AGENCE
DU NUMÉRIQUE
EN SANTÉ

Volet Transport Synchrone API Rest

CI-SIS – Juin 2023

Statut : Validé | *Classification : Publique* | *Version : 1.1*

Agence du Numérique en Santé
9, rue Georges Pitard – 75015 Paris
T. 01 58 45 32 50
esante.gouv.fr

1. TABLE DES MATIERES

INTRODUCTION

Ce document présente les modalités de sécurisation des API REST afin de garantir l'interopérabilité des systèmes d'information de santé en France. Il concerne la couche de transport et est indépendant des cas d'usages fonctionnels traités par ces API.

L'interopérabilité des Systèmes d'Information (SI) est un enjeu majeur pour les services de la e-santé en France. L'objectif de l'interopérabilité est de faciliter les échanges entre les différents SI tout en garantissant la bonne sécurisation des données partagées entre ces SI.

Ce volet du Cadre d'Interopérabilité des Systèmes d'Information (CI-SIS) prend en compte les éléments suivants :

- Les exigences de sécurité les plus élevées, en particulier celles qui concernent la protection des access token et des informations sensibles.
- La conformité aux normes d'architecture technique et la conformité aux exigences de la Politique Générale de Sécurité des Systèmes d'Information de Santé (PGSSI-S).
- Les spécificités sectorielles de la e-santé en France, telles que les identifiants sectoriels, Pro Santé Connect ou encore IGC Santé.
- Les standards techniques de sécurité actuels, tels que OAuth 2.0, OpenID Connect, l'API REST, etc.

Dans cette vision, le cadre d'interopérabilité doit permettre de garantir l'intégration de ces nouvelles technologies. Cela passe notamment par la mise en place de protocoles d'authentification, de contrôle d'accès et de gestion des identités.
À noter que les technologies existantes, telles que le SAMLV2, l'API SOAP, etc. ne sont plus préconisées en cible. Elles sont toutefois utilisées dans l'architecture provisoire Services cibles existants utilisant une assertion SAML.
Ces technologies sont documentées dans les volets transports suivants : CI-SIS Volet Transport Synchrone Client Lourd et CI AMO Volet Transport Synchrone pour les services de l'Assurance Maladie [1].

Ce document est destiné à un public disposant de compétences techniques avancées.
Une compréhension des standards OAuth 2.0 et OpenID Connect est notamment recommandée pour la lecture de ce document.

2. LE POSITIONNEMENT DU CI-SIS VOLET TRANSPORT REST DANS L'EXISTANT DOCUMENTAIRE

Ce volet spécifie la couche Transport qui identifie les standards de transport des données et qui permet la communication des éléments d'identification et d'authentification des utilisateurs, physiques ou moraux, nécessaires à la mise en place des contrôles d'accès.

Il spécifie la couche Transport pour :

- **Un système cible** offrant un service auquel il est possible de se connecter de façon synchrone.
- **Un système initiateur** (application Web, client lourd, client natif) se connectant au service de façon synchrone.

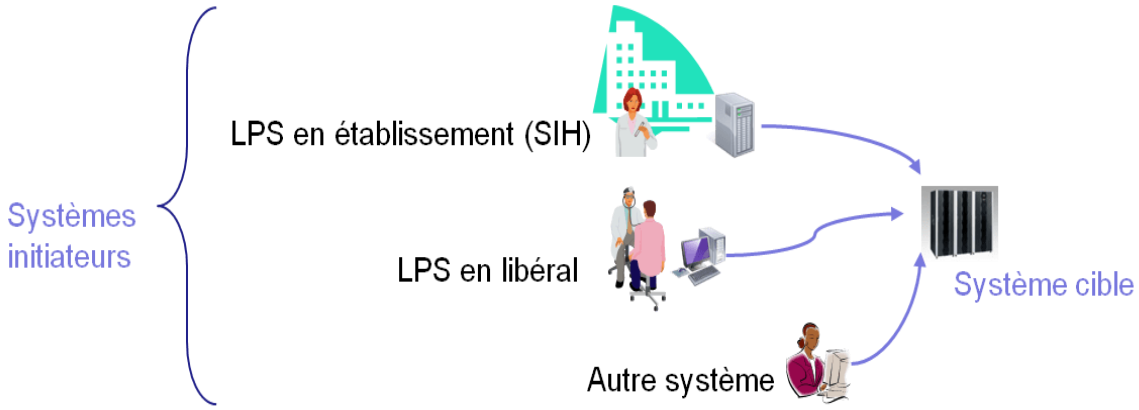

Figure 1 : Rôle des systèmes

Le ***Cadre d'Interopérabilité des Systèmes d'Information de Santé*** **(CI-SIS)** établit les règles pour une informatique communicative dans le domaine de la santé, du médico-social et du social. Le CI-SIS préconise des **normes techniques** et **sémantiques** pour les acteurs de la santé impliqués dans des projets d'échange et de partage de données de santé.

La mise en place des volets du cadre d'interopérabilité permet de minimiser les coûts d'intégration de nouvelles interfaces et les difficultés techniques d'interopérabilité, de même que de faciliter l'utilisation de produits d'éditeurs.

Les SI de santé en France sont composés de plusieurs entités indépendantes proposant un certain nombre d'applications et de services de nature hétérogène (client léger/web, client lourd, client applicatif). Les applications des différents Fournisseurs de Services (FS) sont susceptibles de communiquer entre elles.

Afin de faciliter les échanges et d'assurer l'intégrité des données transmises d'un SI à l'autre, il est indispensable de standardiser les échanges. C'est ainsi qu'intervient le Cadre d'Interopérabilité des SI de Santé (CI-SIS) pour résoudre les problèmes d'interopérabilité des applications de santé.

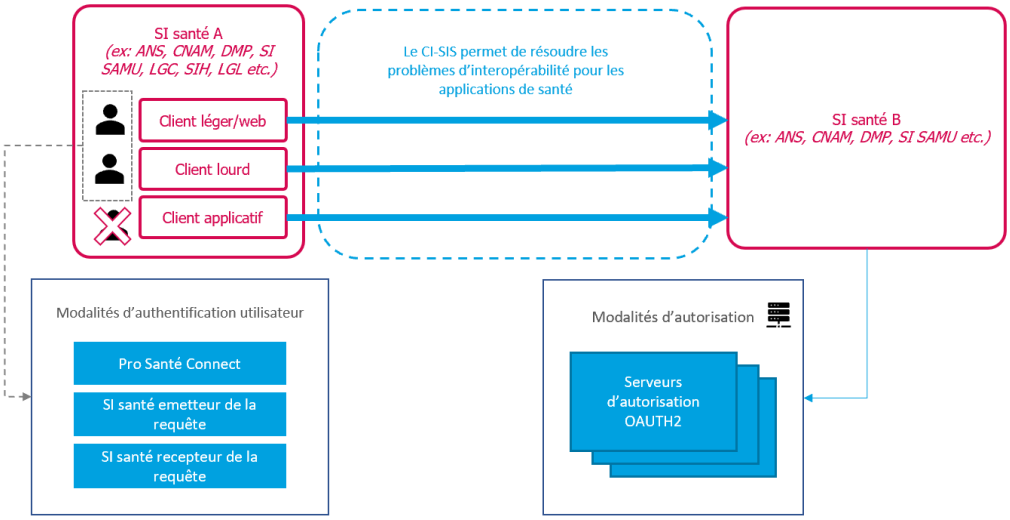

Figure 2 : Les composants en jeu dans le CI-SIS Volet Transport REST

Par ailleurs, ce volet du CI-SIS s'insère dans l'ensemble documentaire décrit ci-dessous :

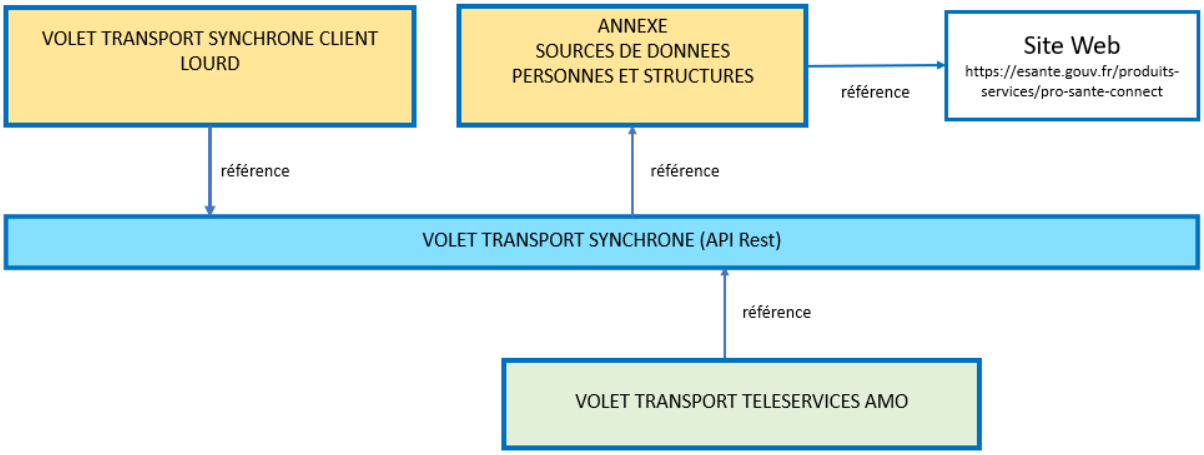

Figure 3 : Ensemble documentaire CI-SIS Volet Transport

En dehors du volet transport du CI-SIS, il existe des référentiels d'interopérabilité à la maille :

- **Sémantique** : décrivent les normes et terminologies standardisées et partagées qui doivent être utilisées pour décrire différentes catégories de données (diagnostics, médicaments, examens de biologie médicale, etc.). [2]

- **Syntaxique** : l'interopérabilité syntaxique permet à deux ou plusieurs systèmes de communiquer et d'échanger des données. Pour être efficace, la conception de toute solution d'interopérabilité doit tenir compte de la granularité des données à partager.

Afin de sécuriser les échanges entre les SI de santé, le présent volet du CI-SIS définit les modalités d'autorisation d'accès à un service de santé et se base sur les protocoles OAuth 2.0 et OIDC. Les normes d'interopérabilité décrite dans le présent volet sont applicables aux applications **Pro Santé Connectées** et **non Pro Santé Connectées.**

- Une application **Pro Santé Connectée** est une application qui permet aux professionnels de santé de se connecter à un réseau d'échange d'informations de santé, afin de faciliter la gestion des informations médicales via les MIE PSC compatibles.
- Une application **non Pro Santé Connectée** est une application qui ne nécessite pas d'utiliser Pro Santé Connect pour accéder aux informations de santé.

Descriptif des acteurs et des composants du cadre

La PGSSI-S distingue :

- **Les personnes physiques** : acteurs des secteurs sanitaire, médico-social et social et les usagers des SI santé.
  - Les usagers sont identifiés par :
    - **L'identité INS** (Identité Nationale de Santé).
    - **L'identité locale fournie** par un fournisseur de service.
  - Les acteurs des secteurs sanitaire, médico-social et social sont identifiés par leur :
    - **Numéro RPPS**, (Répertoire Partagé des Professionnels de Santé) à utiliser en priorité s'il existe pour la personne à identifier : utilisé dans les cartes CPx et e-CPS.
    - **Numéros locaux** composés d'identifiants privés à portée nationale (ex : identifiants portés dans le VIHF).

N.B. (en cours de migration) **: Le numéro ADELI** (Automatisation DEs LIstes) est toléré de façon transitoire jusqu'à son remplacement définitif par l'identifiant **RPPS.**

- **Les personnes morales :** toutes les structures de soins, telles que les maisons et centres de santé, mais aussi les laboratoires de biologie médicale, les services de protection maternelle infantile ou encore les services de santé au travail. Ces dernières sont identifiées par :
  - Le numéro **FINESS** juridique (FINESS EJ) ou géographique (FINESS ET ou EG).
  - Le numéro **SIREN** ou **SIRET.**

Pour toute information sur les référentiels d'identification des personnes physiques ou morales, se référer au corpus documentaire [3]

Les systèmes en jeu dans le cadre de ce volet du CI-SIS sont les suivants :

Composant	Typologie de client
Système initiateur (client)	○ **Client lourd** : logiciel d'un professionnel de santé installé sur le poste de travail. Il peut être chez un libéral ou bien un établissement de santé. ○ **Application mobile ou native** utilisée par un PS ○ **Application web** (client léger) : application disponible sur un navigateur web utilisable par un PS ○ **Client applicatif** : serveur applicatif appartenant à une structure identifiée (la personne morale). Ce client utilise une API proposée par le système cible
Système cible	○ **Serveur applicatif** : serveur applicatif appartenant à une structure identifiée (la personne morale). Ce serveur expose une API
Fournisseur d'identité (Identity Provider ou IDP)	○ **Pro Santé Connect (PSC)** : Fournisseur d'identité permettant aux PS de s'authentifier via les MIE PSC compatibles ○ **SI local** : fournisseur d'identité propre à un service (applications non pro-santé connectée)

3. LES PREREQUIS A L'IMPLEMENTATION

Pour être conformes au présent volet, les systèmes initiateurs et les systèmes cibles doivent pouvoir s'appuyer sur des certificats émis par une **IGC Santé** autorisée par les **référentiels d'authentification de la PGSSI [3]** associés à des personnes physiques (ex. carte CPS nominative) ou à des personnes morales (ex. certificat de personne morale, certificat serveur).

Est détaillé dans ce présent volet transport les différents standards nécessaires à l'interopérabilité, à savoir :

- **Échanges synchrones via API REST en format JSON.**
- **Protocole OAuth 2.0** pour la sécurisation des échanges et les contrôles d'accès.
- **Protocole OIDC** pour la propagation de l'identité de l'utilisateur.
- **Certificat en TLS mutuel (mTLS**) permettant d'établir une connexion sécurisée entre deux parties, généralement un client et un serveur.
- **Protocole CIBA avec Pro Santé Connect :** Flux d'authentification découplé, qui permet d'éviter la redirection en décorrélant l'appareil d'utilisation d'une application et l'appareil utilisé pour s'authentifier. L'utilisation du flux CIBA est particulièrement adaptée pour les clients lourds en authentification locale.

