AGENCE
DU NUMÉRIQUE
EN SANTÉ
La transformation commence ici

Guide d'appui

Outil de conformité pour Proxy e-Santé

Statut : Validé | *Classification : Publique* | *Version : v1.6*

Agence du Numérique en Santé
2 - 10 Rue d'Oradour-sur-Glane – 75015 Paris
T. 01 58 45 32 50
esante.gouv.fr

SOMMAIRE

1. CONTEXTE

Pro Santé Connect est un fournisseur d'identité sectorielle basé sur le standard OIDC qui permet aux professionnels de santé de s'authentifier aux services en lignes par la saisie d'un code PIN qui permet soit de lire le certificat contenu dans une carte CPS, soit d'ouvrir le certificat de l'application smartphone e-CPS.

En plus de l'authentification, **Pro Santé Connect propose un espace de confiance dans lequel les services interconnectés peuvent échanger des données de façon sécurisée**. Cet espace de confiance propose une traçabilité et une sécurité poussée qui permettent de retracer tout le parcours des échanges, depuis le PS jusqu'à l'API Pro Santé Connectée en passant par un proxy e-Santé.

Afin de simplifier le processus d'adhésion à l'espace de confiance, **l'ANS propose un outil de conformité aux éditeurs de proxy e-Santé**

2. OBJECTIF

Le **Proxy e-Santé** est le serveur intermédiaire, ne disposant pas d'interface utilisateur, destiné à **sécuriser les échanges de données entre le fournisseur de service Utilisateur (FSU) et l'API Pro Santé Connectée**.

Cet outil de test permet aux éditeurs de Proxy e-Santé de valider la conformité de leur composant en testant en autonomie et à leur rythme que ce dernier réponde à un sous-ensemble d'exigences de l'espace de confiance du Proxy e-Santé garantissant une sécurité commune à tous.

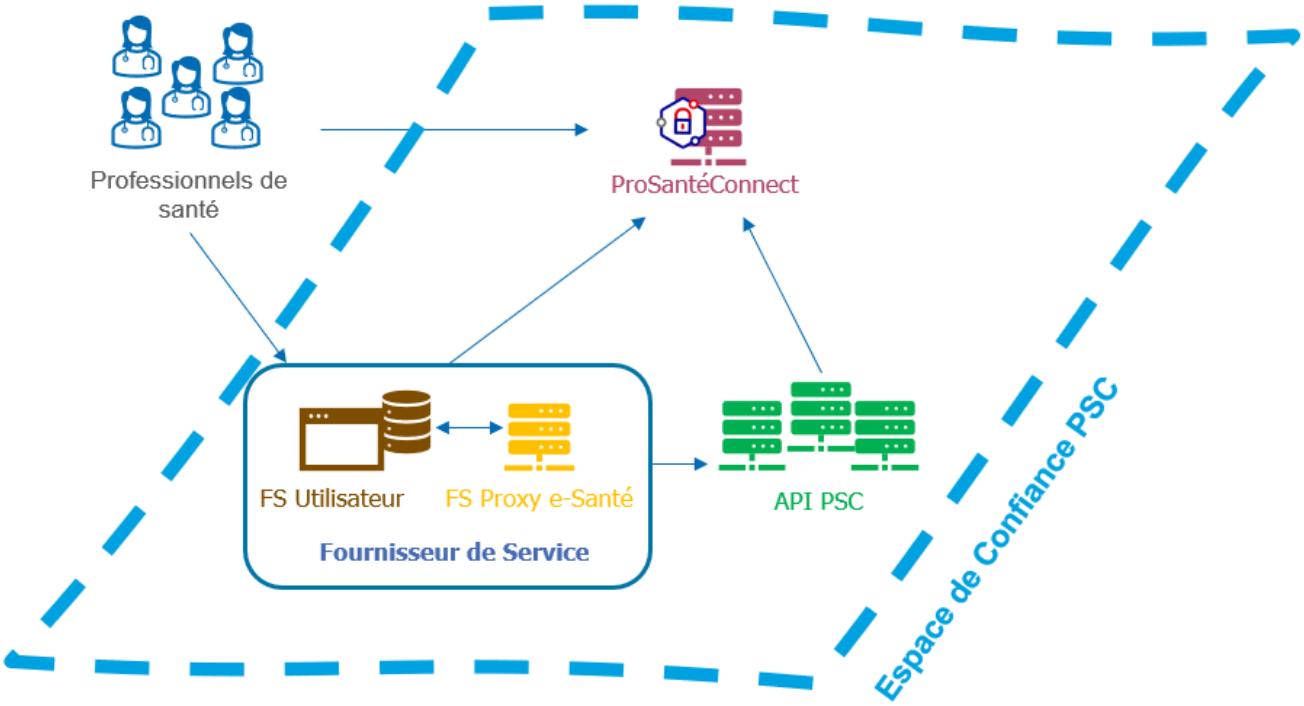
Convention : le texte surligné en jaune correspond aux dernières modifications par rapport à la version précédente du document

