Architectures de fédération d'identités et interopérabilité

Mikaël Ates
mikael.ates@univ-st-etienne.fr

Christophe Gravier
christophe.gravier@univ-st-etienne.fr

Jeremy Lardon
jeremy.lardon@univ-st-etienne.fr

Jacques Fayolle
jacques.fayolle@univ-st-etienne.fr

Bruno Sauviac
bruno.sauviac@univ-st-etienne.fr

Equipe SATIn – Laboratoire DIOM
Institut Supérieur des Techniques Avancées de Saint-Etienne
23 rue Paul Michelon 42023 Saint-Etienne

Résumé

La gestion d'identités numériques intra et inter-systèmes d'informations, ainsi que la sécurisation des architectures orientées services, sont les fondamentaux de la fédération d'identités. Ce type d'architecture est destiné à l'interopérabilité des systèmes d'informations.

Cependant, les architectures existantes n'offrent pas de solutions d'interopérabilité dans une architecture hétérogène. Dans cet article, nous essayons d'initier *une réflexion en la matière grâce à l'étude de deux spécifications majeures: SAML et WS-Federation.*

Dans un premier temps, nous dressons un portrait de la fédération d'identités. Puis, nous traitons de l'interopérabilité d'architectures de fédération différentes. Ensuite, nous étudions les similitudes et différences de SAML2 et WS-Federation1.1B afin de donner les voies de l'interopérabilité entre ces deux architectures.

Mots clefs

Identité, Fédération, Confiance, SAML, WS-Federation, WS-Trust, WS-Security, Interopérabilité, Architecture.

1 Introduction

La mise à disposition de ressources numériques entre partenaires peut se traduire par l'interconnexion des systèmes d'informations. Une des problématiques soulevée par cette interconnexion est la gestion des identités, permettant le contrôle d'accès aux ressources.

La fédération d'identités vise à offrir une solution d'interopérabilité des systèmes de gestion des identités. Ce qui ouvre la voie à l'interconnexion des domaines de sécurité des systèmes d'informations.

La fédération d'identités a plusieurs origines :

- La multiplicité des processus administratifs des architectures de gestion des identités (Authentication, Authorization, Accounting et Auditing – AAAA)
- La sécurisation des architectures orientées services.
- La gestion des identités entre systèmes d'informations.

Deux architectures issues des technologies du Web, respectivement d'authentification unique et de sécurisation des architectures orientées services, ont chacune abouti, par des évolutions successives, à la conception d'architectures de fédération d'identités majeures: SAML[1] et WS-Federation[2].

Dans cet article, nous abordons la problématique de l'interopérabilité de ces deux architectures. Nous décrivons, dans un premier temps, les notions générales d'une architecture de fédération. Puis, nous traitons la problématique de l'interopérabilité d'architecture de fédération différentes. Ensuite, nous comparons sommairement, SAML2[1] et WS-Federation1.1B[2][3], afin de définir les points de convergence et donc les voies de l'interopérabilité.

2 Introduction à la confiance

La fédération consiste à définir une architecture permettant le transport d'informations portant sur des identités numériques. Ceci, dans le but d'offrir l'accès à des ressources de domaines de sécurité différents de ceux d'appartenance des identités. Ce principe gravite autour de trois problématiques.

[1]Security Assertion Markup Language

[2]Web Services Federation Language

2.1 La délégation des tâches administratives

Dans un environnement où chaque application est en charge d'assurer l'exécution des procédés administratifs de gestion des identités, la saisie répétée d'identifiants est un problème majeur en termes de confort d'utilisation des applications et de transmission répétée d'identifiants sur le réseau. Les architectures d'authentification unique basées sur la délégation des procédés administratifs auprès de tiers de confiance apportent une solution à ce problème. Ce type d'architectures repose sur l'établissement de liens de confiance entre fournisseurs de services et autorités en charge des procédés administratifs. Les entités ainsi liées forment un cercle de confiance. Les fournisseurs de services acceptent les affirmations faites par les autorités administratives, qui ont la charge de fournir la preuve de leur identité auprès des fournisseurs de services. Cela se traduit dans la pratique par le partage de secrets permettant la signature des affirmations, généralement, grâce à une infrastructure à clés publiques.

2.2 L'ouverture des systèmes d'informations

L'ouverture des systèmes d'informations permet la mise à disposition de ressources internes à des intervenants extérieurs, humains ou logiciels. Pour des raisons évidentes de charge administrative, et donc de coût financier, un même système d'informations ne peut directement prendre en charge la gestion des identités des systèmes tiers auxquels appartiennent ces intervenants. Cette ouverture doit donc s'accompagner de l'ouverture des systèmes de gestion des identités, et de la mise en place d'infrastructures d'interconnexion de ceux-ci. L'établissement de liens de confiance entre les systèmes d'informations souhaitant interopérer est donc dans ce cas également justifié. Les systèmes d'informations appartenant globalement à des entités organisationnelles tierces, la notion de fédération prend tout son sens. Il s'agit, dans un premier temps, de formaliser administrativement un partenariat afin de former une fédération. Cette fédération définit ensuite sa propre politique de gestion des identités, entre systèmes, et basée sur des liens de confiance.

