is used to "Confirm" subject, so it should not be an element of what it confirms

–This element SHOULD be part of the Query element

•Requesters need to be able to ask for an assertion whose SubjecConfirmation method addresses the threats which exist in the context in which the assertion will be used

•Only the SubjectConfirmation/ConfirmationMethod attribute should be specified in the Query

Semantics of SubjectConfirmation Attributes

•The Attributes of the SubjectConfirmation element are:

–SubjectConfirmation/ConfirmationMethod: values of this attribute include

•Bearer

•HolderOfKey (Hal proposes "name" as a way to get Cert/PubKey)

•ChallengeProtocol

•SenderVouches

–SubjectConfirmation/ConfirmationData

•Specifies the data needed to confirm subject using ChallengeProtocol method

–SubjectConfirmation/ds:KeyInfo

•Specifies the key needed to confirm subject using HolderOfKey method

Section 4: Receipt of Currently Invalid Assertions

Assertion Requester: Responsibility to Validate

•Requesting party SHOULD verify validity of each assertions returned in response to its request to an authority. Authorities MAY return expired assertions. Authorities MAY return assertions which are not yet valid, but will be valid starting at some time in the future.

Assertion Relying Party: Responsibility to Validate

•Relying party SHOULD verify validity of each assertions it receives before relying upon them. Senders MAY transmit expired assertions to Relying Parties. Senders MAY transmit to Relying Parties assertions which are not yet valid, but will be valid starting at some time in the future.