3. L'ESPACE DE CONFIANCE DE PRO SANTE CONNECT

L'espace de confiance est un constitué d'un ensemble de composants qui permettent des échanges en toute sécurité aux différents intervenants dans cet espace.

Il est composé de :

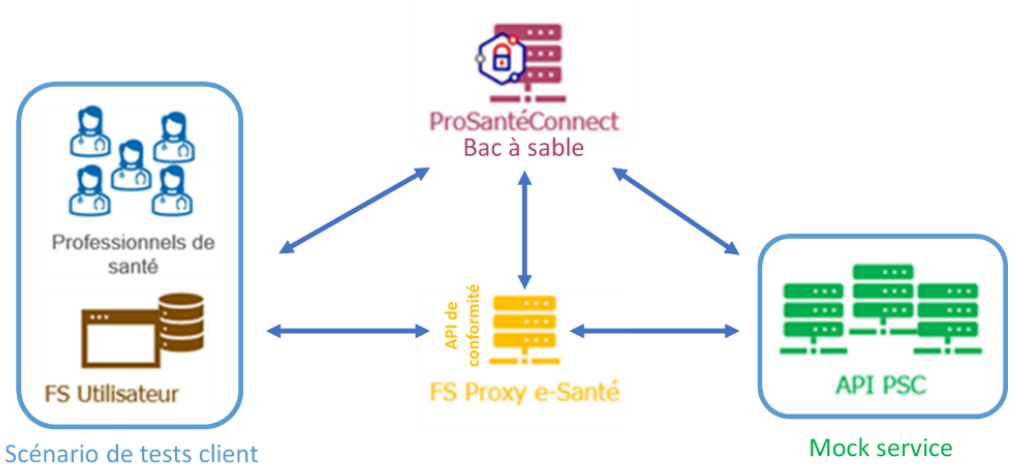
- Un mécanisme d'authentification basé sur OpenID Connect utilisant ClientID + certificat x509
- Une API Crédential, qui liste des clients ID des services qui sont habilités "espace de confiance"
- Des Proxy e-Santé qui garantissent la séparation des contextes et la traçabilité des échanges
- Des API Pro Santé Connectées qui fournissent des données métier uniquement dans cet espace



4. L'OUTIL DE CONFORMITE

Cet outil a pour objectif d'aider les éditeurs de Proxy e-Santé à tester leur système pour s'assurer qu'il répond aux spécifications du référentiel.

Il est disponible dans l'environnement Bac à Sable de Pro Santé Connect.



Il s'articule autour de **3 composants** :

- Le **scénario de Tests client** :
  Ce composant envoie les requêtes prévues par le plan de tests et simule les actions des PS utilisant des logiciels PS (LPS) ou toutes autres solutions impliquant des appels à des API Pro Santé Connectées via un Proxy e-Santé donné.

