Les protocoles d'authentification
(SAML et OAuth2)

Dans ce document, je vais tenter d'expliquer un peu plus clairement et simplement les protocoles SAML et OAuth2 que ce qu'on retrouve sur les Internets.

OAuth2 n'est pas un simple protocole d'authentification et j'espère pouvoir vous en convaincre plus bas, mais je le rajoute dans ce document parce que je n'ai pas envie de faire deux documents séparés. De toute façon, la plupart des logiciels récents capables de faire du OAuth2 sont aussi capables de faire du SAML.

1- SAML

SAML, c'est Security Assertion Markup Language. C'est un protocole d'échange d'informations liées à la sécurité et plus spécifiquement à l'authentification, basé sur XML.

Il existe plusieurs versions du protocole dont 1.0, 1.1 et 2.0. Le développement de SAML est intrinsèquement lié à Shibboleth.

Il s'agit essentiellement de tickets/jetons de sécurité en format XML échangés entre l'autorité d'émission (SAML Authority) appelé Identity Provider (IDP, IdP, fournisseur d'identités), et le consommateur du jeton (SAML consumer) appelé Service Provider (SP, fournisseur de service), parce qu'il fournit un (ou des) service (en échange du ticket consommé).

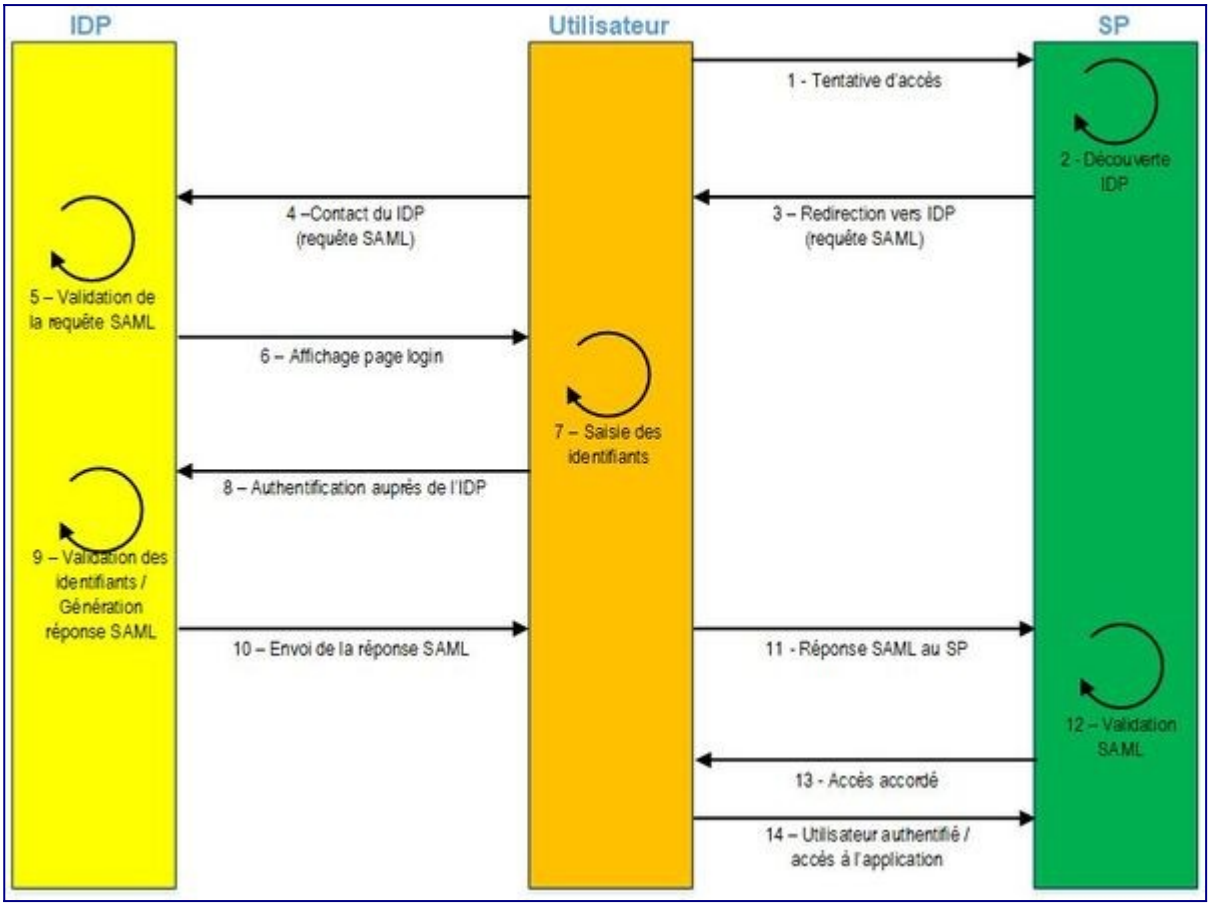
- **Très important à noter et à retenir** :