4. GLOSSAIRES

Tableau 1 : glossaire général

Terminologie	Description
ADELI	Automatisation Des Listes. Répertoire recensant, à l'échelle nationale, les professionnels de santé dont les professions sont réglementées par le Code de la Santé Publique. A termes, ADELI serait remplacé par le RPPS.
ANSSI	Agence Nationale de la Sécurité des Systèmes d'Information
API	API (application programming interface ou « interface de programmation d'application ») est une interface logicielle qui permet de « connecter » un logiciel ou un service à un autre logiciel ou service afin d'échanger des données et des fonctionnalités
API Publique	Une API est appelée publique lorsque l'appel des ressources se fait via clé API. L'API publique est accessible à tous les utilisateurs. Elle est généralement utilisée pour fournir des services ou des fonctionnalités à des tiers.
API Privée	L'API privée est accessible uniquement aux personnes autorisées, généralement les membres d'une organisation spécifique. Elle est utilisée pour des échanges de données internes ou pour offrir des services spécifiques à des clients sélectionnés.
API REST	Representational State Transfer Interface de programmation qui suit les principes de l'architecture REST.
FINESS géographique	Fichier National des Etablissements Sanitaires et Sociaux. Assure l'immatriculation des établissements géographiques (ET) correspond à une implantation géographique et est caractérisé par une et une seule catégorie d'établissement.
FINESS juridique	Une entité juridique (EJ) est une personne morale juridiquement responsable des activités réalisées auprès du public pris en charge. Elle est identifiée par son numéro FINESS EJ.
HTTP	Hypertext Transfer Protocol. Protocole d'échange standard pour le Web et pour les API REST ou SOAP
HTTPS	Le protocole HTTPS (Hyper Text Transfer Protocol Secure) est une extension sécurisée du protocole HTTP, le « S » pour « Secured »
IDP	IDentity Provider ou Fournisseur d'identité.
INS	Identité National de Santé.
MIE PSC	Moyen d'Identification Electronique Pro Santé Connect.
RPPS	Répertoire Partagé des Professionnels de Santé.
OAuth 2.0	Open Authorization Protocole de sécurité mis en place pour permettre d'accéder aux ressources hébergées (voir la section 5.2 pour davantage de précision).
OIDC	OpenID Connect (voir la section 5.3 pour davantage de précision)
PGSSI-S	Politique Générale de Sécurité des Systèmes d'Information de Santé Référentiels et documents fixant les exigences relatives aux différents aspects de la sécurité des systèmes d'information en santé.
SIREN	Système d'Identification du Répertoire des ENtreprises Numéro qui permet d'identifier une entreprise auprès des administrations.
SIRET	Système d'Identification du Répertoire des ETablissements Numéro qui permet d'identifier un établissement.

Tableau 2 : glossaire des acteurs

Acteurs	Description
Client Lourd = Thick client	Logiciel Professionnel de santé (LPS) installé sur le poste de travail de l'utilisateur.
Device d'authentification	Appareil du professionnel de santé utilisé lors du protocole CIBA pour l'authentifier (soit téléphone portable + e-CPS, soit carte CPx).
Fournisseur de services = Service Provider	Serveur d'application en charge de réaliser les requêtes auprès du fournisseur d'identité, du serveur d'autorisation et du service cible.
Navigateur = Single Page Application	Navigateur web sur lequel l'utilisateur initie les requêtes d'accès aux données du service cible.
Pro Santé Connect	Fédérateur de fournisseurs d'identité au standard OpenID. L'authentification se fait par l'utilisation d'un MIE PSC compatible, actuellement les cartes e-CPS ou CPS (carte physique).
Proxy LPS API*	Serveur d'application qui joue le rôle de passerelle, tiers de confiance, dans le cas d'une authentification d'un client lourd.
Proxy LPS FS*	Serveur d'application dédié aux étapes d'authentification lors du protocole CIBA dans le cas d'une authentification d'un client lourd. Il est nécessaire pour l'intégration de Pro Santé Connect.
Serveur d'autorisation OAtuh 2.0 = Authorization server	Serveur d'autorisation qui effectue les contrôles d'accès via le protocole OAuth 2.0. Le serveur d'autorisation est porté par le service cible qui a la charge de protéger ses données.
Service cible	Service offrant les données dont l'utilisateur a besoin et auquel il est possible de se connecter de façon synchrone.
Service initiateur	Application web ou client lourd se connectant au service cible de façon synchrone.

*Ces deux proxys doivent l'objet d'une mutualisation : voir Définition des proxys plus bas.

Tableau 3 : glossaire des paramètres

Paramètres	Description
`access_token`	Jeton d'accès délivré par le serveur d'autorisation OAuth 2.0 donnant accès aux ressources du service cible.
`actor_token`	Paramètre dans la requête du fournisseur de services auprès du serveur d'autorisation. Il s'agit d'un champ permettant d'ajouter des données additionnelles utiles à l'authentification (ex : identifier le fournisseur de service).
`Client_ID_AS`	Client ID attribué au fournisseur de services par le serveur d'autorisation OAuth2.0 du service cible. Il est utilisé lors de la requête d'un `access_token` pour accéder au système cible.
`Client_ID_FS`	Client ID attribué au fournisseur de services par le fournisseur d'identité. Le `Client_ID_FS` est lié à l'OU dans le certificat TLS.
`scope`	Paramètre dans la requête du fournisseur de services auprès du serveur d'autorisation afin de déterminer les scopes (périmètres) métiers demandés.
`Structure_ID`	Identifiant de la structure du système initiateur utilisé dans le cadre d'une authentification nécessitant un mTLS.
`subject_token`	Token passé en paramètre dans la requête du fournisseur de services auprès du serveur d'autorisation afin d'authentifier l'utilisateur qui souhaite accéder aux ressources. **Dans le cas des API ProSantéConnectées :** **`subject_token` = Access Token PSC**

5. LES STANDARDS D'ECHANGE (REST, OAUTH 2.0 ET OIDC)

5.1. Les API REST

5.1.1. Généralités sur le concept d'API REST et bonnes pratiques

Une **API REST** est une interface de programmation qui suit les **principes de l'architecture REST**.
Elle permet aux clients (des applications web ou mobiles) de communiquer avec un serveur en utilisant des **requêtes HTTP**. Les ressources cibles sont identifiées par des URLs.
Dans une API REST, les opérations **CRUD** (Create, Read, Update, Delete) sont implémentées via les méthodes dans une requête : POST, GET, PUT, et DELETE respectivement.
Les réponses à ces requêtes sont retournées en format JSON.

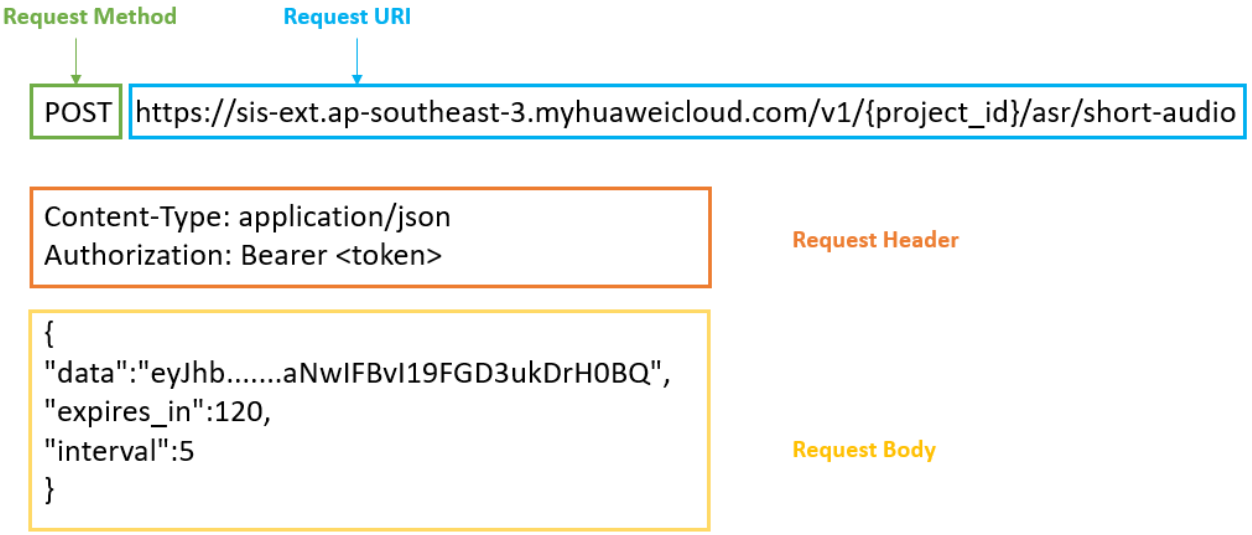

Figure 4 : exemple de requête d'une API REST avec la méthode POST

5.1.2. Liaison avec le protocole de transport ou binding

Le protocole **HTTP1.1** encapsulé dans une connexion sécurisée **TLS** doit être utilisé.

5.2. Le standard OpenID Connect pour la gestion de l'authentification des applications Pro Santé Connectées

Pro Santé Connect est le fournisseur d'identité de la santé pour les acteurs de santé professionnels en France. Il s'agit d'un service basé sur le protocole standard OpenID Connect. Il permet aux professionnels de santé de s'identifier de manière simple, sécurisée et unifiée. Ils se connectent aux services numériques en santé, en passant de l'un à l'autre de manière fluide.

Une application Pro Santé Connectée est une **application web** ou **client lourd** qui permet aux professionnels (PS) de s'authentifier à l'aide d'un **MIE PSC compatible**. Ces applications sont généralement utilisées par des établissements de santé/e-santé pour la gestion de leurs dossiers médicaux.-

Pour accéder au service d'authentification de PSC permettant d'accéder à des API Pro Santé connectées, deux cinématiques sont disponibles :

- Via le flux **authorization code [4]**
- Via le flux **Client Initiated Backchannel Authentication (CIBA) [5]**

Description du protocole OpenID Connect

OpenID Connect (OIDC) est un protocole d'authentification qui s'appuie sur OAuth 2.0 pour fournir une identification sécurisée et basée sur des jetons pour les applications Web et mobiles.

OIDC fournit une couche d'authentification supplémentaire au-dessus d'OAuth 2.0, qui permet aux clients d'obtenir des informations d'identification et de profil sur les utilisateurs en utilisant des **ID tokens**.

Ce protocole utilise des jetons **JSON Web Tokens (JWT)** pour échanger des informations d'identification entre l'application cliente, le fournisseur d'identité et le service/système cible.

La **documentation technique Pro Santé Connecté [4]** fournit des détails sur le fonctionnement du protocole OIDC au sein de PSC.

5.3. Le standard OAuth 2.0 pour la gestion des autorisations et accès

OAuth 2.0 est un **protocole standard d'autorisation** qui permet à une application tierce **d'accéder à des ressources protégées**. Le protocole OAuth 2.0 définit des étapes à suivre pour obtenir et utiliser un **jeton d'accès** (`access_token`). Ce jeton est délivré par le serveur d'autorisation et permet d'accéder aux ressources d'une application protégée. Les services de e-santé utilisent ce standard OAuth 2.0 pour la gestion des autorisations et des accès.

5.3.1. Description du serveur d'autorisation du système cible

OAuth 2.0 est un **protocole d'autorisation** qui permet à une application (le client) de se connecter aux ressources protégées sur service cible en utilisant un `access_token` émis par un **serveur d'autorisation**. Ce dernier est **responsable du contrôle d'accès du client**, de **l'émission et de la gestion de ces jetons d'accès** et de la **vérification du jeton d'accès**.

D'autre part, le serveur d'autorisation va effectuer le **contrôle d'accès sur les scopes** de l'application appelante à l'aide de l'identifiant du client (`Client_ID_AS`) et des scopes associés dans la requête.
À l'issue du contrôle d'accès, le serveur d'autorisation va délivrer un jeton d'accès (`access_token`) permettant d'accéder aux ressources du système cible.

Le serveur d'autorisation est un composant essentiel du flux OAuth 2.0 qui permet de garantir la sécurité et la confidentialité des données de l'utilisateur et doit être implémenté côté **service/système cible**.

La règle générale d'autorisation portera toujours sur une structure et son service. Une structure est soit une structure de santé ou une structure autorisée (proxy éditeur).

5.3.2. Gestion et contrôle des accès

5.3.2.1. Le contrôle d'accès avec les scopes

- **Définition d'un scope dans le protocole OAuth 2.0**

Dans le protocole OAuth 2.0, les scopes sont utilisés par **le client** pour demander un **accès limité aux données du propriétaire** de la ressource.

Les scopes sont utilisés pour définir les autorisations que le client demande auprès du service cible propriétaire de la ressource. Ils décrivent les actions que l'application sera autorisée à effectuer ou le périmètre de données sur lequel l'application sera autorisée d'agir.

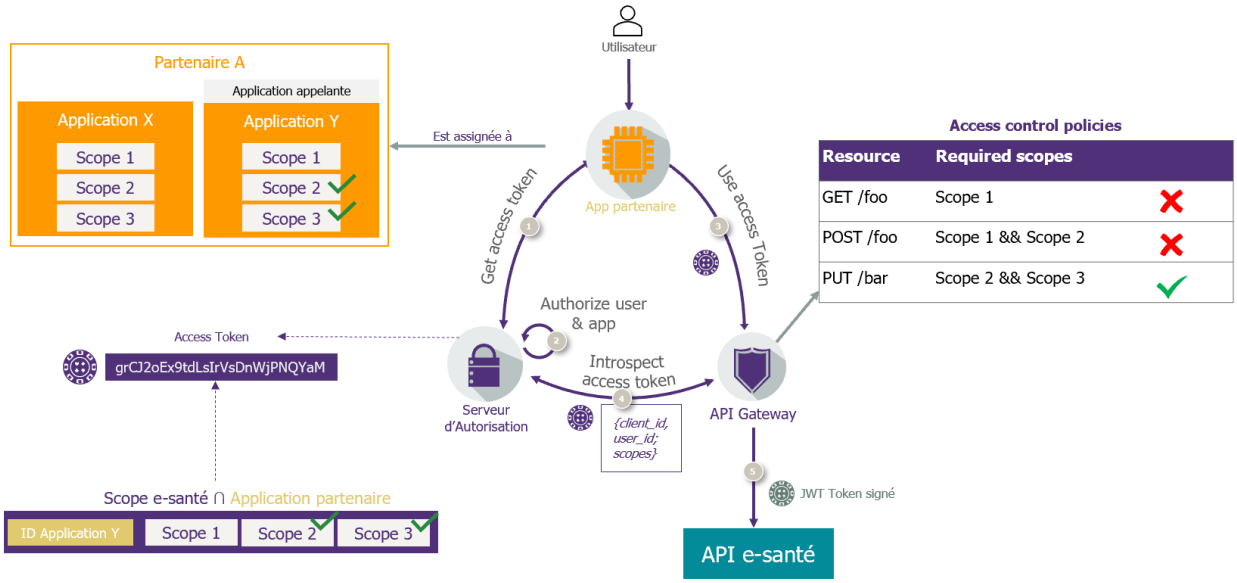

Figure 5 : Gestion des scopes avec OAuth 2.0

Afin d'illustrer l'usage du scope au sein d'un processus fonctionnel, on peut prendre **l'exemple simplifié** d'un utilisateur qui souhaite **accéder aux ressources protégées d'un service de e-santé.**

Dans le cas des API Pro Santé Connectées, le Professionnel de Santé (PS) est redirigé vers **PRO Santé CONNECT** par le fournisseur de service. Une fois le PS authentifié sur PSC, celui-ci envoie les **tokens PSC** au fournisseur de services.
L'authentification du fournisseur de services auprès du serveur d'autorisation se fait selon la RFC 7235 [11] c'est-à-dire avec un `Client_ID_AS` dans le header de sa requête suivant la méthode « Authentification Basic » qui encode les credentials en base 64.
Le fournisseur de service envoie une requête auprès du serveur d'autorisation qui contient comme paramètres, un `subject_token` (**dans le cas des API ProSantéConnectées :** `subject_token = Access Token PSC`), un **certificat de structure** et des **scopes métier**.
Une fois le contrôle d'accès est réalisé par le serveur d'autorisation, le PS peut accéder au service de e-santé.