2.3 Sécurisation des architectures orientées services

L'interconnexion d'applications intra ou inter-systèmes d'informations, et la conception d'applications inter-systèmes d'informations, sont des problèmes complexes en terme de gestion des identités. Il est en effet nécessaire d'assurer la sécurité des échanges, mais aussi de gérer les identités de clients applicatifs autonomes. Qui plus est, une architecture de fédération d'identités basée sur les technologies du Web est une architecture orientée services. Elle apporte une couche de sécurité et de gestion des identités au sein d'une architecture orientée service existante, en conditionnant les échanges de messages, et en ajoutant des informations de sécurité aux entêtes des messages échangés.

3 Architecture de fédération

Nous abordons dans cette partie les principes généraux d'une architecture de fédération.

3.1 Exprimer une information de sécurité

Les trois observations précédentes reposent sur la circulation, intra ou inter-systèmes, d'informations de sécurité sur les identités. Ces informations sont de deux types :

- descriptives (identifiants et attributs) ;
- résultantes de l'exécution des procédés administratifs de gestion des identités.

3.2 Architecture de confiance et cycle de vie des informations de sécurité

Une architecture de confiance repose sur l'établissement de liens de confiance basés sur le partage de secrets permettant la signature des messages. L'architecture de confiance doit permettre de gérer le cycle de vie des informations de sécurité par un protocole de requêtes simple, intégrant la création, le renouvellement, la validation, et l'annulation des informations de sécurité. L'architecture de confiance doit permettre diverses topologies de confiance basées sur des liens de confiance directs et indirects. Enfin, l'architecture de confiance doit définir la sécurisation des messages véhiculant les informations de sécurité : intégrité, confidentialité, anti-rejeu et durée de vie.

3.3 Définir les protocoles de la fédération d'identités et les rôles des entités

Une architecture de fédération repose sur l'architecture de confiance pour véhiculer les informations de sécurité et étend son protocole afin de répondre aux fonctionnalités de fédération, à savoir, la délégation des procédés administratifs, et le conditionnement du cycle de vie des informations de sécurité. Une architecture de fédération définit les rôles principaux des entités, ainsi que son implémentation au sein de protocoles applicatifs de transport.

3.4 Le Client

Les architectures de fédération doivent prendre en compte les différentes interactions, d'un utilisateur avec le système, et d'un applicatif autonome. Cela suppose deux types de clients :

- le navigateur Web, relais passif et interface utilisateur;
- le client riche, interface utilisateur ou applicatif autonome, capable d'interagir directement au sein de l'architecture en consommant des services Web.

4 Interopérabilité d'architectures de fédération hétérogènes

La fédération permet d'offrir un service à une identité appartenant à un domaine de sécurité tiers. Pour cela, une

autorité est en charge de fournir des informations sur cette identité au fournisseur de service, avec lequel elle possède un lien de confiance, direct ou indirect. La problématique traitée dans cet article consiste à permettre à une autorité de fournir des informations à un fournisseur de service, alors que ces deux sont issues d'architectures de fédération différentes.

La prise en charge de l'interopérabilité est à attribuer à l'autorité ou le fournisseur de service s'ils possèdent un lien de confiance direct.

Si la topologie de l'architecture est centrée sur l'autorité, la charge de l'interopérabilité est à attribuer à l'autorité (*cf. fig1*).

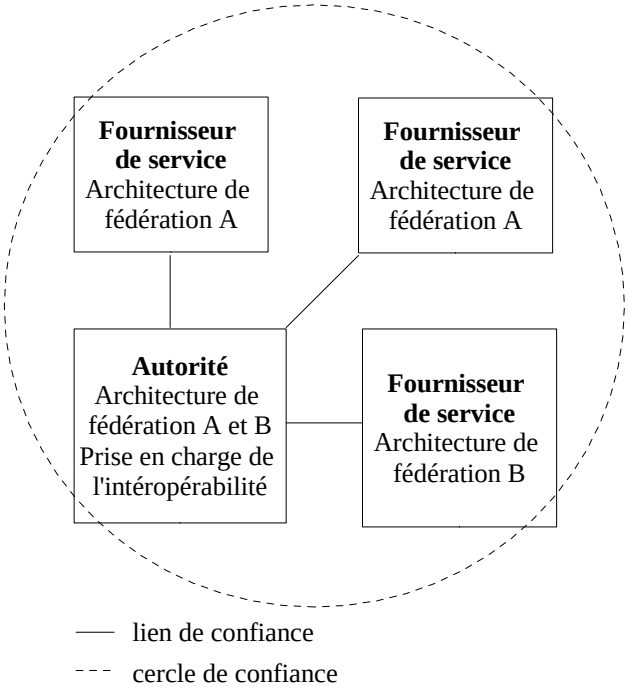
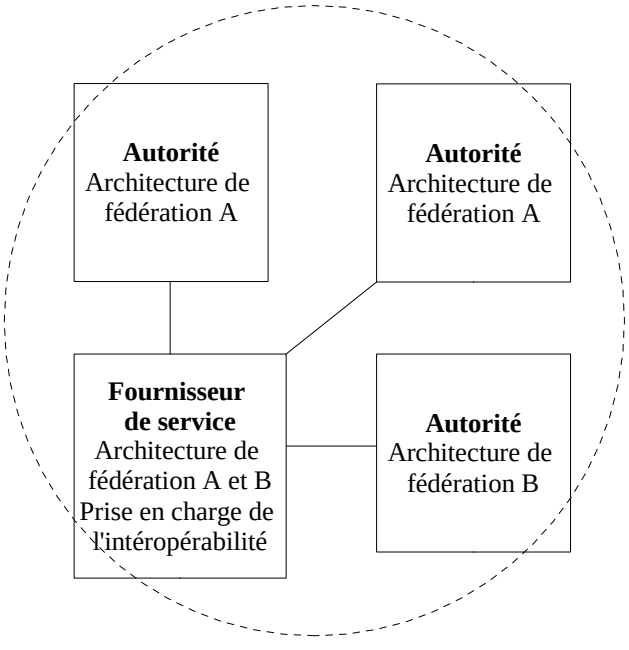