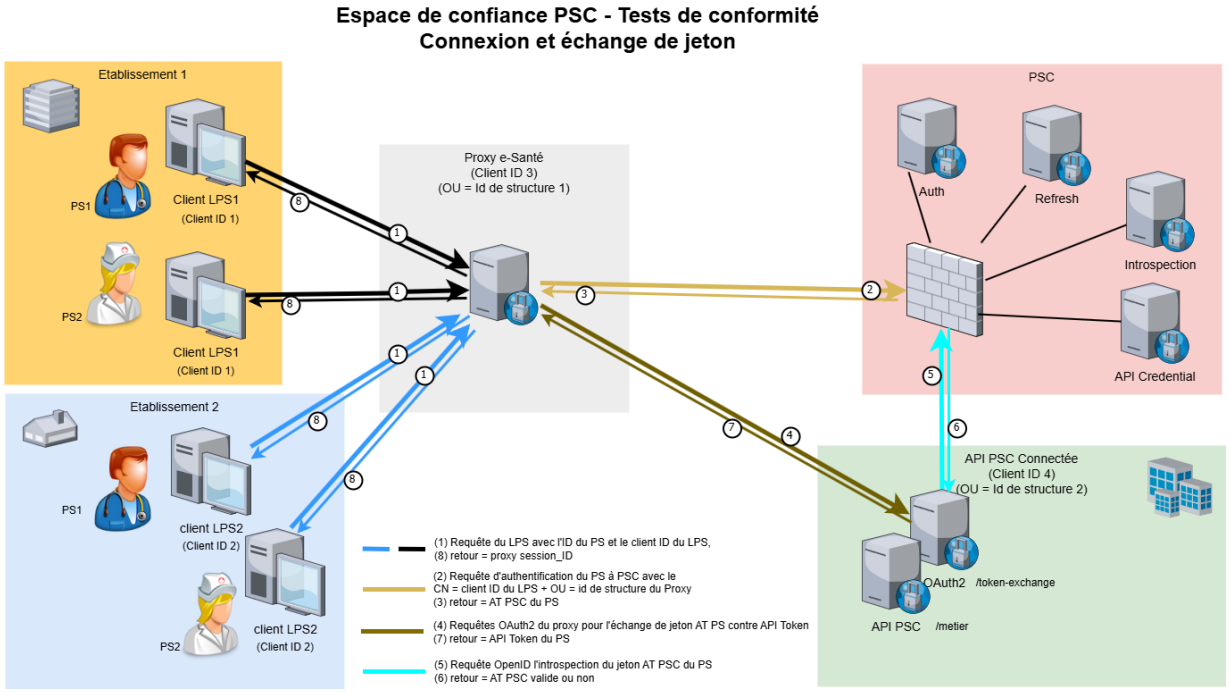
- L'**API de conformité** :
  Ce composant permet d'obtenir par simple requête API, les informations présentes au niveau du Proxy e-Santé afin de valider que la gestion des informations répond aux exigences. Afin de collecter ces informations (contexte des requêtes « id PS, id LPS » et des connexions), une API spécifique aux tests de conformité et imposée par l'ANS. Elle être exposée par le Proxy e-Santé pour permettre :
  - ✓ D'initialiser la connexion entre le PS-LPS et le Proxy e-Santé (endpoint /connect)
  - ✓ D'invalider la session en cours (endpoint /disconnect)
  - ✓ De transmettre certaines requêtes à l'API PSC (endpoint /send)
  - ✓ De retourner les traces stockées au niveau du Proxy e-Santé (endpoint /traces)

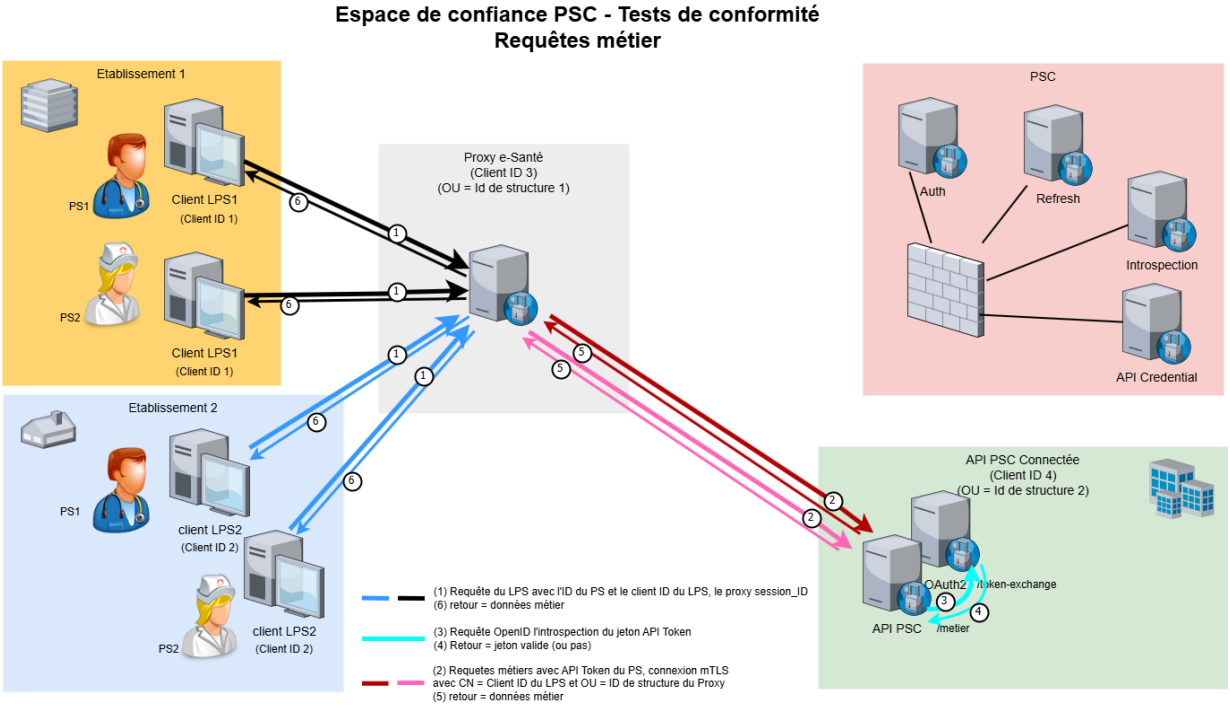
- Le **mock service API PSC**:
  Ce composant simule un fournisseur de données, qui doit répondre aux différents cas de tests du scénario, en retournant des données prévues par le scénario. Pour cela il expose une API Pro Santé Connectée au sens du volet transport du CI-SIS avec une requête OAUTH2 /token-exchange et une requête métier vers les données du service