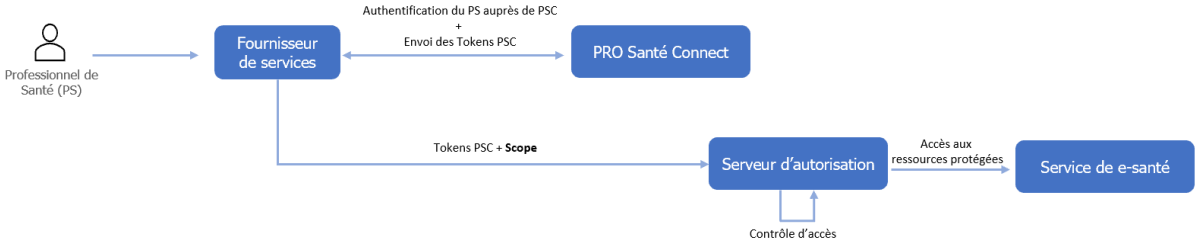

Figure 6 : Schéma d'exemple cas métier de l'usage du scope

5.3.2.2. Les normes de conservation de l'access token

La sécurité liée au stockage de l'access token et à sa conservation est définie par l'ANSSI dans le document **Recommandations pour la sécurisation de la mise en œuvre du protocole OpenID Connect dans la section 3.4 Recommandation R23 à R26. [6]**

Recommandation :

Une fois envoyé, les attributs de validité de l'`access_token` pourront être conservés par le système cible, notamment au sein d'un cache au niveau du serveur d'autorisation.
Afin de résoudre le problème de l'éventuelle révocation de l'`access_token`, la recommandation est de **définir la durée de validité de l'`access_token`**, émis par le serveur d'autorisation, entre **1h** et **4h**. Il est également fortement recommandé de vérifier la validité de l'`access_token` toutes les demi-heures en utilisant le service d'introspection du serveur d'autorisation.
Si l'`access_token` est un jeton autoporteur JWS l'introspection n'est pas nécessaire, mais si c'est un jeton JWT opaque, elle l'est.

5.3.2.3. Les données de l'access token

Les données de l'access token peuvent être accessibles soit dans un jeton JWT signé par le serveur d'autorisation, soit via un appel à un service d'introspection.

Les paramètres de l'access token sont définis par les champs suivants :

Champs	Description
iss	Identifiant de l'émetteur du jeton
sub	Identifiant de l'utilisateur pour lequel le jeton a été émis
aud	Identifiant ou liste d'identifiants des ressources qui peuvent être accessibles avec ce jeton
nonce	Chaîne de caractères aléatoires unique qui peut être utilisée pour identifier les échanges entre le client et le serveur
exp	Heure d'expiration du jeton, après laquelle il ne sera plus valide. Il est de la responsabilité des services de déterminer la durée de vie de l'access token (une durée de vie pas trop courte est à privilégier)
iat	Heure à laquelle le jeton a été émis
scope	Définit le type et le périmètre des ressources octroyées par le serveur d'autorisation. Sur cette base, le serveur d'autorisation authentifie l'utilisateur et recueille son consentement pour la transmission des ressources

Exemple de contenu d'un access_token dont la durée de vie est d'une heure :

```
{
  "iss":
"https://server.example.com",
  "sub": "identifiant_national",
  "aud": [Ressource A, Ressource B],
  "nonce": "n-0S6_WzA2Mj",
  "exp": 1626921770,
  "iat": 1311280970,
  "scope": “scope 2 scope 3”,
}
```

5.3.2.4. L'intégration du mTLS dans le cadre du workflow OAuth 2.0

Dans le cadre d'un workflow OAuth 2.0, l'utilisation du mTLS permet :

- La sécurisation des échanges (chiffrement)
- L'authentification du client vis-à-vis du serveur
- L'authentification du serveur vis-à-vis du client

Ci-dessous une illustration de l'utilisation du mTLS avec OAuth 2.0.

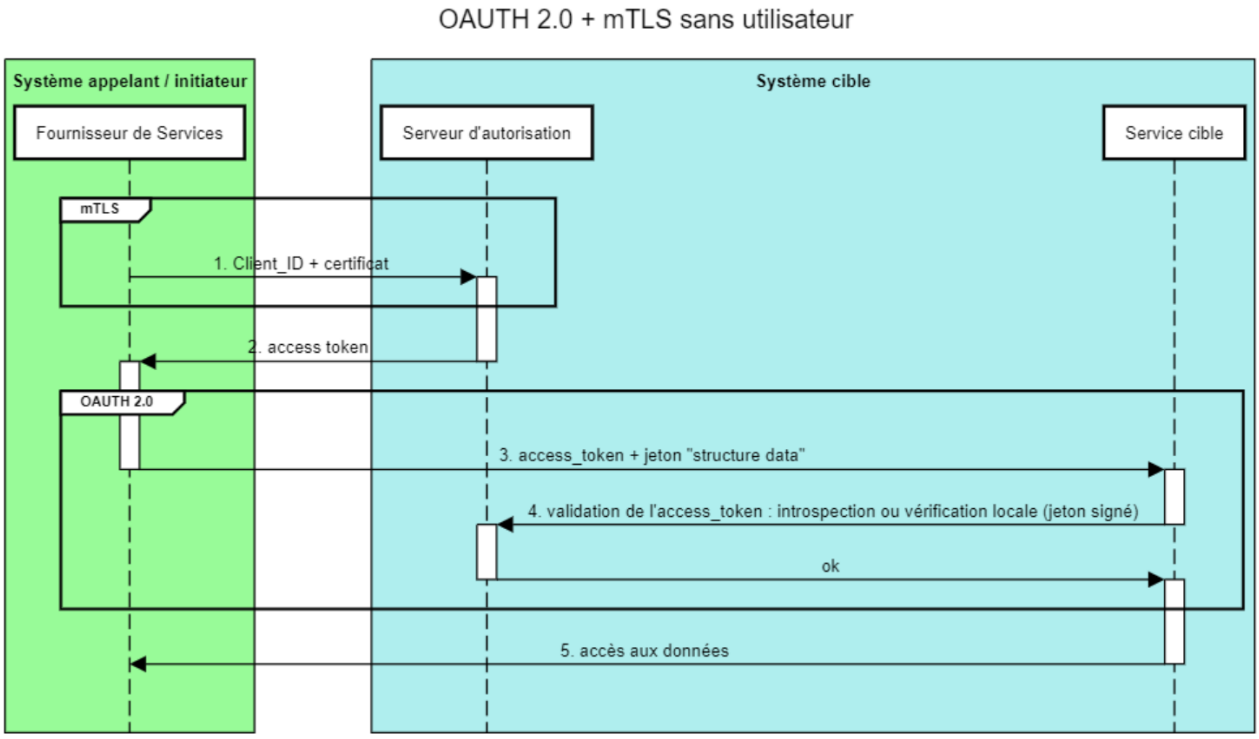

Figure 7 : Schéma TLS mutuel avec OAuth 2.0

Le détail des flux OAUH 2.0 + mTLS est défini dans la partie 7.6.3 Cas OAuth 2.0 + mTLS : accès à des données sensibles

Le protocole TLS est assuré sur la couche transport. Le certificat client utilisé dans le mTLS peut être transmis au serveur d'autorisation. Le protocole OAUTH est réalisé sur la couche applicative et permet de faire le contrôle du certificat client et de générer un access_token.

5.3.2.5. Le lien entre un certificat TLS (champ Organisational Unit), sa requête et la réponse de l'access token.

Rappel sur les certificats émis une IGC Santé :

Un **certificat de personne morale** (également appelé certificat d'entreprise) est un document électronique qui identifie une personne morale, telle qu'une entreprise, une organisation gouvernementale ou une association, auprès d'une autorité de certification (AC). Il sert à garantir l'authenticité et l'intégrité des communications électroniques émises par la personne morale, en fournissant une signature électronique associée à son identité.

Un **certificat serveur** (également appelé certificat SSL/TLS) est un document électronique qui identifie un serveur web auprès d'une autorité de certification (AC). Il est utilisé pour garantir l'authenticité et la confidentialité des communications entre un navigateur web et un serveur, en chiffrant les données échangées. Les certificats serveur sont utilisés pour sécuriser les connexions HTTPS, qui sont devenues la norme pour les sites web.

Recommandation :

- L'attribut OU du certificat issu de l'IGC Santé contient un identifiant de type `struct_IdNat` qui lui-même est constitué d'un préfixe suivi de l'identifiant SIRET ou FINESS.
- Pour une structure de santé, le FINESS pourra être le FINESS juridique.
  Par exemple, les hôpitaux et les EHPAD génèrent des certificats sur leur FINESS juridique

Pour plus de précisions sur le contenu du certificat, voir le document [7]

Pour plus de précisions sur le contenu du `struct_IdNat` voir la section 5.5 [8]

Si certaines structures souhaitent indiquer le FINESS géographique, elles le pourront, mais la recommandation reste le FINESS juridique.

6. LES DISPOSITIONS DE SECURITE SELON LES CAS D'USAGE

6.1. Confidentialité

Dans le volet transport synchrone pour application mobile : la confidentialité des échanges au niveau du transport est gérée par l'encapsulation du flux dans une connexion sécurisée TLS.
L'usage du TLS version 1.3 ou supérieure est recommandé et celui de la version 1.2 encore supporté. Les version 1.0 et 1.1 sont interdites. L'usage de SSL 3.0 ou inférieure est interdit.

6.2. Intégrité

L'intégrité des échanges au niveau du transport est gérée par l'encapsulation du flux dans une connexion sécurisée TLS.

6.3. La sécurisation de l'architecture selon les cas d'usages

6.3.1. Les niveaux de sensibilité des données

La PGSSI-S met en avant des référentiels d'identification électronique pour personnes physiques et morales définis dans les documents suivants [11] et [12].

Dans le cadre des **API PSC Connectées** (7.4 Cas d'usage #1 de ce présent document), les données échangées sont des **données sensibles** qui nécessitent une architecture sécurisée.

Dans le cadre des **API non PSC Connectées** (7.5 Cas d'usage #2) et du cas d'usage (7.6.2 authentification par OAuth 2.0) les données échangées sont des **données non-sensibles** et sont dites **restreintes**.

Dans le cadre d'une **authentification par mTLS** (Cas d'usage 7.6.4 et 7.6.5) et par **mTLS + OAuth 2.0** (7.6.3), les données échangées sont des **données sensibles** qui nécessitent une architecture sécurisée.

Le présent volet du CI-SIS a pour objectif de présenter les méthodes pour sécuriser les échanges de données **selon la sensibilité de la donnée échangée**.

6.3.2. La déclinaison des spécifications par cas d'usages et exigences de sécurité

L'objectif de la section présentée ci-dessous est de décrire les spécifications techniques permettant les échanges entre les différents SI Santé de façon sécurisée.

Les éléments permettant de déterminer le niveau de sensibilité de la donnée sont explicités dans la documentation de la PGSSI-S.

- **Si l'API appelée est publique :**

  L'appel aux ressources/services cibles se fait par clé API fournie par le service cible à la suite d'un enrôlement de l'application appelante. Le service cible n'exige pas d'authentification en plus de la part du système initiateur, car les données exposées sont publiques.

- **Si l'API appelée est privée :**

  - **Cas de l'authentification indirecte de l'utilisateur au sein d'une structure**

    On parle d'identification indirecte d'un professionnel de santé dans la mesure où le service cible s'appuie sur une authentification du professionnel réalisée localement par la structure.

    Dans le cas où seule la personne morale (structure) a besoin de s'authentifier, celle-ci est directe et réalisée par la connexion mTLS avec le certificat de structure.
    L'accès à l'API se fait avec OAuth 2.0 et/ou mTLS. Si les données sont sensibles, l'utilisation d'une connexion mTLS à minima est nécessaire.
    Le mTLS permet d'authentifier le client par un `Client_ID_AS` + mTLS au lieu de s'authentifier avec son `client_secret/Client_ID_AS`. Dans ce cas le `client_secret` n'est pas nécessaire car l'authentification est portée par le mTLS (certificat client).

    La structure appelante doit fournir les informations utilisateur selon les besoins / cas d'usage métier (ex : RPPS, profession, autres informations utilisateur).

  - **Cas d'authentification directe d'un professionnel de santé**

    - **Cas d'authentification avec Pro Santé Connect :**
      NB : Ce cas est décrit dans cette version du CI-SIS

      Dans le cas où l'utilisateur a besoin de s'authentifier avec **Pro Santé Connect** (pour accéder à des données sensibles), l'utilisateur s'authentifie avec son MIE PSC et l'accès à l'API du système cible se fait avec OAuth 2.0.

      - Soit le PS s'authentifie à un FS avec PSC qui fournit lui-même des données de santé.
      - Soit le PS accède à un service avec PSC, et doit accéder à un autre service fournisseur de données de santé par des API Pro Santé connectées.

    - **Cas d'authentification sans Pro Santé Connect :**

      Dans le cas où l'utilisateur a besoin de s'authentifier physiquement avec un fournisseur d'identité tiers au service cible, l'utilisateur s'authentifie selon les modalités spécifiques requises et l'accès à l'API du système cible se fait avec OAuth 2.0 avec PKCE.

Un serveur d'autorisation est mis à disposition à par la/les structure(s) cible(s)/appelée(s) dans le cadre de la réalisation du protocole OAuth 2.0.

6.3.3. La définition des proxys

Le **proxy** est un **serveur applicatif** mis à disposition par le **système initiateur** qui souhaite accéder aux données du **système cible**.

L'utilisation d'un **proxy est motivée par des enjeux de :**

Sécurisation :

- Par la gestion des credentials pour s'authentifier sur PSC et sur le serveur d'autorisation du service cible. Chaque proxy détiendra un `Client_ID_AS` et un certificat TLS client.
- Séparation de la gestion des credentials entre le module qui utilise l'access_token (Fournisseur des services pour le cas web et Proxy API/LPS pour le cas client lourd) et le module qui le demande (navigateur, mobile ou LPS). Le proxy empêche donc la remontée de ces informations au niveau de l'instance qui les demandent.
- Un contrôle d'accès d'un proxy au service cible réalisé grâce aux scopes

Découplage des requêtes au service cible et des requêtes vers PSC :

- L'access token PSC est utilisé une seule fois pour initier l'authentification sur le service cible lors de l'échange de jetons : `subject_token` (access_token PSC) contre un `access_token`
- La durée de validité de l'access_token API est de 1h contre 2 minutes pour l'access_token PSC.

Cela évite une sursollicitation de PSC lors de l'envoie des requêtes vers le service cible.