fig 1. Prise en charge de l'interopérabilité pour une architecture de fédération centrée sur l'autorité avec lien de confiance direct.

Si la topologie est centrée sur le fournisseur de service, celui-ci a la charge de l'interopérabilité (*cf. fig2*).



lien de confiance

cercle de confiance

fig 2. Prise en charge de l'interopérabilité pour une architecture de fédération centrée sur le fournisseur de services avec lien de confiance direct.

Cependant, dans des topologies plus complexes, où les cercles de confiance intègrent de multiples autorités et fournisseurs de services, il est intéressant de confier la charge de l'interopérabilité à un tiers dédié (*cf. fig3*).

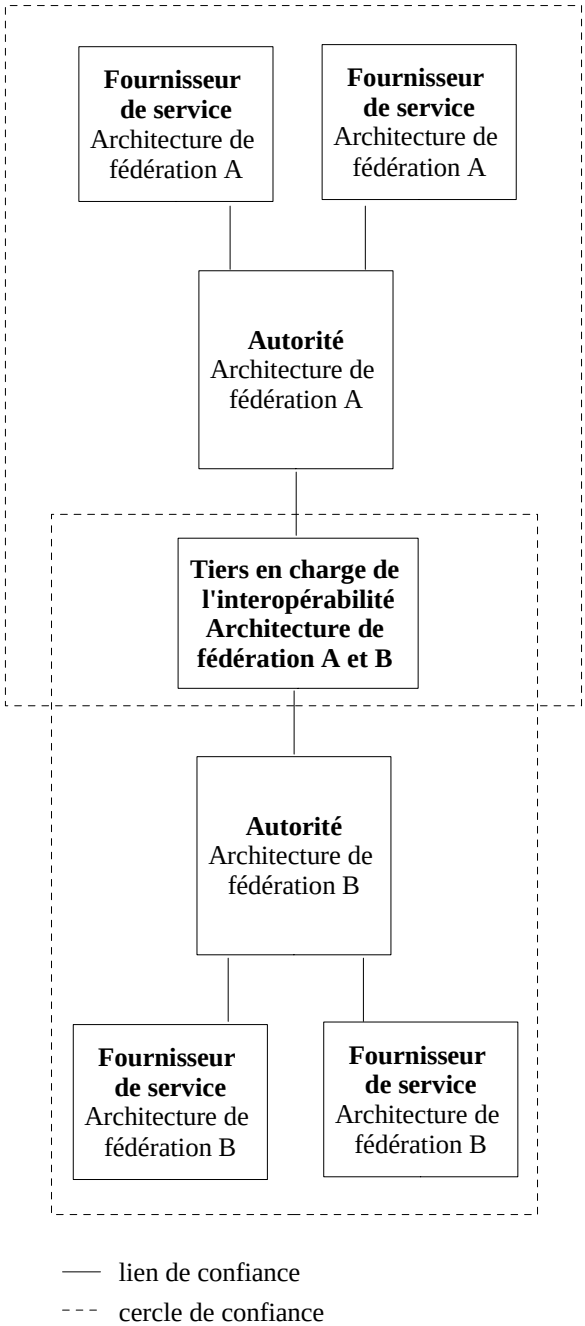


fig 3. Prise en charge de l'interopérabilité pour une architecture de fédération complexe service avec lien de confiance indirect.

Ainsi, une autorité et un fournisseur de service, qui interagissent en étant de spécifications différentes, ne sont pas modifiés par la prise en charge de l'interopérabilité, puisque celle-ci est assurée par un tiers. Ils possèdent avec celui-ci un lien de confiance direct, qui les relie donc indirectement.

5 Principe de l'interopérabilité entre SAML2 et WS-Federation1.1B

Nous illustrons cela en étudiant les spécifications SAML2 et WS-Federation1.1B afin de déterminer les opérations nécessaires à l'interopérabilité, ainsi que les voies de l'implémentation du tiers dédié à l'interopérabilité.

Les consortiums OASIS[3], avec la norme SAML, et Liberty Alliance, avec les spécifications ID-FF[4] basées sur SAML, ont travaillé sur des architectures de fédération très proches ayant conduit à l'élaboration de SAML2.