5. SCENARIOS DE TESTS

Pour les besoins des tests le système impliquera :

- 2 Professionnels de santé de test, ayant chacun activé leur e-CPS respectives
- 2 Logiciels de professionnel de santé (LPS)
- Le Proxy e-Santé à tester
- 1 API Pro Santé Connectée
- L'environnement Pro Santé Connect Bac à Sable





Afin de valider la conformité du Proxy e-Santé aux exigences de l'espace de confiance, la suite de cas métier comprendra les cinq scénarios suivants :

Scénario 1 : Cas Nominal - Connexion PS1 LPS1 + 2 Requêtes API PSC
Scénario 2 : Cas Nominal - PS1 LPS1 et PS2 LPS1+ 2 Requêtes API PSC
Scénario 3 : Cas Nominal - PS1 LPS1 et PS1 LPS2 + 2 Requêtes API PSC
Scénario 4 : Cas Nominal - PS1 LPS1 et PS2 LPS2 + 2 Requêtes API PSC
Scénario 5 : Cas Nominal - PS1 LPS1 +2 Requêtes API PSC + déconnexion

Les 2 requêtes API PSC sont décrites comme suit :
- Une requête OAUTH2 token-exchange sur le serveur d'autorisation de l'API PSC,
- Une requête métier sur le mock service API PSC.

L'outil de test qui est utilisé pour exécuter ces scénarios s'appelle "Platines" c'est un outil de test d'interopérabilité déjà utilisé par le ROR, TOM, et le RASS.

6. PREREQUIS

Comme vu dans le paragraphe précédent avant de pouvoir exécuter les tests de conformité, l'éditeur doit avoir réalisés les points suivants :

- ✓ Disposer de 2 CPS de test ou e-CPS de test (deux téléphones sont nécessaires dans ce cas). Il est possible d'utiliser une CPS de test avec CPS Gestion qui permet les authentification CIBA et une e-CPS de test.
- ✓ Avoir configuré l'outil de test Platines
  - Création de la chaine de confiance
  - Création de l'application « consommateur de données » qui utilise la chaine de confiance
  - Création de l'application « fournisseur de données »
- ✓ Disposer des certificats d'Authentification IGC-SANTE ELEMENTAIRE ORGANISATION de test pour la connexion des PS auprès de PSC bac-à-sable :
  - Certificat 1 : CN=Client ID du LPS 1 (ans-odc-lps1-edc-bas) + OU=id de structure de la carte de test utilisée pour commander le certificat
  - Certificat 2 : CN=Client ID du LPS 2 (ans-odc-lps2-edc-bas) + OU= id de structure de la carte de test utilisée pour commander le certificat
  -
- Les certificats sont utilisés par le proxy e-Santé pour la connexion CIBA des PS 1 et 2 à Pro Santé Connect "au nom" des LPS 1 et 2
- Ces certificats sont également utilisés pour les requêtes SSL entre le Proxy e-Santé et l'API PSC (end point du serveur d'autorisation et end point du mock service).