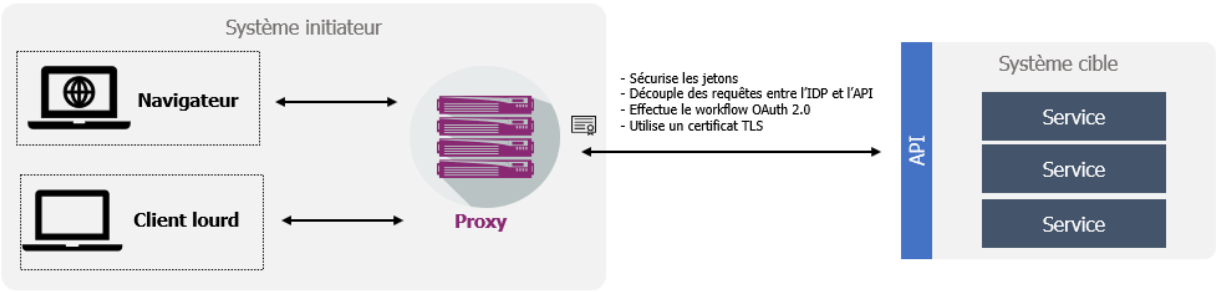

Figure 8 : Rôle du proxy

Un proxy est soumis aux exigences de sécurité suivantes :

- Sécurisation de la connexion entre chaque instance qui demande les tokens (Navigateur ou LPS) et l'instance qui l'utilise (FS pour le cas web et proxy LPS FS/API pour le cas client lourd).
- Fiabilité dans le mapping entre l'instance demandant les tokens et celle qui l'utilise. Par exemple, on souhaite s'assurer que `subject_token` délivré par PSC est bien distingué par le proxy de l'`access_token` délivré par le serveur d'autorisation.

L'outil utilisé pour la sécurisation des échanges et la bonne affection des jetons entre le client et le proxy varient selon le type de client :

- Si le client est un navigateur, **un cookie applicatif** avec une durée de vie limitée est initié par le proxy et est utilisé par le navigateur pour chaque nouvelle requête.
- Si le client est lourd, un **jeton applicatif** avec une durée de vie limitée est initié par le proxy et est utilisé par le client lourd pour chaque nouvelle requête

Dans les deux cas, un mapping par couple (clé,valeur) du type (ID jeton applicatif / ID cookie applicatif ; token) est utilisé pour permettre au proxy qui manipulent ces tokens de les distinguer.

Dans le cas des clients lourds, les différences quantitatives de cardinalité entre les instances LPS client, les proxys LPS et les serveurs d'autorisation sont décrites ci-dessous, du point de vue des éditeurs, pour l'accès à un seul service cible :

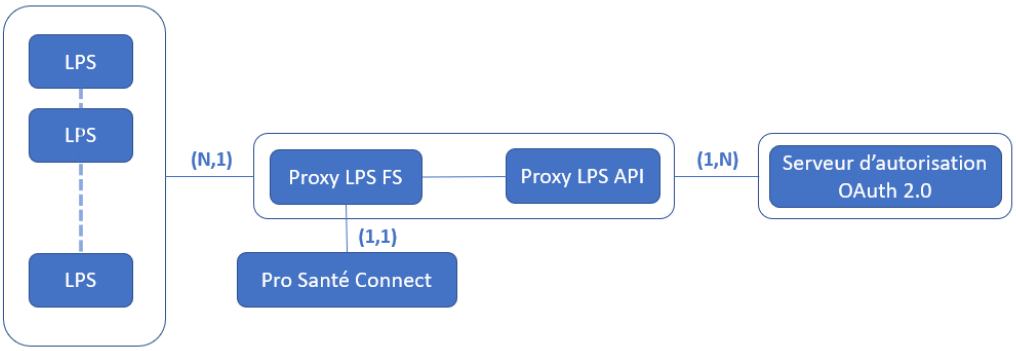

Figure 9 : Cardinalité entre les instances LPS client, les proxys LPS et les serveurs d'autorisation

Selon les éditeurs, il est possible qu'un seul proxy LPS FS soit mis à disposition et dans ce cas, la cardinalité entre le proxy LPS FS et le proxy LPS API est (1,1).

Du point de vue du serveur d'autorisation du service cible, les cardinalités sont décrites ci-dessous :

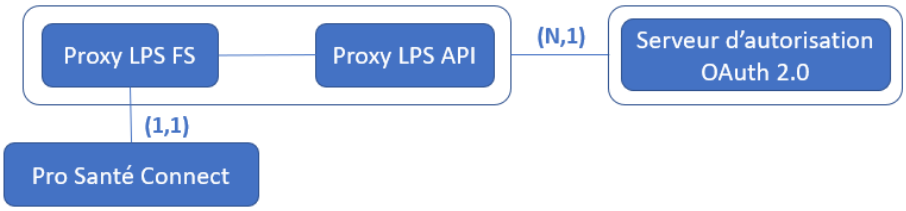

Figure 10 : Cardinalité entre les instances les proxys éditeurs et du serveur d'autorisation

Les proxys LPS API et LPS FS ont été rendus distincts car leurs rôles le sont. Il est toutefois possible pour un éditeur de regrouper ces deux fonctions au sein d'une même structure logicielle lors de l'implémentation de la sécurisation de l'architecture.

6.4. Cas d'usage #1 : API nécessitant une authentification Pro Santé Connect

Cas d'usage pour lequel le système cible nécessite une authentification de l'utilisateur par Pro Santé Connect pour l'accès à ses données. Pour ce faire, le fournisseur de services redirige d'abord l'utilisateur vers Pro Santé Connect pour l'authentifier.

Cependant, si le fournisseur de services dispose déjà d'un `subject_token` valide délivré par Pro Santé Connect (voir le cas d'utilisation #1a), la première étape d'authentification n'est pas nécessaire.

Une fois l'utilisateur authentifié, le fournisseur de services utilise le protocole OAuth 2.0 pour initier une requête au nom de l'utilisateur pour récupérer ses données auprès du service cible au sein du système cible.
Le système initiateur peut être une **application web** ou un **client lourd**.

Dans le cadre du token exchange RFC 8693 [12], **le jeton échangé contre l'`access_token`** délivré par le serveur d'autorisation OAuth 2.0 permet **d'identifier l'utilisateur, il se nomme** `subject_token`.

Afin de garantir une cohérence avec les paramètres du RFC et dans le cas des API Pro Santé connectées : `subject_token` = Access Token PSC

6.4.1.1. Appel depuis une application web

L'utilisateur souhaite, via **l'application web de son navigateur**, accéder à des données protégées d'un service cible. L'utilisateur s'authentifie premièrement auprès de Pro Santé Connect via l'application web de son fournisseur de services.
Une fois authentifié, le fournisseur de services émet une requête auprès du serveur d'autorisation (`subject_token` + scope) pour accéder aux ressources.
Si le `subject_token` et les scopes sont jugés valides par le serveur d'autorisation, ce dernier fournit un `access_token` au fournisseur de services.

Le fournisseur de services, s'assure de la validité de `l'access_token` à sa réception puis envoie une requête auprès du service cible avec `l'access_token` contenu dans l'entête. Le service cible introspecte `l'access_token` auprès du serveur d'autorisation afin vérifier la validité de `l'access_token` et des scopes.

Le fournisseur de services utilise `l'access_token` et les scopes auprès du service cible pour accéder aux données protégées.

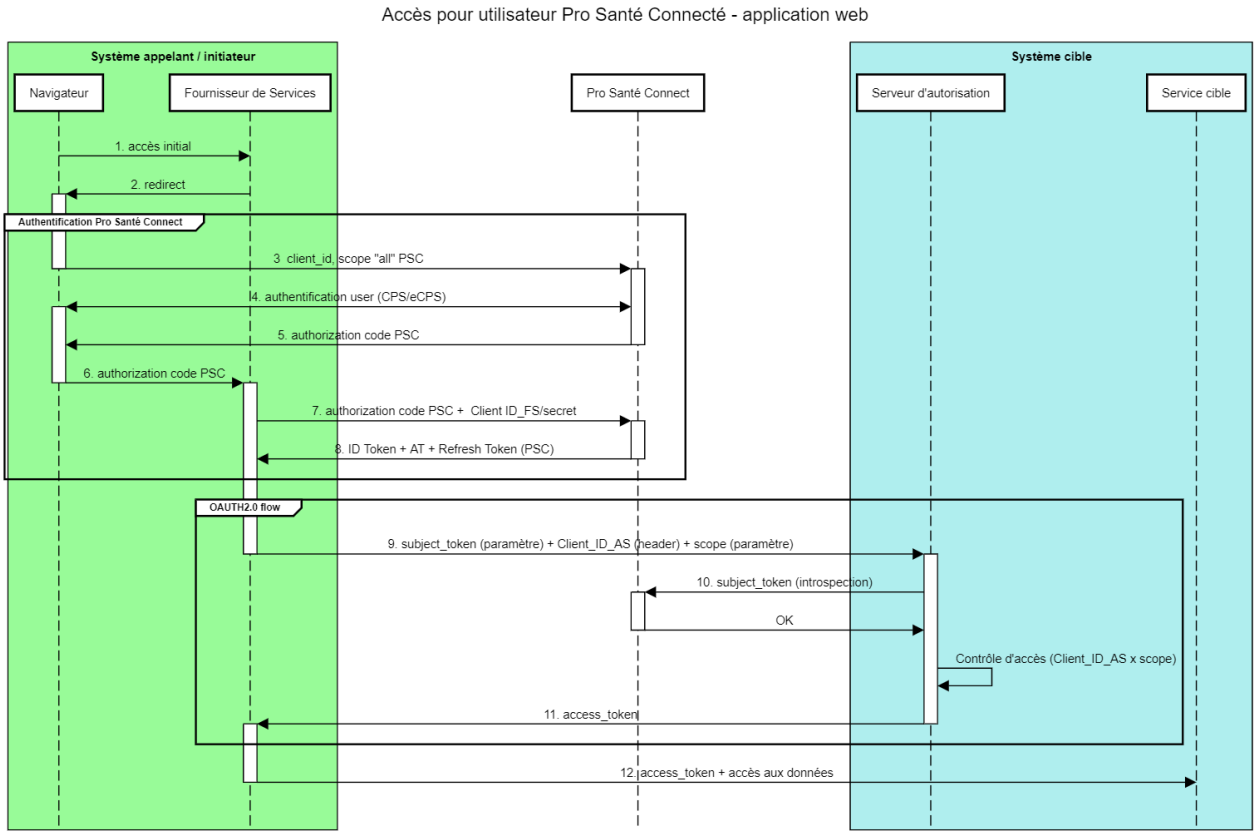

Figure 11 : Accès pour un utilisateur Pro Santé Connecté - Application Web

Description du workflow :

1. **Depuis le navigateur, l'utilisateur souhaite accéder à des données protégées d'un service cible**

   Origine : Navigateur
   Cible : Fournisseur de services
   Méthode : GET

   Exemple :

```
GET /application/ressource_privee
Host:
applicationserver.appelant.org
```

2. **Le fournisseur de services redirige le navigateur vers Pro Santé Connect pour authentifier l'utilisateur**

   Origine : Fournisseur de services
   Cible : Navigateur

   Le fournisseur de services renvoie un **code de redirection 302** avec **l'URL** de Pro Santé Connect

   Exemple :

```
302
Location: https://wallet.esw.esante.gouv.fr/auth?client_id=${client
id}&redirect_uri= ${callback uri} &response_type=code&state=<random
string>&scope= openid+scope_all&acr=eidas1&nonce=<random string>
```

3. **L'utilisateur accède à la mire d'authentification Pro Santé Connect**

   L'authentification est décrite dans la documentation officielle de PSC [4]

4. **Authentification de l'utilisateur sur Pro Santé Connect au moyen du MIE PSC**

   L'authentification est décrite dans la documentation officielle de PSC [4]

5. **Pro Santé Connect génère un authorization code PSC et redirige l'utilisateur vers le navigateur**

   Origine : Pro Santé Connect
   Cible : Navigateur

   Une fois l'authentification terminée, PSC renvoie un code HTTP 302 pour demander au navigateur d'envoyer l'authorization code au FS.

   Exemple :

```
302 Location:
https://applicationserver.appelant.fr/application/callback?code=3159339c
2f1326f9fa128&
state= <random_string>
```

Pro Santé Connect redirige l'utilisateur vers la callback URI avec en paramètre de la requête :

Paramètres	Obligatoire d'après la documentation officielle PSC	Obligatoire pour l'interopérabilité des SIS	Valeur
code	Oui	Oui	'code'
state	Oui	Oui	Valeur envoyée par le Fournisseur de services lors de la demande d'autorisation

6. Le navigateur envoie l'authorization code vers le fournisseur de services

Origine : Navigateur
Cible : Fournisseur de services
Méthode : GET

Exemple :

```
GET /application/callback ?code=3159339c2f1326f9fa128&state=<random string>
Host: applicationserver.appelant.fr
```

7. Le fournisseur de services s'authentifie auprès de PSC avec l'authorization code PSC et son `Client_ID_FS` et certificat TLS

Origine : Fournisseur de services
Cible : Pro Santé Connect

L'authentification du serveur d'application est décrite dans la documentation [4]

8. Pro Santé Connect vérifie l'authorization code et génère un ID Token, un access token PSC et un Refresh Token (PSC)

Origine : Pro Santé Connect
Cible : Fournisseur de services

Les détails de cette étape et des jetons délivrés sont dans la documentation [4]

Réponse de PSC :

```
{
      'access_token':
${access_token},
      'token_type': 'Bearer',
      'refresh_token':
      ${refresh_token},
      'expires_in':
      ${expiration},
       'id_token': ${id_token}
}
```

9. Le fournisseur de services s'authentifie et demande un échange de jetons auprès du serveur d'autorisation avec un `subject_token`, certificat de structure et scopes métier

NB : La section OAUTH2.0 Flow n'est pas dépendante du type de FS

Origine : Fournisseur de services
Cible : Serveur d'autorisation
Méthode : POST

Lors de ce flux, le fournisseur de services effectue une connexion mTLs avec le certificat de structure, auprès du serveur d'autorisation puis s'authentifie et envoie sa requête d'échange de jetons selon le protocole OAuth 2.0.

Concernant l'échange de jetons, il suit le cadre définit par le Token Exchange dans la **RFC 8693 [12]**. Elle est nativement **en voie d'adoption** par les éditeurs du marché (méthode Delegation Token Exchange).
Le fournisseur de services envoie dans les paramètres de sa requête le **`subject_token`** et les **scopes** métiers auxquels il souhaite accéder.
Ces derniers permettent d'authentifier la **personne physique** ou la **personne morale** appelante et de contrôler l'accès aux ressources requêtées.
Une fois ces éléments introspectés et validés, le **serveur d'autorisation renvoie un `access_token` pour permettre au fournisseur de services d'accéder aux ressources du service cible**.

Paramètres	Obligatoire d'après la documentation officielle (OAuth 2.0)	Obligatoire pour l'interopérabilité des SIS	Valeur
`grant_type`	Oui	Oui	urn:ietf:params:oauth:grant-type:token-exchange
`subject_token`	Oui	Oui	Access Token PSC
`subject_token_type`	Oui	Oui	urn:ietf:params:oauth:token-type:jwt
`scope`	Non	Oui	A déterminer par les équipes métiers = scopes métier

L'authentification du fournisseur de services auprès du serveur d'autorisation se fait selon la RFC 7235 [11], c'est-à-dire dans le header de la requête suivant la méthode « Authentification Basic » qui encode les credentials en base 64.