Un consortium comprenant entre autres Microsoft, IBM, Novell, VeriSign et Computer Associates, a développé une architecture de fédération appelée WS-Federation qui repose sur d'autres spécifications, principalement WS-Trust[4] et WS-Security[5], toutes deux normalisées par l'OASIS.

L'objectif de notre étude est de permettre à un fournisseur de service SAML2 d'obtenir une information de sécurité de la part d'une autorité WS-Federation1.1B et inversement. Les fondamentaux de l'interopérabilité repose sur la mise en correspondance de l'expression des informations de sécurité et des protocoles de requêtes de ces informations. Pour cela, nous effectuons dans un premier temps une comparaison sommaire de ces deux architectures.

6 Comparaison entre SAML2 et WS-Federation1.1B

6.1 Informations de sécurité

La norme SAML fournit le schéma XML de préfixe *saml* et d'espace de nom *urn:oasis:names:tc:SAML:2.0:assertion* des informations de sécurité appelées « *assertions*[5] ». Une assertion indique sa source, l'identité assujettie décrite par un identifiant (« *nameID* ») sous divers formats (« *email* », « *x509 subject name* », « *Windows domain qualified name* », « *Kerberos principal name* », « *entity identifier* », « *persistant identifier* » et « *transient identifier* ») ainsi que l'information à véhiculer. Cette information est scindée en trois sections facultatives appelées « *statements*[6] »: authentification, autorisation ou attributs du sujet.

La norme WS-Security fournit un schéma XML de préfixe *wsse* et d'espace de nom *http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd* des informations de sécurité appelées « jetons de sécurité ». Un jeton de sécurité peut être de type « nom d'utilisateur » (« *username* »), binaire (« certificats X509 » ou « tickets Kerberos ») ou XML (par exemple une *assertion* SAML). Les jetons de sécurité véhiculent des informations appelés « *claims*[7] ». L'objectif initial de WS-Security est de sécuriser des messages SOAP[8] en plaçant des jetons de sécurité dans les entêtes des messages à des fins de chiffrement et de signature. Comme nous le verrons par la suite, WS-Security peut également être utilisé pour véhiculer des informations de sécurité dans une architecture de confiance, le jeton de sécurité étant placé dans le corps d'un message SOAP.

6.2 Fédération d'identités

Les architectures de fédération proposent chacune un modèle destiné à répondre à des fonctionnalités précises. Ce modèle définit le rôle des entités, la description de celles-ci par des méta-données, des topologies d'architecture, un ensemble de profils d'exploitation et leur implémentation au sein des protocoles standards du Web.

6.2.1 Sémantique

En SAML, l'autorité en charge de produire des *assertions* est appelée « fournisseur d'identités » (« *Identity Provider* » – IP), le tiers consommateur d'*assertions* est appelé « fournisseur de services » (« *Service Provider* » – SP), enfin, l'identité qui fait l'objet de la transaction est appelée « principal ». Le principal désigne le sujet des informations de sécurité véhiculées et non pas le client passif, à savoir le navigateur Web.

En WS-Federation, l'autorité en charge de produire des jetons de sécurité est appelée « serveur de jetons de sécurité fournisseur d'identités » (« *Identity Provider Security Token Server* » – IP/STS[9]), le tiers consommateur de jetons de sécurité est appelé « *Relying Party* » (ou « *Ressource Provider* » – RP), enfin, le client source des requêtes de jetons est appelé « requêteur » (« *Requestor* »). Il est à noter que ce rôle désigne le client SOAP, et non pas, le sujet des informations de sécurité véhiculées. En effet, WS-Federation a été conçu pour un fonctionnement avec un client SOAP constituant une interface utilisateur ou un applicatif autonome.

Dans la suite de l'article, nous désignerons par SP le fournisseur de services consommateur d'informations de sécurité, *assertions* ou jetons, et IP le fournisseur d'identités, producteur de celles-ci.

---

[3] Advancing Open Standards for the Information Society

[4] Identity-Federation Framework

[5] « Assertion » peut être traduit par « affirmation ». Cependant, d'autres termes relatifs aux informations de sécurité peuvent être traduit par « affirmation », ce qui porte à confusion. Nous conserverons donc la dénomination anglophone.

[6] « Statement » peut aussi être traduit par « affirmation ». Nous conserverons la dénomination anglophone (cf note de bas de page précédente).

[7] « Claim » peut aussi être traduit par « affirmation ». Nous conserverons la dénomination anglophone (cf note de bas de page précédente).

[8] Simple Object Architecture Protocol

[9] Les termes IP, STS et IP/STS peuvent être utilisés de façon équivalente.

6.2.2 Méta-données de fédération

Les méta-données de fédération permettent de divulguer les clés publiques et de décrire les références terminales des services implémentant les protocoles de la fédération pour chacune des entités.

En SAML, les méta-données sont échangées préalablement à la fédération. En WS-Federation, les méta-données peuvent être échangées dynamiquement grâce au protocole WS-MetadataExchange[7].