> ***Remarque : Il n'est pas nécessaire de demander des client_ID auprès de l'équipe PSC pour ces tests, les client_ID existent déjà, ce sont ceux que Platines utilise pour simuler les LPS1 et 2. Par contre votre Proxy e-Santé doit présenter les certificats de ces LPS, dont les client_ID vous sont fournis.***
> ***Le processus de commande de certificats est disponible à cette adresse :***
> ***https://industriels.esante.gouv.fr/produits-et-services/certificats-logiciels***

7. CONFIGURATION DE PLATINES

Un guide de l'utilisateur de Platines est disponible ici, néanmoins il est important de comprendre que l'outil Platines est avant tout un outil de test d'interopérabilité.

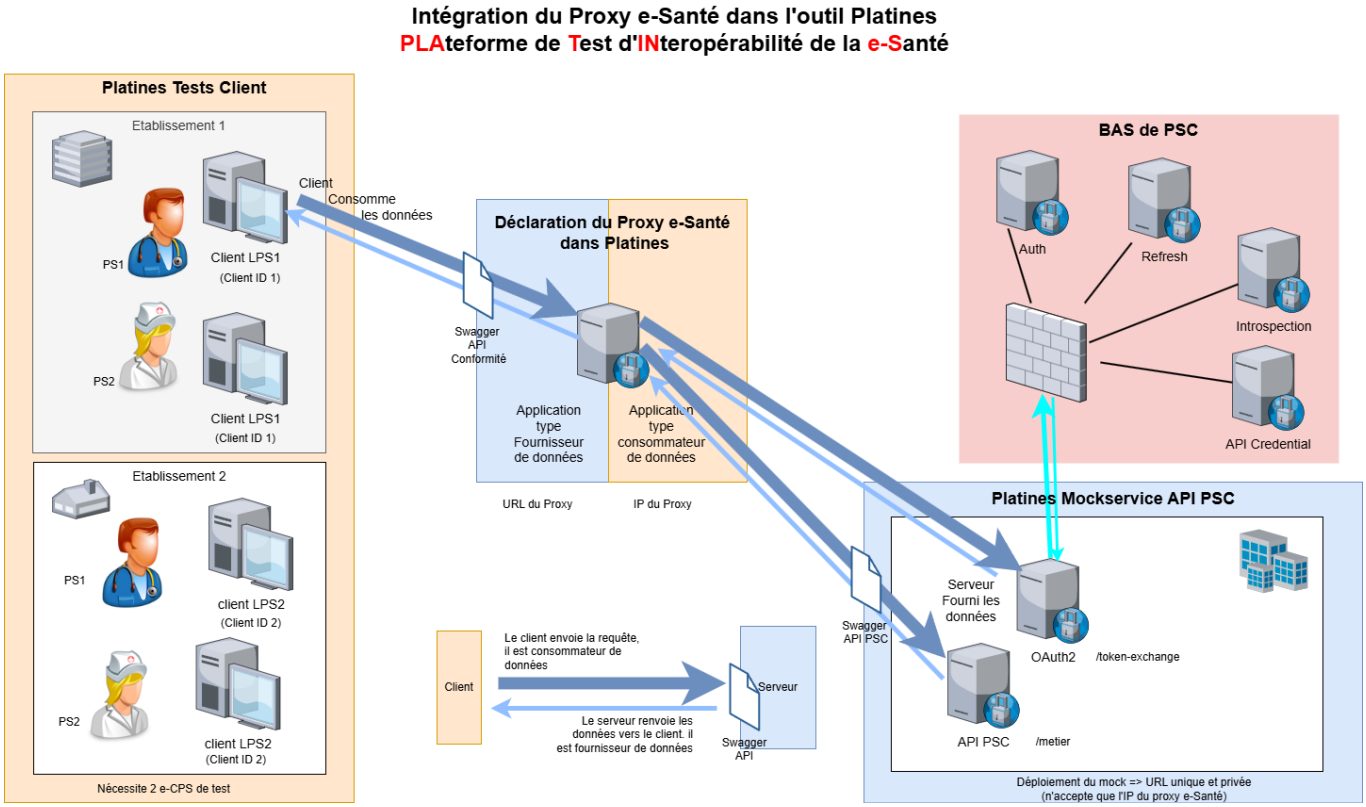
Ce qui signifie qu'il teste soit

- le comportement d'un client qui effectue des requêtes vers un serveur (conformité technique et métier des requêtes)
- le comportement d'un serveur qui envoie des données à un client (conformité technique et métier des données envoyées par le serveur)

Dans le cadre de l'Espace de Confiance, le Proxy e-Santé joue les deux rôles,

Donc il faut configurer deux applications dans l'outil Platines :

- Une application de type "Fournisseur de données" : le coté API de conformité qui renvoie les données au client qui réalise les requêtes du scenario de test. Cette application doit avoir une URL accessible depuis internet. Il s'agit de l'URL du Proxy e-Santé.
- Une application de type "consommateur de données" : le coté du proxy e-Santé qui interroge l'API PSC et qui consomme ses données. Cette application doit présenter une adresse IP qui est enregistrée en liste blanche pour accéder au mock service API PSC.