Exemple :

```
 POST /as/token.oauth2 HTTP/1.1
 Host: systemecible.example.com
 Authorization: Basic cnMwODpsb25nLXNlY3VyZS1yYW5kb20tc2VjcmV0
 Content-Type: application/x-www-form-urlencoded

&grant_type=urn%3Aietf%3Aparams%3Aoauth%3Agrant-type%3Atoken-exchange
&subject_token=accVkjcJyb4BWCxGsndESCJQbdFMogUC5PbRDqceLTC
&subject_token_type= urn%3Aietf%3Aparams%3Aoauth%3Atoken-type%3Aaccess_token
&scope=
```

Concernant la connexion mTLS, elle se réalise dans le cadre d'une authentification OAuth 2.0 et suit la **RFC 8705 [13].** Le fournisseur de service établit une connexion TLS mutuelle avec le serveur d'autorisation en présentant son certificat client, préalablement aux échanges applicatifs.

Le client_secret est remplacé par la preuve de possession de la clé privée qui est assurée par le mTLS. En effet, le mTLS permet d'authentifier le client par un `Client_ID_AS` + mTLS au lieu de s'authentifier avec son `client_secret/Client_ID_AS.` Dans ce cas le `client_secret` n'est pas nécessaire car l'authentification est portée par le mTLS (certificat client IGC Santé).

Dans la pratique, il est envisagé d'utiliser **Certificat IGC Santé ORG AUTH_CLI qui** identifiera le fournisseur de services auprès du serveur d'autorisation (**`Client_ID_AS`**) lors de la requête.

Le contenu du certificat, en particulier le DN, est accessible au serveur d'autorisation.
Le **DN** sujet du certificat client possède un attribut **OU** (Organizational Unit) qui lui contient :

- L'identifiant de la structure porteuse du fournisseur de services (**`Structure_ID`**) issu du référentiel d'identité utilisé par l'IGC Santé
- Un attribut **CN** (Common Name) qui a une valeur libre, et qui fera le lien entre la structure et le certificat

L'attribut CN du certificat peut permettre de lier un certificat à un service d'une structure.

Afin de garantir la cohérence et la bonne gestion des accès, le serveur d'autorisation aura la charge de faire le mapping entre le DN sujet du certificat TLS client et le `Client_ID_AS` (créé à l'enrôlement auprès du serveur d'autorisation). Si la corrélation échoue alors il doit y avoir un retour d'erreur.

10. Le serveur d'autorisation vérifie auprès de PSC la validité du `subject_token` (introspection)

Origine : Serveur d'autorisation
Cible : Pro Santé Connect
Méthode : POST

L'introspection du `subject_token` est **définie dans la documentation PSC [8].** La réponse de l'introspection est définie par **une structure de code[1] HTTP standard**.

11. Le serveur d'autorisation contrôle l'accès aux ressources selon le `Client_ID_AS` et les scopes du fournisseur de services puis lui délivre l'`access_token`

Origine : Serveur d'autorisation
Cible : Fournisseur de services

Le contrôle d'accès est basé sur le contrôle de l'association entre **scopes** et **`Client_ID_AS`** contenus dans le serveur d'autorisation.

Une fois le contrôle d'accès effectué et validé, le serveur d'autorisation forge un `access_token` permettant d'accéder au service cible et le délivre au fournisseur de services appelant.

[1] (Wikipédia, n.d.) : Liste des codes HTTP — Wikipédia (wikipedia.org)

Exemple :

```
HTTP/1.1 200 OK
Content-Type: application/json
Cache-Control: no-cache, no-store
{
"access_token":"eyJhbGciOiJFUzI1NiIsImtpZCI6IjllciJ9.eyJhdWQiOiJodHRwczovL2JhY2t
lbmQuZXhhbXBsZS5jb20iLCJpc3MiOiJodHRwczovL2FzLmV.40y3ZgQedw6rxEQgU85AI9x3KmsPott
VMLPIWvmDCMy5-kdXjwhw",
 "issued_token_type":
      "urn:ietf:params:oauth:token-type:access_token",
 "token_type":"Bearer",
 "expires_in":60
}
```

L'interopérabilité impose d'être conforme à la section 2.2 du RFC Token Exchange [12]

12. Le fournisseur de services requête la ressource protégée auprès du service cible

Origine : Fournisseur de services
Cible : Service cible
Type d'appel : Requête de ressources protégées
Méthode : GET

Le fournisseur de services envoie une requête auprès du service cible avec l'access_token contenu dans l'entête. Le service cible introspecte l'access_token auprès du serveur d'autorisation afin vérifier la validité de ce dernier ainsi que ses scopes associés.

Exemple :

```
GET /resource/v1 HTTP/1.1
Host:
Authorization: Bearer oab3thieWohyai0eoxibaequ0Owae9oh
```

Une fois l'introspection validée, le fournisseur de services a accès aux ressources protégées du service cible.

Cas d'usage #1a : un utilisateur déjà Pro Santé Connecté a besoin du Refresh Token PSC pour poursuivre sa navigation et accéder à une ressource

Dans ce cas, l'utilisateur est déjà authentifié auprès de Pro Santé Connect et souhaite accéder à une ressource d'un service cible, mais **son `subject_token` a expiré**. Afin de poursuivre son parcours, le fournisseur de services a besoin de récupérer un nouvel `subject_token` grâce à son **Refresh Token PSC encore valide (<2 min)**.

Ici, lorsque le serveur d'autorisation effectue une introspection auprès de PSC pour vérifier la validité des jetons, PSC détecte que le `subject_token` n'est plus valide (durée de validité limitée à **2 minutes [4]**).
Le fournisseur de services effectue alors une demande de renouvellement de jetons auprès de Pro Santé Connect. Il utilise son **Refresh Token [4]** encore valide (<30 minutes) afin d'obtenir un nouvel **`subject_token`**.

Une fois le `subject_token` récupéré, le serveur d'application relance une nouvelle cinématique OAuth 2.0.

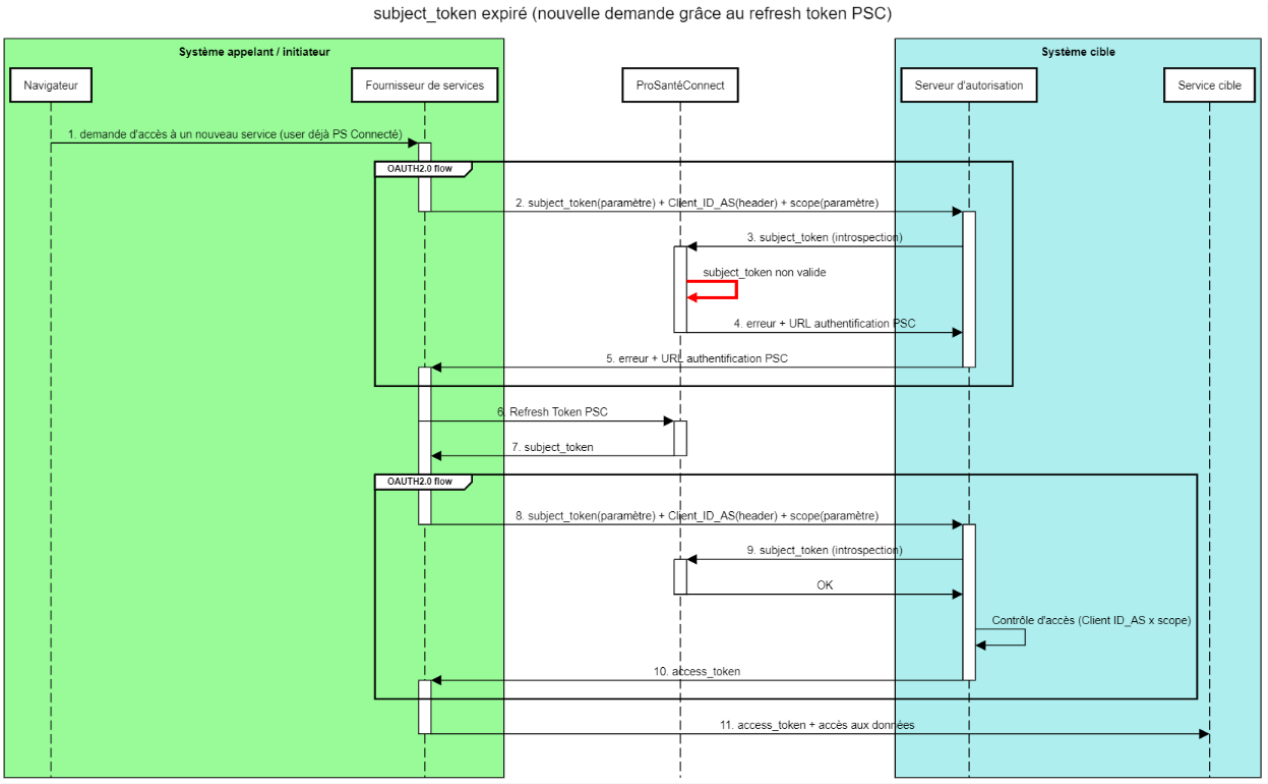

Figure 12 : Demande d'un nouveau subject_token grâce à un Refresh Token PSC pour un utilisateur déjà Pro Santé Connecté - Application Web

Cas d'usage #1b : La session PSC est expirée, l'utilisateur doit se réauthentifier sur la mire d'authentification

L'utilisateur est déjà authentifié auprès de Pro Santé Connect et souhaite accéder à une ressource d'un service cible mais sa session a expiré (cookie de session > 4 heures ou 30 min d'inactivité) [4]. Par conséquent, **son `subject_token` ainsi que son Refresh Token PSC sont expirés** (`subject_token` > 2min et Refresh Token > 30min).

- **Cas d'usage #1b.1 : Le cookie de session de l'utilisateur est expiré et redirection vers mire authentification - Application Web**

Dans la navigation, si l'utilisateur a besoin des ressources nécessitant d'être PSConnecté, le fournisseur de services va vérifier que la session est valide auprès de PSC (cookie de session). Lors de l'introspection du cookie de session par PSC, ce dernier ne valide pas son cookie de session car il a expiré (>4 heures). PSC renvoie une réponse d'erreur et l'URL de sa mire d'authentification afin que le fournisseur de services redirige l'utilisateur vers PSC pour s'authentifier à nouveau.

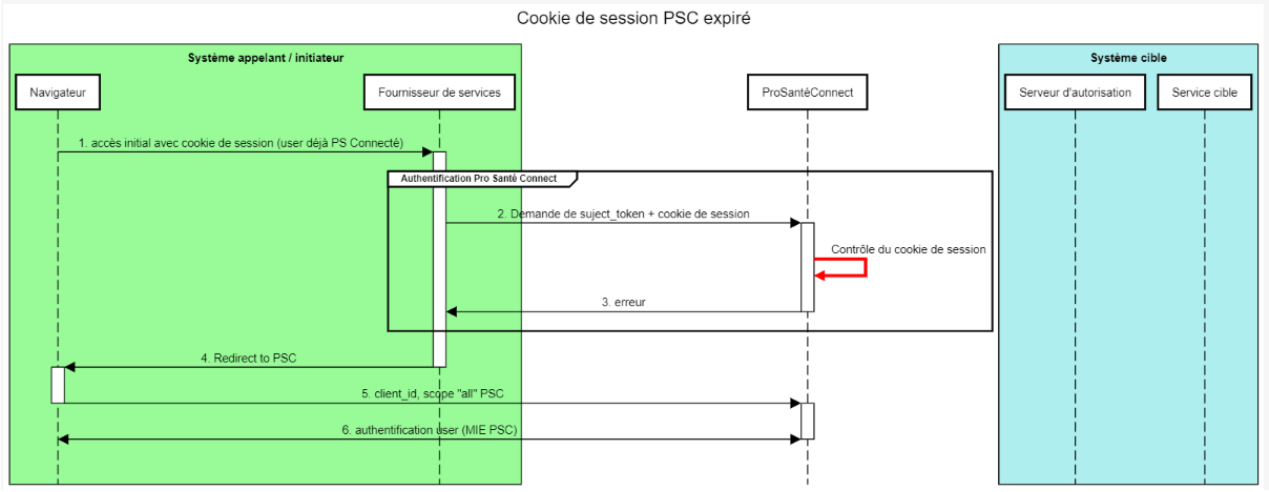

Figure 13 : Le cookie de session de l'utilisateur est expiré et redirection vers mire authentification - Application Web

- **Cas d'usage #1b.2 : Tentative d'accès aux ressources avec un `subject_token` expiré, suivi d'une tentative de nouvellement avec Refresh Token PSC lui aussi expiré, puis redirection vers la mire authentification - Application Web**

Lors de l'introspection du `subject_token`, PSC ne le valide pas car il a expiré. PSC renvoie une réponse d'erreur au fournisseur de services qui tente de renouveler son `subject_token` avec son Refresh Token PSC.
Cependant, ce dernier est lui aussi expiré, amenant PSC à rediriger le PS vers l'URL de la mire d'authentification PSC pour s'authentifier de nouveau.