1. Le IDP et le SP ne communiquent pas directement entre eux, mais par l'intermédiaire du client.
2. C'est le client qui récupère le token du IDP, et qui le présente au SP, à qui revient la charge de le valider.
3. Le ticket est signé (puis chiffré dans certains cas) numériquement de telle sorte que le SP puisse valider son intégrité et son authenticité.
4. Il y a donc un échange préalable (outbound) de certificats/clés de sécurité entre SP et IDP, soit directement entre eux, soit en étant membres de fédérations communes telles que CAF, EduGain, FER, ...

- En termes plus simples, SAML est un protocole d'authentification qui permet aux applications de déléguer leurs services d'authentification à une application d'authentification centralisée telle que Shibboleth.

Voyons le à travers le diagramme ci-dessous.

- **Diagramme de flux SAML :**



- **Quelques explications du diagramme :**

1. - Le client tente (via son navigateur) d'accéder à l'application sécurisée (ou service) du fournisseur de service (SP).
2. - Le fournisseur de service (SP) devra découvrir le IDP (fournisseur d'identités) correspondant à l'utilisateur en question. Cela peut se faire en proposant au client de choisir parmi une liste de IDP existants et déjà pré-configurés au niveau du SP ou bien parce qu'il vient avec une URL WAYFless.

3. - Le fournisseur de service redirige alors le client vers son IDP. Le fournisseur de service crée une requête SAML (SAML Authentication Request), et demande au client de la transmettre à l'IDP.
4. - Le client contacte alors son IDP et lui transmet la requête SAML.
5. - L'IDP vérifie et valide que la requête provient bien d'un SP avec qui il est en relation de confiance. Cette relation de confiance a préalablement été configurée, soit par échanges directs de certificats, soit en passant par une fédération. En cas d'échec de validation (pas de relation de confiance), le processus s'arrête.
6. - L'IDP affiche la page de login au client.
7. - Le client entre ses paramètres d'authentification (nom d'utilisateur, mot de passe, MFA) dans le formulaire présenté par l'IDP.
8. - Les paramètres d'authentification sont transmis à l'IDP.
9. - Le fournisseur d'identités IDP valide les informations d'authentification du client (requête LDAP, base de données, …), et dans le cas où l'authentification réussie, génère une réponse SAML (SAML Response) à transmettre au SP. La réponse SAML indique au SP que le client est bien authentifié avec l'heure d'authentification et contient un certain nombre d'attributs (ex : nom, prénom, affiliation, courriel, groupes, etc.).
10. - La réponse SAML est transmise au client.
11. - La réponse SAML arrive chez le fournisseur de service en ayant transité via le client.
12. - Le fournisseur de service valide la réponse SAML par rapport à la requête SAML qu'il avait précédemment émise. Il vérifie l'authenticité et l'intégrité de la réponse, puis examine les attributs pour déterminer les accès à accorder au client. Si tout est conforme à ses requis, il donne accès au client.
13. - Le client dépasse la page d'authentification.
14. - Le client authentifié a accès à l'application selon les attributs fournis.

2- OAuth2

OAuth2, c'est du **Open-standard Authorization version 2**.

En effet, on parle ici d'autorisation et pas que d'authentification.

C'est "à peu près la même chose" que SAML, sauf que les autorisations se font entre applications, plutôt qu'entre utilisateur et application.

En effet, OAuth2 est utilisé pour fournir l'accès d'API à d'autres services (sites web, API, applications, ...) pour le compte d'un usager.

On va donc un peu plus loin que la simple authentification.