6.2.3 Confidentialité des données personnelles

Une assertion SAML porte sur un sujet désigné par un identifiant unique (*« nameID »*). Celui-ci peut être un identifiant du sujet ou un pseudonyme. Il existe deux types de pseudonymes assurant la confidentialité des données personnelles au sein de SAML :

- L' « identifiant variable » (« *transient identifier* »), qui est un alias variant à chacune des sessions du sujet et propre à chaque lien entre IP et SP . Le sujet ne peut donc pas être associé à un compte local du fournisseur de service.
- L'« identifiant persistant » (*« persistent identifier »*), qui est un pseudonyme invariant du sujet et propre à chaque lien entre IP et SP. Cela permet à la fois de préserver l'anonymat des échanges et de procéder à une association de comptes.

WS-Federation permet deux mécanismes équivalents en terme de confidentialité :

- L'identifiant « Pairwise » dont le fonctionnement est identique à l'identifiant persistant de SAML.
- Un pseudonyme équivalent à l'identifiant variable de SAML mais qui, couplé à un service d'enregistrement de pseudonymes, permet au SP de faire une association de comptes.

6.2.4 Architecture de fédération

SAML2 a été initialement conçu pour répondre aux problématiques d'authentification unique au sein d'une architecture de fédération. Cette architecture a donc été conçue pour un client passif (navigateur Web). SAML fournit un second schéma XML de préfixe *samlp* et d'espace de nom *urn:oasis:names:tc:SAML:2.0:protocol* définissant les échanges protocolaires. Le déclenchement des échanges protocolaires est à la charge du fournisseur de services lorsqu'il souhaite établir un contexte de sécurité. SAML2 est composé d'un ensemble de protocoles précis permettant de répondre aux profils de fonctionnement: authentification unique pour navigateur Web, clients et relais enrichis, déconnexion unique, découverte du fournisseur d'identités, gestion des identifiants de nom, résolution d'artefact, requêtage d'*assertions*, association d'identifiants de nom et requêtage d'attributs. Les différents protocoles conditionnent les *statements* qui seront contenus dans les *assertions*. L'utilisation d'un navigateur Web comme interface utilisateur implique que les mécanismes de transport des messages protocolaires se base sur HTTP 1.1 pour les échanges, où le navigateur Web n'intervient qu'en tant qu'intermédiaire passif. Cependant, les échanges directs entre producteurs et consommateurs d'*assertions* se font en SOAP.

WS-Federation se base sur WS-Trust. WS-Trust a été initialement conçu pour réaliser une architecture de confiance afin de sécuriser une architecture orientée services Web. WS-Trust fournit le schéma XML de préfixe *wst* et d'espace de nom *http://docs.oasis-open.org/ws-sx/ws-trust/200512*. WS-Trust définit une architecture basée sur un client actif, le requêteur, acteur des échanges protocolaires de l'architecture de fédération. Il est à l'origine des requêtes de jetons de sécurité. En pratique, le requêteur peut être un navigateur Web riche, donc une interface utilisateur, ou une application autonome, consommatrice de services Web dans le cadre d'un déploiement pour sécuriser une architecture orientée services. WS-Trust définit un protocole généraliste de requêtes entre requêteurs et serveur de jetons. Ce protocole permet de gérer le cycle de vie des jetons : création, renouvellement, annulation et validation. WS-Trust repose sur WS-Security pour assurer la sécurité des messages et définir les informations de sécurité à véhiculer. WS-Federation définit un modèle général où chaque service Web possède un fichier de description WSDL[10]. Celui-ci indique les capacités du service, en terme de fonctionnalités de fédération, et de ses pré-requis de sécurité, sous la forme d'un document XML au format WS-SecurityPolicy[8], aussi appelé « politique ». Les capacités du service au sein de l'architecture de fédération sont appelées méta-données de fédération. Elles indiquent notamment la localisation des emplacements terminaux SOAP (*SOAP endpoints*). Les pré-requis définis par une politique indique les *claims* qui devront être contenus dans les jetons de sécurité qui lui sont présentés pour établir un contexte de sécurité. Réciproquement, le requêteur déduit de la politique les *claims* qu'il doit obtenir auprès des serveurs de jetons. WS-Federation définit différents profils pour les serveurs de jetons en fonction du type de jetons qu'ils peuvent fournir : authentification, autorisation, attributs et pseudonymes. On notera que le protocole d'obtention des jetons est généraliste, c'est à dire qu'il ne définit pas différents types de requêtes en fonction du type de jeton que l'on souhaite obtenir, à la différence de SAML. Ce sont donc les paramètres de la requête qui varient afin de déterminer le type de jetons délivré.