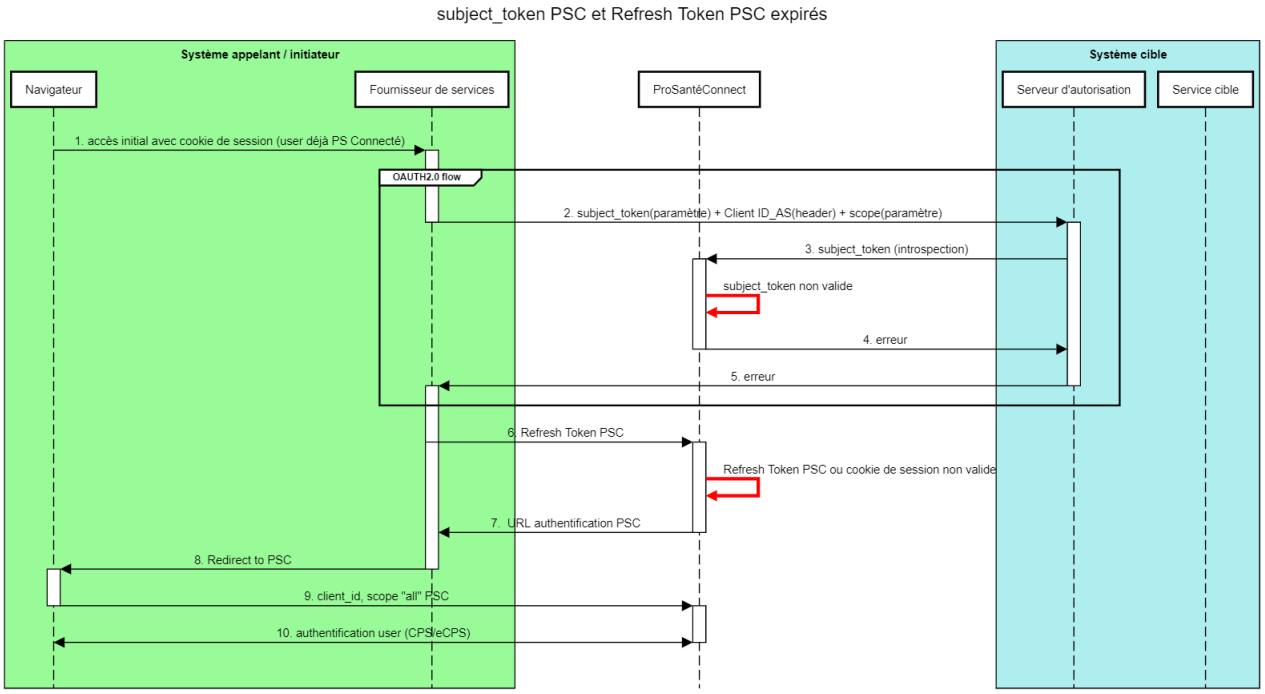

Figure 14 : Le cookie de session de l'utilisateur a expiré – tentative d'accès avec subject_token, de nouvellement avec Refresh Token puis redirection vers la mire authentification - Application Web

6.4.1.2. Appel depuis un client lourd

On considère dans cette section un client lourd comme un mobile. Pour faciliter la rédaction de cette section, on ne mentionnera que le client lourd mais cette section s'applique également au mobile
Un appel depuis un client lourd, comme un LPS ou un mobile, peut se faire via deux processus distincts :

- **Protocole CIBA** : permet à l'utilisateur de s'authentifier via son device (ex : mobile) à Pro Santé Connect – uniquement pour les MIE compatibles Actuellement, Pro Santé Connect ne permet pas une authentification avec les cartes CPS en CIBA. La cible serait de le permettre.
- **Navigateur extérieur (pop-up web) :** ouverte en parallèle du client lourd (pop-up indépendante ou onglet du navigateur système), elle permet l'authentification de l'utilisateur sur Pro Santé Connect.

6.4.1.2.1. Authentification par processus CIBA

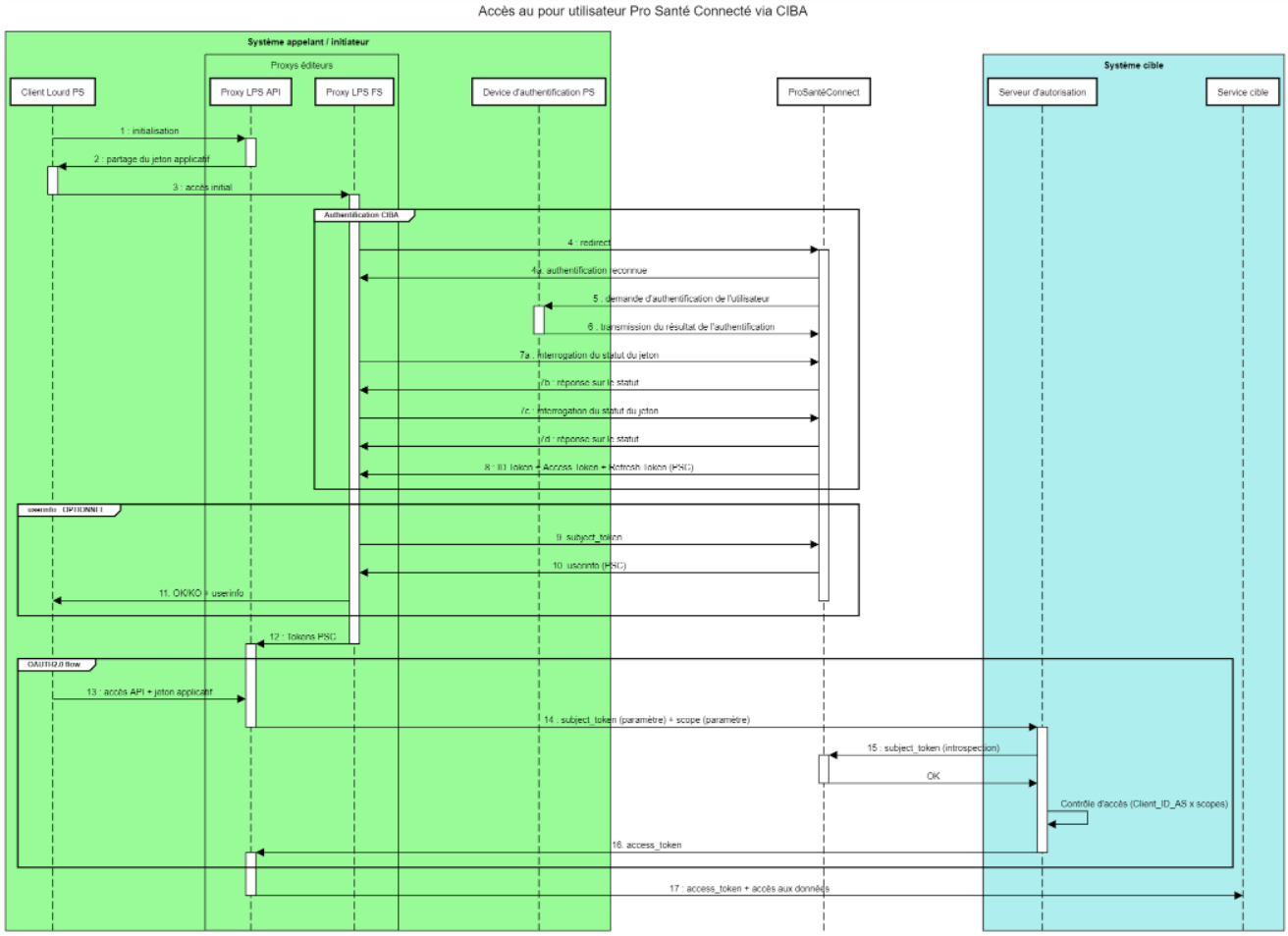

Figure 15 : Accès pour un utilisateur Pro Santé Connecté via le protocole CIBA – Client Lourd

Description du workflow :

1. **Le client lourd initie une demande de jeton applicatif/cookie web auprès du proxy LPS API**

   Origine : Client lourd
   Cible : Proxy LPS API

2. **Le proxy LPS API génère le jeton applicatif et l'envoie au client lourd**

   Origine : Proxy LPS API
   Cible : Client lourd

   Le client lourd conserve le jeton applicatif. Ce jeton applicatif sera utilisé pour chaque requête depuis le client lourd vers le proxy LPS API.

3. **Depuis le client lourd, l'utilisateur souhaite accéder à des données protégées d'un service cible**

   Origine : Client Lourd
   Cible : Proxy LPS FS
   Méthode : GET

   Exemple :

   ```
   GET /application/ressource_privee
   Host: applicationserver.appelant.org
   ```

4. **Authentification CIBA client lourd suivant la documentation du processus CIBA [5]**

   Valable pour les flux **4 à 8**.

9. **[UserInfo - OPTIONNEL] Le FS requête le userinfo avec un `subject_token` en entête au Userinfo Endpoint de PSC**

   Origine : Proxy LPS FS
   Cible : Userinfo Endpoint PSC
   Méthode : GET

10. **[UserInfo - OPTIONNEL] Le FS récupère le jeton UserInfo et renvoie une réponse de succès au client lourd**

    Origine : Userinfo Endpoint PSC
    Cible : Proxy LPS FS

11. [UserInfo - OPTIONNEL] Le Client Lourd récupère le jeton UserInfo

Origine : Proxy LPS FS
Cible : Client Lourd

12. Le Proxy LPS FS fournit au proxy LPS API les tokens PSC qu'il a obtenus à l'issue de l'authentification de l'utilisateur

Origine : Proxy LPS FS
Cible : Proxy LPS API

Les Tokens PSC sont : ID Token PSC, AT PSC et Refresh Token PSC

13. Le PS depuis, son client lourd initie, un accès API sur le proxy LPS API avec le jeton applicatif préalablement fourni par le proxy LPS API

Origine : Client lourd
Cible : Proxy LPS API
Méthode : GET

14. Le proxy LPS API s'authentifie et demande un échange de jetons auprès du serveur d'autorisation avec le `subject_token`, certificat de structure et les scopes métiers.

Origine : Proxy LPS API
Cible : Serveur d'autorisation
Méthode : POST

Lors de ce flux, le fournisseur de services effectue une connexion mTLS avec le certificat structure auprès du serveur d'autorisation puis s'authentifie et envoie sa requête d'échange de jetons selon le protocole OAuth 2.0.

Concernant l'échange de jetons, il suit le cadre définit par le Token Exchange dans la **RFC 8693** [12]. Elle est nativement **en voie d'adoption** par les éditeurs du marché (méthode Delegation Token Exchange).
Le fournisseur de services envoie dans les paramètres de sa requête le **`subject_token`** et les **scopes** métiers auxquels il souhaite accéder.
Ces derniers permettent d'authentifier la **personne physique** ou la **personne morale** appelante et de contrôler l'accès aux ressources requêtées.
Une fois ces éléments introspectés et validés, le **serveur d'autorisation renvoie un `access_token` pour permettre au fournisseur de services d'accéder aux ressources du service cible**.

Paramètres	Obligatoire d'après la documentation officielle (OAuth 2.0)	Obligatoire pour l'interopérabilité des SIS	Valeur
grant_type	Oui	Oui	urn:ietf:params:oauth:grant-type:token-exchange
subject_token	Oui	Oui	Access Token PSC
subject_token_type	Oui	Oui	urn:ietf:params:oauth:token-type:jwt
scope	Non	Oui	A déterminer par les équipes métiers = scopes métier

L'authentification du fournisseur de services auprès du serveur d'autorisation, se fait selon la RFC 7235 [11], c'est-à-dire dans le header de la requête suivant la méthode « Authentification Basic » qui encode les credentials en base 64.

Exemple :

```
POST /as/token.oauth2 HTTP/1.1
Host: systemecible.example.com
Authorization: Basic cnMwODpsb25nLXNlY3VyZS1yYW5kb20tc2VjcmV0
Content-Type: application/x-www-form-urlencoded

&grant_type=urn%3Aietf%3Aparams%3Aoauth%3Agrant-type%3Atoken-exchange
&subject_token=accVkjcJyb4BWCxGsndESCJQbdFMogUC5PbRDqceLTC
&subject_token_type= urn%3Aietf%3Aparams%3Aoauth%3Atoken-type%3Aaccess_token
&scope=
```

Concernant la connexion mTLS, elle se réalise dans le cadre d'une authentification OAuth 2.0 et suit la **RFC 8705 [13].** Le fournisseur de service établit une connexion TLS mutuelle avec le serveur d'autorisation en présentant son certificat client, préalablement aux échanges applicatifs.

Le client_secret est remplacé par la preuve de possession de la clé privée qui est assurée par le mTLS. En effet, le mTLS permet d'authentifier le client par un `Client_ID_AS` + mTLS au lieu de s'authentifier avec son `client_secret/Client_ID_AS.` Dans ce cas le `client_secret` n'est pas nécessaire car l'authentification est portée par le mTLS (certificat client IGC Santé).

Dans la pratique, il est envisagé d'utiliser **Certificat IGC Santé ORG AUTH_CLI** qui identifiera le fournisseur de services auprès du serveur d'autorisation (**`Client_ID_AS`**) lors de la requête.

Le contenu du certificat, en particulier le DN, est accessible au serveur d'autorisation.
Le **DN** sujet du certificat client possède un attribut **OU** (Organizational Unit) qui lui contient :

- L'identifiant de la structure porteuse du fournisseur de services (**`Structure_ID`**) issu du référentiel d'identité utilisé par l'IGC Santé
- Un attribut **CN** (Common Name) qui a une valeur libre, et qui fera le lien entre la structure et le certificat

L'attribut CN du certificat peut permettre de lier un certificat à un service d'une structure.

Afin de garantir la cohérence et la bonne gestion des accès, le serveur d'autorisation aura la charge de faire le mapping entre le DN sujet du certificat TLS client et le `Client_ID_AS` (créé à l'enrôlement auprès du serveur d'autorisation). Si la corrélation échoue alors il doit y avoir un retour d'erreur.

15. Le serveur d'autorisation vérifie auprès de PSC la validité du `subject_token` (introspection)

Origine : Serveur d'autorisation
Cible : PSC
Méthode : POST

L'introspection du `subject_token` est **défini dans la documentation PSC [4].** La réponse de l'introspection est définie par **une structure de code[2] HTTP standard**.

[2] (Wikipédia, n.d.) : Liste des codes HTTP — Wikipédia (wikipedia.org)

16. Le serveur d'autorisation contrôle l'accès aux ressources selon le `Client_ID_AS` et ses scopes puis délivre l'access_token au proxy LPS API

Origine : Serveur d'autorisation
Cible : Proxy LPS API

Le contrôle d'accès basé sur le contrôle de l'association entre scopes et `Client_ID_AS` contenus dans le serveur d'autorisation.

Une fois le contrôle d'accès effectué et validé, le serveur d'autorisation forge un access_token permettant d'accéder au service cible et le délivre au proxy LPS API appelant.

Exemple :

```
HTTP/1.1 200 OK
Content-Type: application/json
Cache-Control: no-cache, no-store
{
"access_token":"eyJhbGciOiJFUzI1NiIsImtpZCI6IjllciJ9.eyJhdWQiOiJod
HRwczovL2JhY2tlbmQuZXhhbXBsZS5jb20iLCJpc3MiOiJodHRwczovL2FzLmV.40y
3ZgQedw6rxEQgU85AI9x3KmsPottVMLPIWvmDCMy5-kdXjwhw",
 "issued_token_type":
      "urn:ietf:params:oauth:token-type:access_token",
 "token_type":"Bearer",
 "expires_in":60
}
```

Le cadre impose d'être conforme à la section 2.2 du RFC 8693 Token Exchange [12].

17. Le proxy LPS API requête la ressource protégée auprès du service cible

Origine : Proxy LPS API
Cible : Service cible
Type d'appel : Requête de ressources protégées
Méthode : GET

Le proxy LPS API envoie une requête auprès du service cible avec l'access_token contenu dans l'entête. Le service cible introspecte l'access_token du serveur d'autorisation afin vérifier la validité de l'AT et des scopes.

Exemple :

```
GET /resource/v1 HTTP/1.1
Host: https://apifournisseurdedonnees.com/scope1/resourceX
Authorization: Bearer oab3thieWohyai0eoxibaequ0Owae9oh
```

6.4.1.2.2. Authentification via un navigateur extérieur (pop-up web)

L'utilisateur souhaite accéder à des ressources chez un **service cible** à partir **de son client lourd**. Afin d'accéder à ces ressources, l'utilisateur s'authentifie auprès de **Pro Santé Connect via un navigateur extérieur (pop-up web).**

L'authentification par une application mobile est similaire à une authentification via un navigateur extérieur (pop-up web).