En d'autres termes, on pourrait dire que OAuth consiste à donner à une application extérieure connectée (third-party app) la possibilité de faire des actions en votre nom dans une autre application (dans laquelle vous avez bien sûr un compte). On pourrait prendre l'exemple d'API LinkedIn qui permettrait à d'autres applications RH de récupérer votre profil LinkedIn en votre nom.

2.1- Les acteurs

1. Usager
2. Application (third-party) dans laquelle l'usager est connecté
3. API (qui a des données de l'usager) - fournisseur de services (SP)
4. Serveur d'autorisation (qui a les informations sur l'identité de l'usager branché à l'API) - comment l'API authentifie

Au nom de l'usager, la third-party app va tenter de manipuler (avoir accès, modifier, ...) une partie de ses données se trouvant chez le fournisseur (SP) via une API dont l'authentification est assurée par IDP (serveur d'autorisation dans le cas OAuth2).

2.2- Diagramme de flux à haut niveau pour OAuth2

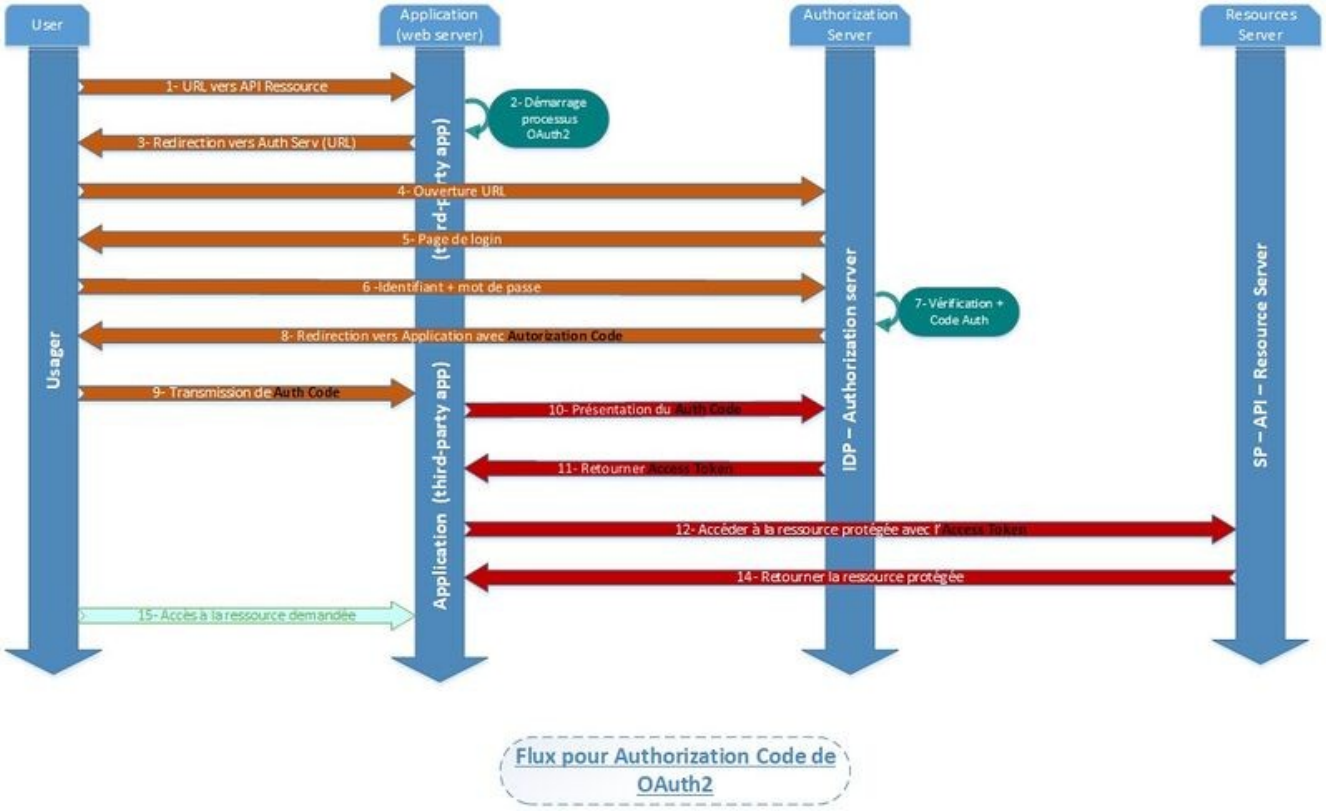
Diagramme de flux haut niveau pour OAuth2

Explications du diagramme haut niveau:

0. L'application doit préalablement être enregistrée avec le service en fournissant l'URL de redirection, le client ID, et le client Secret.