6.2.5 Les architectures de fédération en terre inconnue

SAML définit un profil pour client riche appelé ECP (*« Enhanced Client Profile »*), c'est à dire un client actif susceptible de prendre part aux échanges protocolaires. Cependant, le SP reste à l'origine des échanges. Lorsque le client souhaite accéder à une ressource du SP nécessitant un contexte de sécurité, celui-ci retourne une requête SOAP au sein de la réponse HTTP (*« binding/protocol PAOS »*). Le client est en charge de détecter la requête SOAP qui contient la demande d'authentification SAML signée du SP. Dans l'entête de la requête SOAP, le SP fournit la liste de ses IP de confiance. Le client choisit un des IP et adresse au service Web sélectionné la requête d'authentification. Avant de faire sa requête, le client

[10]Web Services Description Language

extrait la demande d'assertion de l'enveloppe SOAP et re-formate une nouvelle requête SOAP. S'il n'y a pas déjà de contexte de sécurité établi avec l'IP, ou que le SP force la ré-authentification, le client s'authentifie. L'IP délivre une *assertion* contenant un *authentication statement*. Le client retourne l'*assertion* dans une réponse SOAP au SP grâce à une requête HTTP (*« binding/protocole PAOS »*). Le SP établie alors un contexte de sécurité pour le principal et délivre la ressource initialement requêtée. Pour cela, un paramètre *« RelayState »* est initialisé par le SP au sein de l'entête de la requête SOAP initiale. Ce paramètre est ensuite passé dans toutes les entêtes de requêtes/réponses SOAP.

WS-Federation fournit un profil pour client passif. L'implémentation d'un client passif impose que le SP soit actif, c'est à dire qu'il soit à l'origine des échanges protocolaires. Outre le fait qu'il doit donc avoir « conscience » des *claims* qu'il requière, il devient requêteur. Cela suppose la définition d'un protocole de requêtes spécifiques, ou, que celles-ci soient exprimées au travers de paramètres de la requête HTTP lorsque le SP redirige l'utilisateur vers un serveur de jetons.

6.2.6 Les autorisations

La gestion des autorisations ne fait plus partie des prérogatives de SAML2. Le langage XACML[11] issue de l'OASIS est une architecture de gestion des autorisations dont les décisions peuvent être véhiculées au sein des *attribute statements* d'une *assertion*.

L'extension WS-SecurityPolicy de l'architecture WS-Policy permet de définir les politiques de chacun des services de l'architecture. Ceci constitue l'architecture d'autorisation de WS-Federation.

6.2.7 Architecture de confiance

En SAML, un IP peut agir en relais lors d'une requête d'authentification. Ainsi, s'il ne peut satisfaire une requête, il peut émettre une requête d'authentification auprès d'un autre IP, et se comporter ainsi, en un consommateur d'*assertions*. L'IP dans le rôle de relais aura la charge de re-signer l'*assertion*. Ce mécanisme a été conçu pour un client passif mais fonctionne également avec un client actif. Dans ce cas, l'IP retourne une requête SOAP, au lieu d'une réponse SOAP contenant une *assertion* avec un *authentication statement*.

En WS-Federation, lorsque le client a déterminé les *claims* dont il a besoin pour accéder à la ressource, et que son IP est différent de celui de la ressource, il obtient un jeton auprès d'un IP, qu'il devra pouvoir échanger contre un nouveau jeton issu de l'IP de la ressource et comportant les *claims* requis. En client passif, comme en SAML, le fournisseur de service redirige vers son IP, qui lui-même redirige vers l'IP du client.

La « relayage » de requête pour SAML, et l'échange de jetons pour WS-Federation, sont les fondamentaux pour bâtir des architectures de confiance avec des liens de confiance indirects entre producteurs et consommateurs d'informations de sécurité. Les liens de confiance indirects permettent le « chaînage » d'IP et donc l'établissement d'architectures de confiance hiérarchiques. C'est sur ce principe que se base le tiers en charge de l'interopérabilité.

[11]eXtensible Access Control Markup Language

7 Interopérabilité et client passif

7.1 Principe

La problématique de l'interopérabilité est fondamentalement identique qu'il s'agisse d'un client actif ou passif. Nous illustrons donc notre approche en traitant l'architecture avec client passif.

Il n'existe pas aujourd'hui d'implémentation complète des profils actif et passif de WS-Federation1.1B. Le service ADFSv1 sur Windows Server 2003 R2 est une implémentation du profil d'interopérabilité, sous-ensemble du profil pour client passif de WS-Federation1.0. Des travaux ont été menés pour faire interopérer ce service avec Shibboleth v1.3f, basé sur SAML1.1[11]. Ces travaux ont mené à l'élaboration d'un module d'extension de Shibboleth: *« library.adfs.so »*.

Le principe de cette interopérabilité consiste à « traduire » les requêtes protocolaires SAML en requêtes du profil d'interopérabilité pour client passif de WS-Federation. Les documents XML de requêtes et réponses SAML sont respectivement traduit en paramètres URL et en un document WS-Trust *<wst:RequestSecurityTokenResponse>*. Au niveau de l'architecture choisie, le module d'interopérabilité est à installer directement sur les SP et IP Shibboleth qui souhaitent interopérer. Le « chaînage » d'IDP n'est pas supporté par Shibboleth v1.3f ce qui empêche l'implémentation d'un tiers en charge de l'interopérabilité.

Les spécifications du profil passif de WS-Federation offre deux possibilités au SP pour effectuer sa requête auprès d'un IP:

- Par un ensemble de paramètres de l'URL. Solution exploitée dans l'interopérabilité ADFS/Shibboleth.
- Par le passage d'un document de requête WS-Trust au sein d'un paramètre de l'URL.

Cette deuxième possibilité nous apparaît très intéressante. C'est donc celle que nous avons retenu dans le but de faire interopérer le profil passif de WS-Federation et les profils SAML.