Une fois authentifié auprès de Pro Santé Connect, le **fournisseur de services** émet une requête auprès du serveur d'autorisation (`subject_token` + scope) pour accéder aux ressources. Une fois le **contrôle des scopes** effectué, le serveur d'autorisation fournit un access_token au fournisseur de services. Ce dernier l'utilise dans l'entête de sa requête auprès du service cible pour accéder aux données protégées.

Le partage d'un jeton applicatif est nécessaire en amont de la cinématique d'authentification pour permettre l'authentification du client lourd lors de la requête de ce dernier auprès du serveur d'autorisation.

Si le PS, depuis son client lourd, a besoin de préciser des champs liés à ses activités pour requêter des données auprès du service cible, **un flux de récupération du userinfo est exécuté**.
Les données passent par une application web serveur avec des redirections sur des URLs web.
Les données du jeton userInfo peuvent remonter jusqu'à l'application client lourd.

Se référer à la partie Appel depuis une application web pour la description du workflow

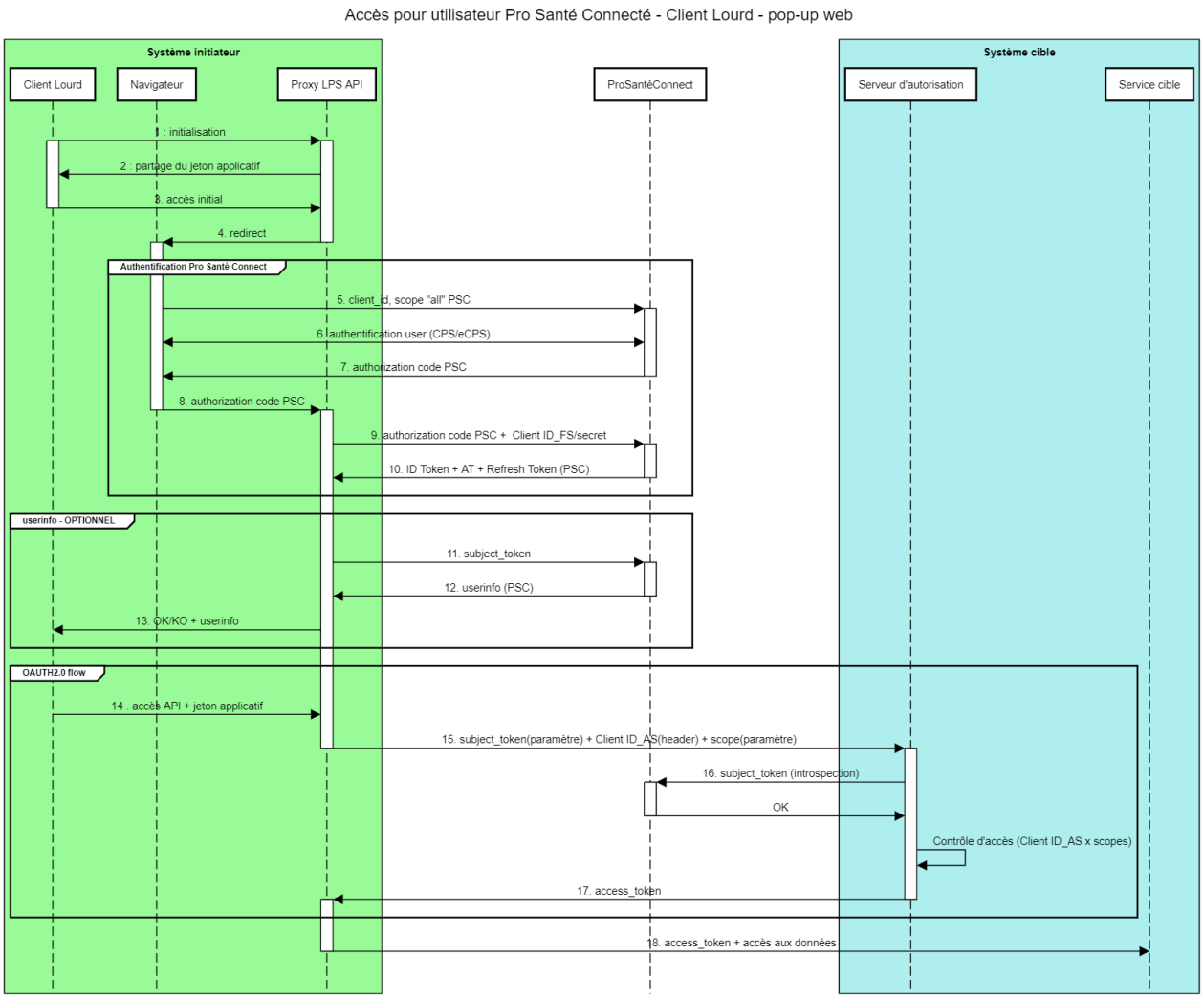

Figure 16 : Accès pour un utilisateur Pro Santé Connecté par un navigateur extérieur (pop-up web) – Client Lourd

6.4.1.2.3. Cas dérogatoires

Les modalités de sécurisations nécessitant de faire des échanges client-serveur entre le client lourd et le proxy LPS API fait l'objet d'un protocole HTTP.
Si les systèmes cibles utilisent des protocoles hors HTTP (par exemple SMTP) cela s'inscrit hors cadre du présent volet transport de ce CI-SIS.

6.4.1.3. User Info PSC

Une fois l'utilisateur authentifié sur Pro Santé Connect, l'obtention du jeton Userinfo du PS est demandé par le service cible.

Depuis un navigateur :

- **Cas où le fournisseur de services souhaite accéder au userinfo**

  La récupération du jeton UserInfo [10] par le fournisseur de services est décrite dans la documentation PSC.

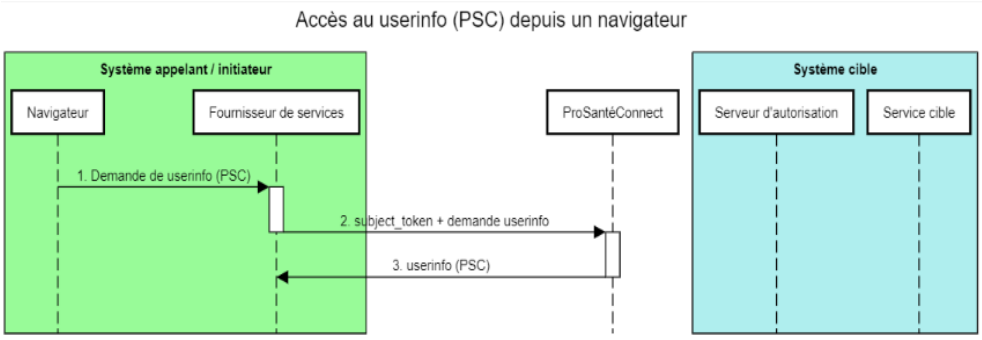

Figure 17 : Récupération de userinfo par le fournisseur de services - Navigateur

- **Cas où le service cible souhaite accéder au userinfo**

  Le service cible, par l'intermédiaire du serveur d'autorisation, souhaite récupérer le jeton userinfo PSC. Le jeton user info ne peut pas être récupéré directement auprès de PSC. Dans ce cas c'est le serveur d'autorisation qui le récupère et qui le met en cache. Le serveur d'autorisation récupère le userinfo durant la durée de validité du `subject_token`.

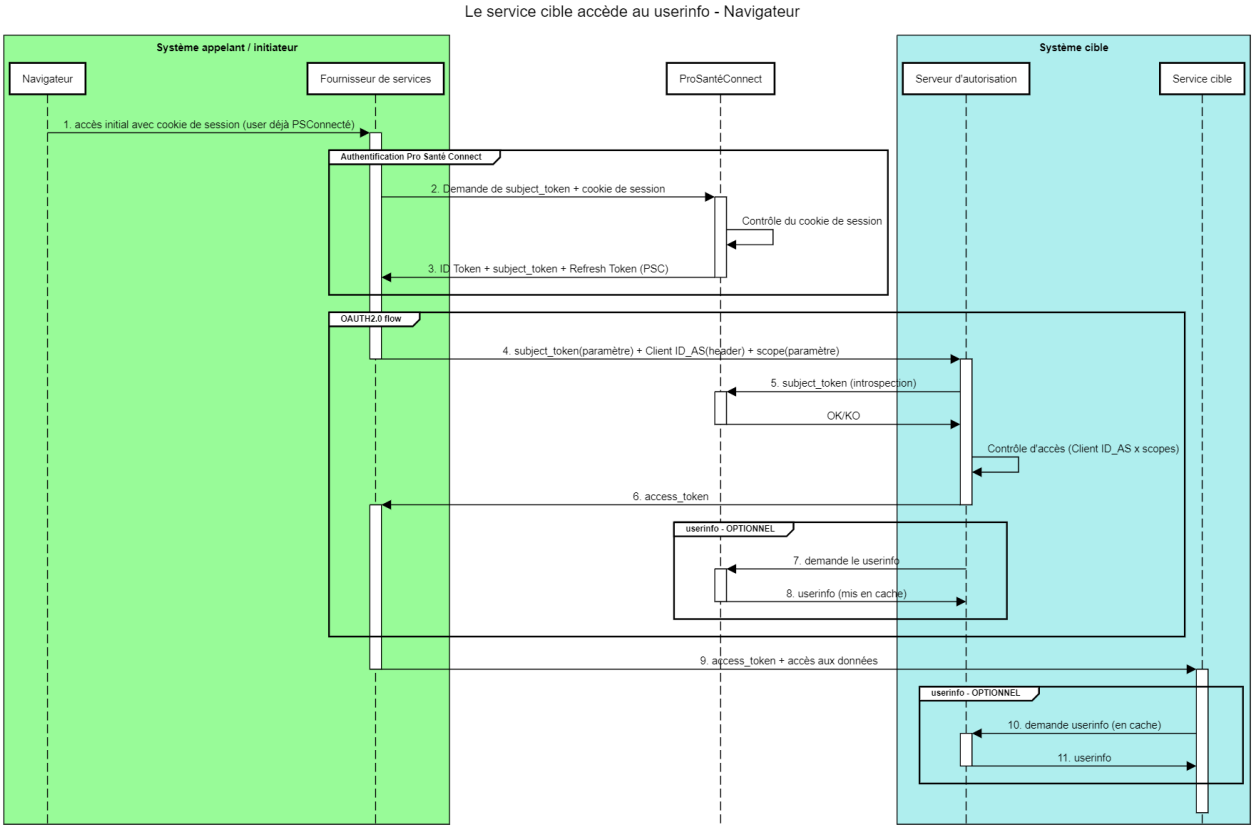

Figure 18 : Récupération du Userinfo par le service cible – Navigateur

Depuis un client lourd :

- **Cas où le PS souhaite consulter le userinfo :**

  Traité dans la partie Authentication via un navigateur extérieur (pop-up web) , **Figure 16 et 17.**

- **Cas où le service cible souhaite accéder au userinfo :**

  Le service cible, par l'intermédiaire du serveur d'autorisation, souhaite récupérer le jeton userinfo PSC. Le jeton user info ne peut pas être récupéré directement auprès de PSC. Dans ce cas c'est le serveur d'autorisation qui le récupère et qui le met en cache. Le serveur d'autorisation récupère le userinfo durant la durée de validité du `subject_token`.

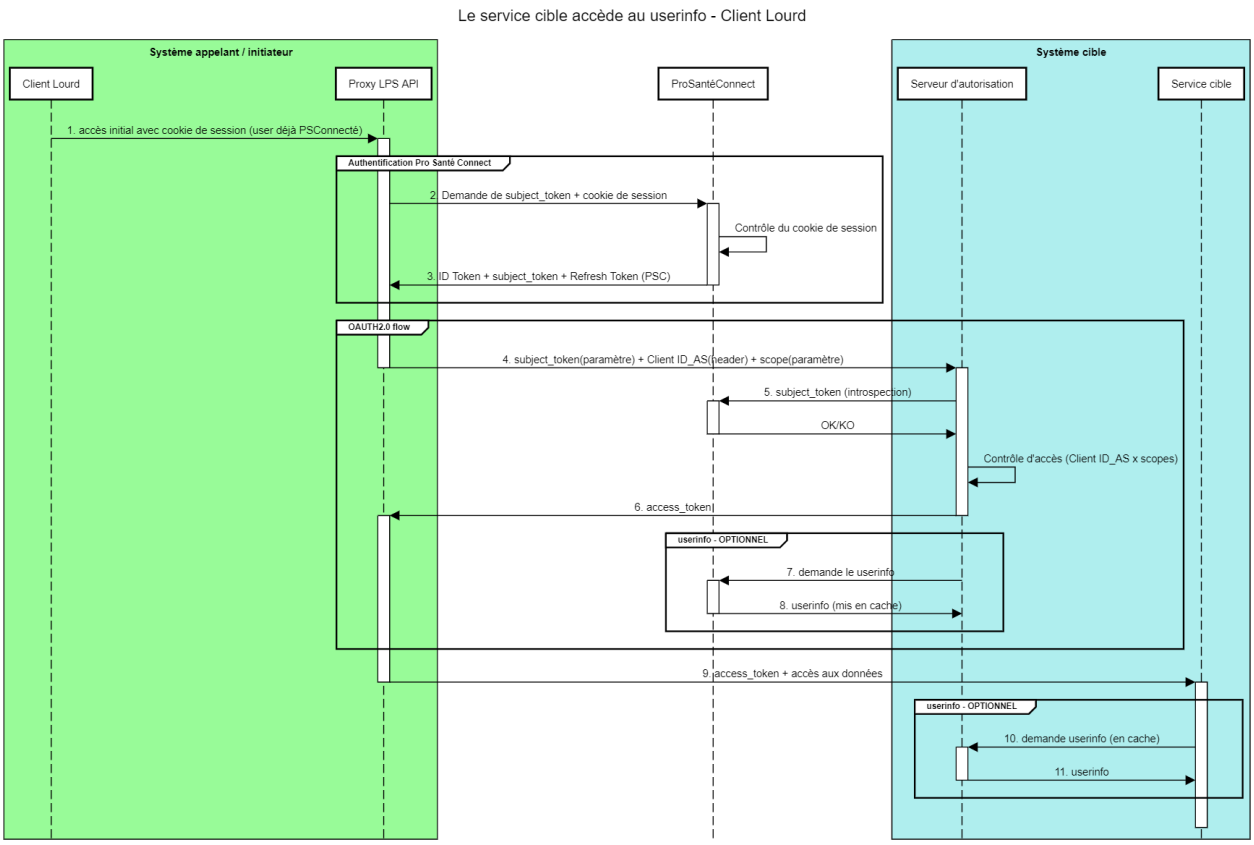

Figure 19 : Récupération de Userinfo par le service cible – Client lourd

6.4.1.3.1. Accès à des API Pro Santé Connectées nécessitant la remontée de données métiers complémentaires

Certains services cibles Pro Santé Connectés nécessitent l'envoi de données métiers complémentaires concernant le LPS, le proxy API ou l'activité du PS. Des services existants historiques assurent ce besoin grâce à l'assertion SAML et souhaitent minimiser l'impact de l'intégration du protocole OAuth 2.0. Pour les autres services souhaitant répondre à ce besoin, ils devront à terme utiliser le paramètre actor_token provenant du protocole OAuth 2.0. Les services historiques pourront basculer sur l'architecture cible ultérieurement.