8. EXECUTION DES TESTS ET RESULTATS

L'exécution des tests s'effectue dans l'ordre suivant :

Dans Platines

- ✓ Déployer le Mock Service API PSC
- ✓ S'assurer que la session est active (durée de session à définir lors du déploiement)

Dans le Proxy e-Santé

- ✓ Enregistrer l'URL du Mock Service dans le Proxy e-Santé (URL reçue par e-mail)
- ✓ Enregistrer l'URL du endpoint d'échange de jeton de l'API PSC dans le Proxy e-Santé https://auth.server.api.edc-psc.esante.gouv.fr/realms/signsessiondata/protocol/openid-connect/token

Dans Platines

- ✓ Déployer le projet des tests API dans Platines

Sur Smarphone

- ✓ Accepter les demandes d'authentifications reçues sur les deux e-CPS de test pendant le déroulement du plan de tests

Sur CPS Gestion

- ✓ Rechercher et Accepter les demandes d'authentifications reçues sur un idNat de carte de test pendant le déroulement du plan de tests si vous avez choisi une mode d'authentification de type CARD pour cet idNat dans la configuration du plan de tests.
- ✓

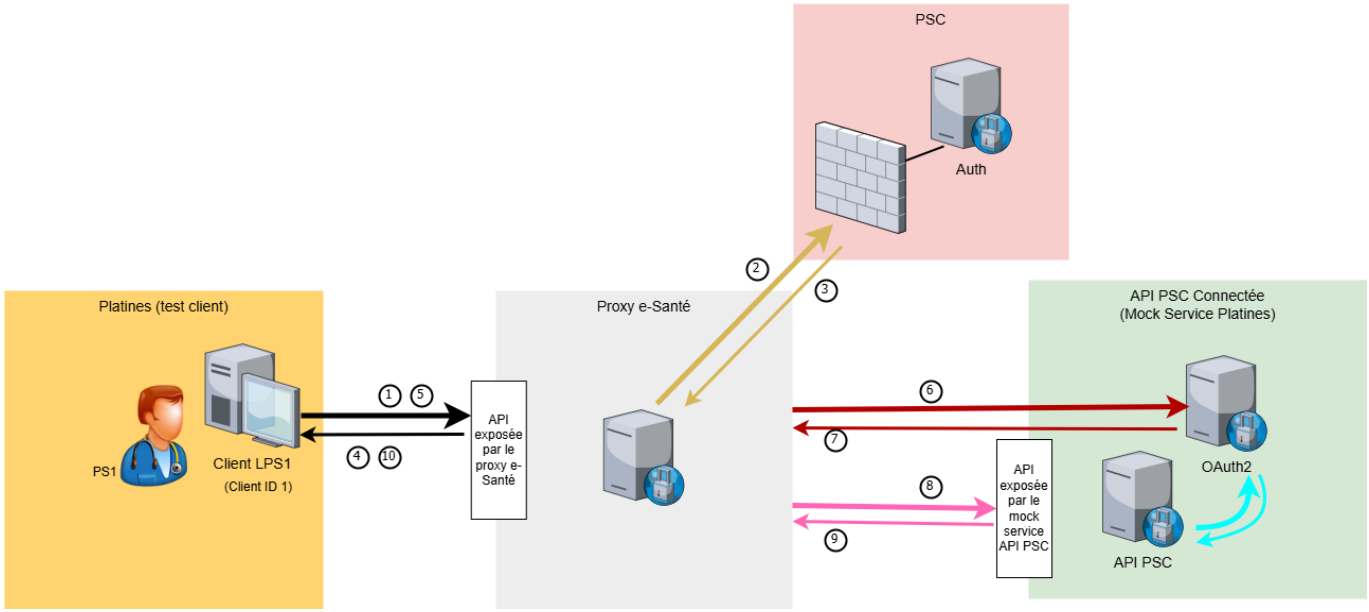
Authentifications attendues lors de l'exécutions du plan de tests