La principale tâche résultante de l'interopérabilité consiste alors à « traduire »:

- Un document WS-Trust *<wst:RequestSecurityToken>* en documents SAML de requête, par exemple *<samlp:AuthnRequest>*, et inversement.
- Un document WS-Trust *<wst:RequestSecurityTokenResponse>* en *<samlp:Response>* SAML, et inversement.

Le tiers assurant l'interopérabilité sera donc en charge d'effectuer les opérations nécessaires de transformation en assurant la « re-signature » des informations de manière à établir le lien de confiance indirect.

Illustrons ce principe pour la délégation de l'authentification.

7.2 Transport

En WS-Federation comme en SAML, le SP redirige le client vers l'IP à l'aide de l'erreur HTTP 302. La requête d'authentification se traduit par des paramètres de l'URL:

- *SAMLRequest* contenant un document XML de l'espace de nom SAML de type *<samlp:AuthnRequest>*, pour SAML.
- *wa* contenant la valeur *wsignin1.0* et *wreq* contenant un document XML de l'espace de nom WS-Trust de type *<wst:SecurityTokenRequest>* pour WS-Federation.

La réponse résultante d'une authentification valide se traduit en SAML, comme en WS-Federation, par un POST HTTP avec pour paramètre:

- *SAMLResponse* contenant une assertion pour SAML.
- *wresult* contenant un document XML de l'espace de nom WS-Trust nommé *<wst:SecurityTokenRequestResponse>* contenant une *assertion* pour WS-Federation.

7.3 Requêtes

Il s'agit ici de faire correspondre les requêtes SAML et les requêtes WS-Trust exploitées dans le cadre de WS-Federation.

7.3.1 SAML « AuthnRequest »

Comme nous l'avons précédemment souligné (6.2.4), les requêtes WS-Trust ne sont pas « typées ». Il faut donc traduire le fait que l'information attendue en retour d'une requête *<samlp:AuthnRequest>* soit le résultat d'un processus d'authentification. De plus, le type de jeton attendu en retour de la requête WS-Trust *<wst:RequestSecurityToken>* est une *assertion* et doit donc contenir l'élément :

<wst:TokenType> urn:oasis:names:tc:SAML: 2.0:assertion </wst:TokenType>.

Le résultat de la requête attendu est une production de jeton et doit donc contenir l'élément :

<wst:RequestType>http://docs.oasis-open.org/ws-sx/ws-trust/200512/Issue</wst:RequestType>.

7.3.2 Nom du sujet

Une requête SAML de type *<samlp:AuthnRequest>* contient l'élément *<samlp:NameIDPolicy>* qui indique le type de nom du sujet attendu en réponse. Cela peut se traduire par la demande d'un *claim* d'autorisation de même type. S'il s'agit d'une adresse email, on obtient pour ces deux requêtes:

- *<samlp:authnRequest>:*

<samlp:NameIDPolicy Format="urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress" </samlp:NameIDPolicy>

- *<wst:SecurityTokenRequest>:*

<wst:Claims Dialect=”http://schemas.xmlsoap.org/ws/2006/12/authorization/authclaims”><auth:ClaimType Uri="urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress" /> </wst:Claims>

7.3.3 Niveau d'authentification

La requête d'authentification SAML permet de stipuler le type d'authentification attendu par le SP. Il s'agit du contexte d'authentification qui se traduit par un élément *<samlp:RequestedAuthnContext>*. Dans une requête WS-Trust, cela se traduit par un élément *<wst:authenticationType>* pour lequel WS-Federation définit un ensemble de valeurs.

Les valeurs prédéfinies par WS-Federation ne sont pas comparables avec les 25 schémas SAML de contextes d'authentification. Cependant, il est possible pour l'implémentation de WS-Federation de prendre en charge les contextes d'authentification SAML.

7.4 Réponses

Il s'agit ici de faire correspondre les reponses SAML et les réponses WS-Trust exploitées dans le cadre de WS-Federation. La réponse contenue dans la réponse WS-Trust *<wst:SecurityTokenRequestResponse>* est une assertion SAML. Le tiers aura donc la charge de l'extraire, de la reformater dans une réponse SAML *<samlp:Response>* et de la « re-signer ».

7.5 Généralisation

Nous n'avons explicité que le principe sur lequel nous souhaitons nous baser pour faire interopérer SAML2 et WS-Federation1.1B. Pour être exhaustif et ainsi permettre de spécifier cette interopérabilité il est nécessaire d'appliquer ce principe pour traduire chaque requête protocolaire SAML en requête WS-Trust. De nombreux autres paramètres que ceux cités précédemment sont à prendre en compte. Par exemple, les paramètres traitant de la durée de validité.

Il est également nécessaire, pour la fourniture d'attribut, d'établir un espace de nom commun. WS-Federation propose un espace de nom des *claim*s très limité basé sur le document non normatif intitulé Passive Requestor Interoperability Profile [12]. SAML offre cinq profils d'attributs: basic, X500/LDAP, UUID, DCE PAC et XACML.

Il est également nécessaire de mettre en correspondance le système de pseudonymes de WS-Federation, assuré par un serveur de jetons dédié, et celui de SAML, assuré par le fournisseur d'identités à l'aide de pseudonymes temporaires ou persistants.