1. L'application demande à l'usager (le propriétaire de la ressource) l'autorisation d'accéder à une ressource sur le serveur de ressource en son nom (via une **Authorization Request**).
2. L'usager autorise l'application qui reçoit une **Authorization Grant**, si non, on s'arrête là.
3. L'application présente cette autorisation (**Authorization Grant**) au serveur d'autorisation pour demander un accès.
4. Le serveur d'autorisation reconnaît l'identité de l'application et valide le **Authorization Grant**, puis délivre à l'application un **Access Token**.
5. L'application présente l'**Access Token** au serveur de ressources pour demander l'accès à la ressource.
6. Si tout est conforme, le serveur de ressources fournit la ressource (**Protected Resource**) à l'application.

2.3- Diagramme de flux détaillé pour OAuth2



Flux pour Authorization Code de OAuth2

Explications du diagramme

- Les étapes 1 à 8 (en orange) concernent l'obtention du code d'autorisation (Authorization Code) qui va permettre à l'application de parler au nom de l'usager.

1. L'usager clique sur un lien dans la page web qui est en fait une demande de ressource distante au serveur de ressource.
2. Le serveur web détecte qu'il doit adresser une demande au serveur de ressources donc démarre le processus OAuth2.
3. Une URL de redirection vers le serveur d'autorisation est présentée à l'usager.
4. L'usager ouvre l'URL qui le redirige vers le serveur d'autorisation.
5. Le serveur d'autorisation présente une page de login à l'usager.
6. L'usager entre ses paramètres d'accès.
7. Le serveur d'autorisation vérifie les paramètres entrés et génère un code d'autorisation.
8. L'usager reçoit du serveur d'autorisation un code d'autorisation qui va être transmis à l'application (serveur web).

9. L'usager fournit le code d'autorisation à l'application, ce qui confirme qu'il a donné son accord pour l'accès à ses données.

- Les étapes 10 à 14 (en rouge) permettent à l'application, munie de l'autorisation de l'usager (**Authorization Code**) d'accéder à la ressource protégée.

10. L'application présente le code d'autorisation au serveur d'autorisation ainsi que d'autres paramètres tels que le **client ID** et **client Secret**.
11. Le serveur d'autorisation valide le code d'autorisation et lui retourne un code d'accès (**Access Token**).
12. L'application est maintenant autorisée à accéder à la ressource de l'utilisateur qui se trouve sur le serveur de ressources.
13. Le serveur de ressource retourne les ressources (données) demandées.

3- Quelques serveurs d'authentification centralisée

Shibboleth : ne fait que du SAML + SSO

CAS : fait du CAS, SAML, OAuth2, OIDC, MFA, SSO, …

KeyCloak : SAML, OAuth2, OIDC, MFA, SSO, …

Gluu : SAML, OAuth2, OIDC, MFA, SSO, …

Sources :

- https://oauth.net/2/
- https://tools.ietf.org/html/rfc6749
- https://en.wikipedia.org/wiki/OAuth
- https://www.mediawiki.org/wiki/Help:OAuth
- https://www.digitalocean.com/community/tutorials/an-introduction-to-oauth-2
- https://docs.oracle.com/cd/E39820_01/doc.11121/gateway_docs/content/oauth_flows.html
- https://fr.wikipedia.org/wiki/OpenID_Connect
- http://www.bubblecode.net/fr/2016/01/22/comprendre-oauth2/
- https://www.csoonline.com/article/3216404/what-is-oauth-how-the-open-authorization-framework-works.html
- https://en.wikipedia.org/wiki/SAML_2.0
- https://www.varonis.com/blog/what-is-saml/
- https://support.google.com/a/answer/6262987?hl=fr
- https://developers.onelogin.com/saml