3 authentifications successives sur l'idNat du PS1
1 Authentification sur l'idNat du PS2
3 authentifications successives sur l'idNat du PS1
1 Authentification sur l'idNat du PS2
2 authentifications successives sur l'idNat du PS1

A l'issue des tests, les résultats OK ou KO sont directement visibles dans Platines, et un rapport est téléchargeable depuis la session de test, ainsi que dans les logs téléchargeables. Le fichier html contenu dans le fichier zip téléchargeable peut être utilisé comme preuve de réussite des tests de conformité.

9. CONFIGURATION DU PROXY E-SANTE

Espace de confiance PSC - Configuration du Proxy e-Santé



Le proxy e-Santé doit être configuré pour transmettre correctement les requêtes reçues aux étapes 1 et 5 vers PSC et l'API PSC, étapes 2 et 6 et 8 du schéma ci-dessus.

Vous trouverez sur le Github de l'ANS à l'adresse :

`https://github.com/ansforge/psc-edc-proxy-esante`

- Un Proxy e-Santé exemple
- Le swagger de l'API qui doit être exposée par le Proxy e-Santé : `API-Proxy-eSante.json`
- Le swagger de l'API qui est exposée par l'API PSC (le mockservice Platines) : `API-PSC-tests de conformité.json`

(1) Le proxy e-Santé reçoit la requête de connexion à PSC sur son endpoint `/connect`
(2) Le proxy e-Santé prend en charge la demande d'authentification du PS+LPS auprès de PSC

> ***Attention : La demande d'authentification DOIT :***
> ***Utiliser le flux CIBA***
> ***Demander le scope_all dans l'attribut "scope"***

(3) PSC renvoie l'AT PSC du PS
(4) Le Proxy e-Santé envoie l'id de session au LPS
(5) Le proxy reçoit les requêtes métier sur une URL au format suivant :

`https : // url du proxy / send / service cible / endpoint cible`

Pour les tests Platines, le proxy les reçoit sur :
`https : // url du proxy /send/apipsc/signsessiondata`

(6) Le Proxy e-Santé appelle le endpoint de token-exchange fournie par l'outil Platines : `https://auth.server.api.edc-psc.esante.gouv.fr/realms/signsessiondata/protocol/openid-connect/token`

(7) Le Proxy reçoit le jeton d'API en échange d'un jeton PSC valide

(8) Le proxy doit transmettre les requêtes métier à l'URL du service cible selon le format suivant :

`https : // url du service cible / service cible / endpoint cible`

Ainsi le mockservice API PSC attend les requêtes métier à l'adresse :

`https : // url du mockservice / service cible / endpoint cible`

En pratique les URL attendues sont du type :

`https://e5d38dcb-5fc6-43a6-9994-2ff60e6aacf7.mockservice.platines.esante.gouv.fr/mockservice/apipsc/signsessiondata`

A chaque déploiement d'un mockservice dans l'outil de conformité, vous recevrez un mail avec l'URL du nouveau mockservice, ces URL sont au format : `https://e5d38dcb-5fc6-43a6-9994-2ff60e6aacf7.mockservice.platines.esante.gouv.fr/mockservice` (n'oubliez pas le `/mockservice` dans l'URL)

`e5d38dcb-5fc6-43a6-9994-2ff60e6aacf7` est unique, il change à chaque nouvelle session de test. Vous devrez donc configurer le proxy e-santé pour transmettre les requêtes métier à la nouvelle URL pour chaque nouveau mockservice déployé.

(9) le Proxy e-Santé reçoit la réponse du endpoint `/signsessiondata`

(10) Le Proxy transmet la réponse au LPS à l'origine de la requête 5

Le service cible et le endpoint cible sont : `/apipsc/signsessiondata` il faut donc envoyer la requête métier à l'API du proxy sur le endpoint `/send`, en précisant le service cible et le endpoint cible, donc sur `/send/apipsc/signsessiondata`

> ***Attention : L'URL transmise par le Proxy e-Santé ne contient pas /send***
> ***Exemple :***
> ***https://e5d38dcb-5fc6-43a6-9994-2ff60e6aacf7.mockservice.platines.esante.gouv.fr/mockservice/apipsc/signsessiondata***