6.4.1.3.1.1. Services cibles existants utilisant une assertion SAML

Pour les services historiques utilisant l'assertion SAML et qui voudraient minimiser les impacts côté client éditeurs, il est possible de créer une architecture temporaire pour l'usage du VIHF et de l'assertion SAML.

Le schéma ci-dessous présente l'intégration du requêtage par protocole SOAP dans un flux OAuth 2.0
Les flux 13 et 14 représentent l'accès à l'API avec la requête SOAP et l'assertion SAML. Le Client lourd LPS crée et peuple l'assertion SAML et génère la requête SOAP. Il est préconisé d'utiliser une connexion mTLS lors du flux 14.
Pour plus précisions pour plus de précisions concernant le flux 13 se référer au document CI-SIS volet transport synchrone Client Lourd et au document CI AMO volet transport synchrone pour les services de l'Assurance Maladie [1].

Le flux 17 représente l'accès au service cible avec la requête SOAP et l'assertion SAML avec l'access_token de l'API dans le bearer.

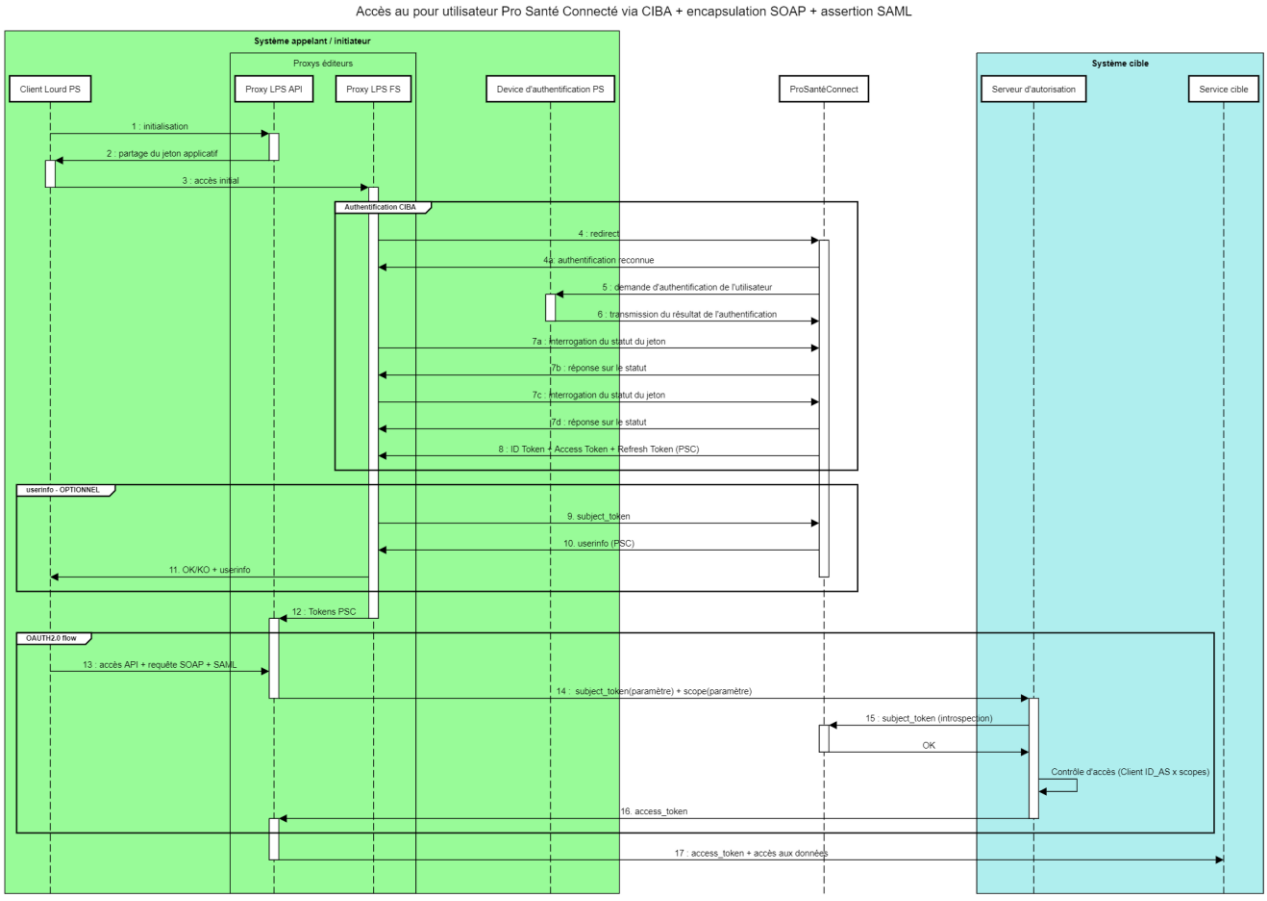

Figure 20 : Intégration du requêtage par protocole SOAP et assertion SAML dans le flow OAuth2.0

6.4.1.3.1.2. Services cibles nouveaux (ce paragraphe est en cours de rédaction)

Lors de cette demande, l'ajout du paramètre **`actor_token` [12]** dans la requête permet au serveur d'autorisation du service cible **d'identifier le proxy LPS API, le LPS et l'activité du PS**.
L'**`actor_token_type`** doit alors y être spécifié conjointement.

L'interopérabilité des SIS **impose** de spécifier un jeton du type JWT dont les claims sont les suivants :
- Ceux relatifs au LPS (lps_nom, lps_id, lps_version…).
- Situation d'exercice choisie par le PS sur son LPS.

Paramètres	Obligatoire d'après la documentation officielle (OAuth 2.0)	Obligatoire pour l'interopérabilité des SIS	Valeur
grant_type	Oui	Oui	urn:ietf:params:oauth:grant-type:token-exchange
`subject_token`	Oui	Oui	Access Token PSC
`subject_token_type`	Oui	Oui	urn:ietf:params:oauth:token-type:jwt
`actor_token`	Non	Non	JWT identifiant : ➢ Proxy LPS API ➢ Le LPS ➢ L'activité du PS
`actor_token_type`	Non	Non	urn:ietf:params:oauth:token-type:jwt
scope	Non	Oui	A déterminer par les équipes métiers = scopes metier

Exemple de claims du JWT décodé correspondant à l'`actor_token`

```
{
  "iss":"https://original-issuer.example.net",
  "exp":1441910060,
  "sub":admin@example.net,
   "lps_nom": <lps_nom>,
   "lps_id": <lps_id>,
   "lps_version": <lps_version>,
   "situation_exercice": <situation_exercice >
}
```

Selon les nécessités du service cible, l'access_token délivré peut être enrichi avec les claims de l'actor_token fournis lors de la requête.

Le JWT `actor_token` dans le cas des API Pro Santé Connectées pourrait être sujet à une signature par la clé privée du certificat cachet serveur au niveau de la structure porteuse du fournisseur de services. Cependant, le cas mTLS couvre ce besoin rendant la signature non nécessaire.

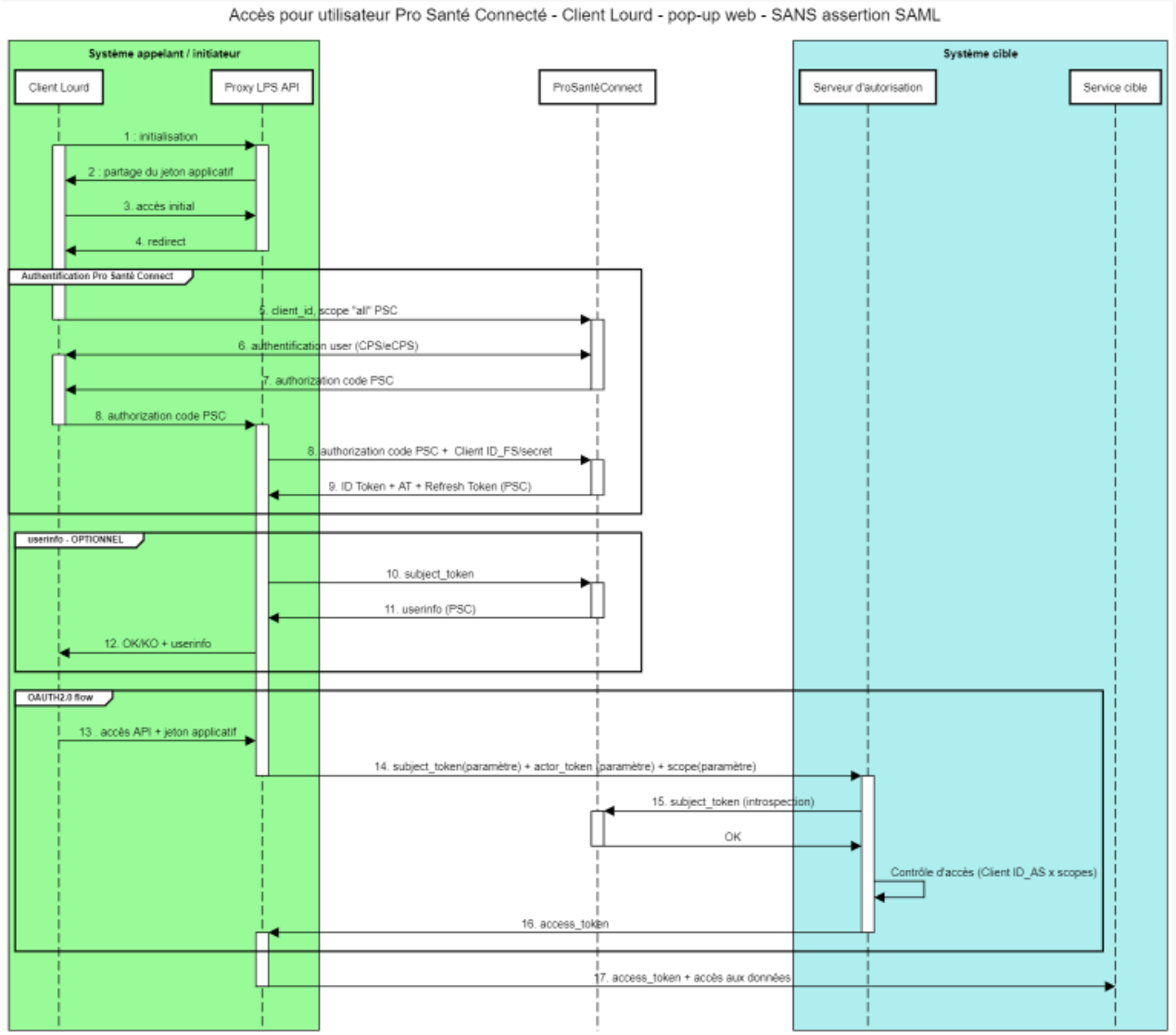

Figure 21 : Requêtage avec l'actor_token dans le flow OAuth2.0

6.5. (En cours de rédaction) Cas d'usage #2 : API non Pro Santé Connectées nécessitant l'authentification de l'utilisateur par le système cible

Cette section est en cours de rédaction et sera publiée dans une version ultérieure.

6.6. (En cours de rédaction) Cas d'usage #3 : Authentification utilisateur portée par la structure appelante

Cette section est en cours de rédaction et sera publiée dans une version ultérieure.

6.7. (En cours de rédaction) Cas d'usage #4 : API qui utilise un IDP tiers

Cette section est en cours de rédaction et sera publiée dans une version ultérieure.

6.8. Bibliographie

Wikipédia. (s.d.). Récupéré sur https://fr.wikipedia.org/wiki/Liste_des_codes_HTTP
Wikipédia. (s.d.). Récupéré sur https://fr.wikipedia.org/wiki/Liste_des_codes_HTTP

Annexe 1 : Historique du document

Version	Rédigé par		Vérifié par		Validé par	
1.0	ANS	Le 04/04/2023	ANS	Le 04/04/2023	ANS	Le 04/04/2023
	Motif et nature de la modification : **Publication pour première phase de concertation sur le chapitre API PSC Connectées**					
1.1	ANS	Le 02/06/2023	ANS	Le 02/06/2023	ANS	Le 02/06/2023
	Motif et nature de la modification : **Publication chapitre PSC Connectées**					

Annexe 2 : Documents de référence

Documents de référence	
[1]	ANS : CI-SIS Volet Transport Synchrone pour Client Lourd https://esante.gouv.fr/volet-transport-synchrone-pour-client-lourd GIE SESAM VITALE : CI AMO Volet transport synchrone pour les téléservices de l'Assurance Maladie https://industriels.sesam-vitale.fr/group/integrer-tlsi
[2]	ANS : SMT Serveur Multi Terminologie https://esante.gouv.fr/produits-services/smt
[3]	ANS : Corpus documentaire PGSSI-S https://esante.gouv.fr/produits-services/pgssi-s/corpus-documentaire?field_thematique_pgssi_target_id%5B745%5D=745&field_type_document_pgssi_target_id%5B759%5D=759
[4]	ANS : Document technique Concept de base du protocole OpenID Connect https://industriels.esante.gouv.fr/produits-et-services/pro-sante-connect/documentation-technique
[5]	ANS : CIBA https://industriels.esante.gouv.fr/produits-et-services/pro-sante-connect/ciba
[6]	GUIDE ANSSI : Recommendations pour la sécurisation de la mise en œuvre du protocole Open ID connect https://www.ssi.gouv.fr/uploads/2020/09/anssi-guide-recommandations_pour_la_securisation_de_la_mise_en_oeuvre_du_protocole_openid_connect-v1.0.pdf
[7]	IGC Santé : Les gabarits des certificats X.509 et des CRLs http://igc-sante.esante.gouv.fr/DA/DA-CONTRATS-TYPE-V1.0.pdf
[8]	ANS : Sources des données personnes et structures https://esante.gouv.fr/annexe-sources-des-donnees-personnes-et-structures
[9]	ANS : Introspection https://industriels.esante.gouv.fr/produits-et-services/pro-sante-connect/l-introspection
[10]	ANS : User Info https://industriels.esante.gouv.fr/produits-et-services/pro-sante-connect/userinfo

Annexe 3 : RFCs de référence

Documents de référence
[11] RFC 7235 Authentification Basic : https://www.rfc-editor.org/rfc/rfc7235 [12] RFC 8693 OAuth 2.0 Token Exchange : https://www.rfc-editor.org/rfc/rfc8693.html [13] RFC 8705 OAuth 2.0 mTLS Client Authentication and Certificate-Bound Access Tokens : https://www.rfc-editor.org/rfc/rfc8705 [14] RFC 7636 4.1 Code Verifier et 4.2 Code Challenge https://www.rfc-editor.org/rfc/rfc7636#section-4.1 [15] RFC 6749 Authorization Request https://www.rfc-editor.org/rfc/rfc6749#section-4.1.1 RFC 6749 Authorization Code Grant https://www.rfc-editor.org/rfc/rfc6749#section-4.1.1:~:text=4.3.1.%20%20Authorization%20Request%20and%20Response