Enfin, il n'existe pas de systèmes de découverte dynamique des méta-données en SAML. Il s'agit de ce que l'on peut considérer comme le « plus petit dénominateur commun ». Les liens de confiance entre les architectures et le tiers de confiance devront donc être pré-établis.

L'interopérabilité pour client actif se base sur le même principe. Il faut cependant ajouter à cela le fait qu'il devient également actif pour l'interopérabilité, non pas pour la conversion des données et des requêtes, mais parce qu'il devra implémenter les mécanismes de transport du requêteur WS-Trust et du profil SAML ECP.

8 Etat actuel de la question de l'interopérabilité

L'architecture de fédération IDFF1.2[6] a contribué à l'élaboration de SAML2. SAML2 et IDFF1.2 sont deux architectures très proches mais non interopérables [13]. SAML2 n'est de plus pas interopérable avec ses versions antérieures (1.x). Il existe de nombreuses implémentations conformes SAML2 : Lasso (Entrouvert), I-dLive (NTT), Identity Management (Oracle), PingFederate (Ping Identity Corp.), Site Minder (CA), ou bien encore, OpenView Select Federation (HP).

Shibboleth v1.3f, implémentation basée sur SAML1.1, n'est donc pas nativement interopérable avec toute implémentation de SAML2 ou IDFF1.2.

Les spécifications Liberty Alliance Phase 1, IDFF1.2 basée sur SAML1.1, et WS-Federation 1.0 ont vus des efforts communs entre SUN (Liberty Alliance) et Microsoft afin de produire des spécifications visant à l'interopérabilité. Ces travaux ont abouti à l'élaboration de deux nouvelles spécifications, le protocole Web Single Sign On MetadataExchange[9] et le profil Web Single Sign On Interoperability[10]. Ces spécifications ont pour objectifs de permettre à un SP de découvrir l'IP du client et de lui permettre de découvrir les protocoles supportés par l'IP. A titre d'exemple, des profils interopérables ont été déterminé : « Liberty ID-FF1.2 browser POST profile » et « WS-Federation Passive Requestor Profile ». Ces spécifications sont très sommaires et n'ont pas été mises à jour depuis leur première parution.

9 Conclusion

Les architectures de fédération sont encore en mutation. Cependant, des tendances générales se dégagent et vont dans le sens de la convergence des architectures.

La philosophie de « l'identité 2.0 » place l'utilisateur au coeur des échanges sur son identité. Cela correspond parfaitement aux architectures à client riche : WS-Trust construit pour un requêteur, SAML avec son profil ECP, et Liberty Alliance via avec un client dit « avancé » au sein de ses spécifications ID-WSF2.0. L'avantage de ce type d'architecture est de permettre à l'utilisateur d'être fournisseur d'identités ou d'attributs, mais également, d'avoir une meilleure appréhension de ses multiples identités numériques. Cette tendance est un point de convergence des architectures de fédération.

Cependant, bien que l'interopérabilité des systèmes de gestion d'identités soit le coeur des architectures de fédération, l'interopérabilité entre architectures de fédération hétérogènes est loin d'être acquise.

Nos travaux à venir, dont la base est illustrée dans cet article, seront menés plus avant afin de spécifier plus précisément l'interopérabilité entre SAML et WS-Federation.

Bibliographie

[1] Security Assertion Markup Language (SAML) V2.0 Technical Overview, Working Draft 10, 9 octobre 2006. OASIS Standard Specification.

[2] Web Services Federation Language (WS-Federation), version 1.1, Décembre 2006. BEA, BMC Software, CA Inc., IBM, Layer 7 Technologies, Microsoft, Novell Inc., VeriSign.

[3] Understanding WS-Federation, 28 mai 2007, version 1.0. IBM, Microsoft.

[4] WS-Trust 1.3, 19 mars 2007. OASIS Standard Specification.

[5] Web Services Security: SOAP Message Security 1.1 (WS-Security 2004), 1er fevrier 2006. OASIS Standard Specification.

[6] Liberty ID-FF Architecture Overview, version 1.2-errata-v1.0. Liberty Alliance.

[7] Web Services Metadata Exchange (WS-MetadataExchange), version 1.1, août 2006. BEA, CA Inc., IBM, Microsoft,SUN Microsystems, SAP, WebMethods.

[8] Web Services Security Policy Language (WS-SecurityPolicy), version 1.1 , Juillet 2005. IBM, Microsoft, VeriSign, RSA Security.

[9] Web Single Sign On Metadata Exchange Protocol, Avril 2005. Microsoft, SUN Microsystems.

[10] Web Single Sign On Interoperability Profile, Avril 2005. Microsoft, SUN Microsystems.

[11] Achieving Interoperability between Active Directory Federation Services and Shibboleth, Fevrier 2007. Oxford Computer Group.

[12] Passive Requestor Federation Interop Scenario, Version 0.4, 20 Fevrier 2004. Microsoft, IBM.

[13] Cross Operation of Single Sign-On, Federation and Identity Web Services Frameworks, version 1.1. Sampo Kellomäki, Symlabs,Inc.