10. SUITE DE TESTS

10.1. Scénario 1 : Cas Nominal - Connexion PS1 LPS1 + Requête API PSC

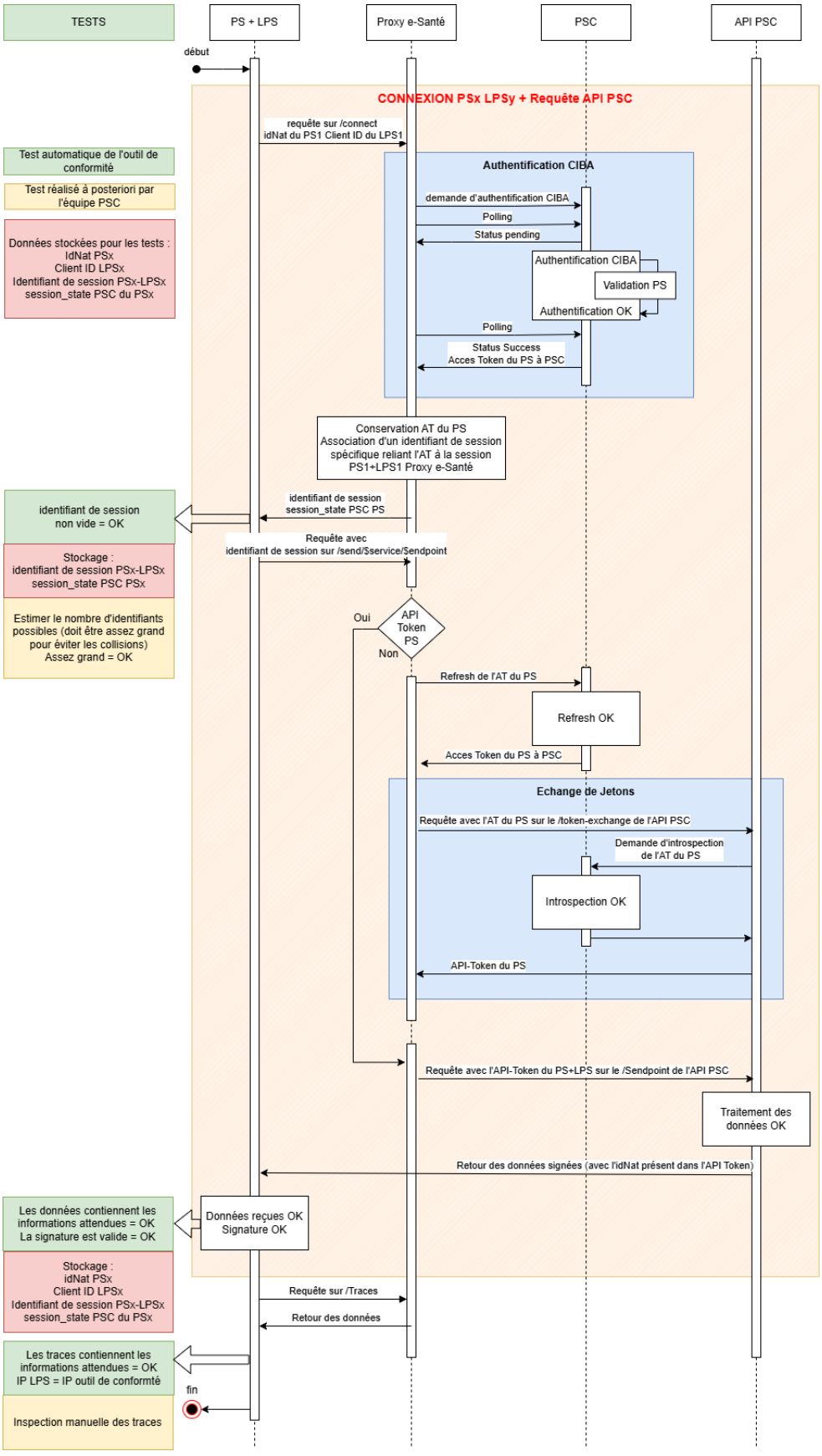
10.1.1. Objectif

Ce scénario de test doit :

- Simuler la connexion d'un seul PS via un LPS au Proxy e-Santé
- Effectuer une requête métier (demande de signature) à l'API PSC
- Effectuer une requête (récupération des traces) au Proxy e-Sante

> ***Remarque : la requête de signature est une fonctionnalité de l'API Pro Santé Connectée. Elle consiste à signer les données qu'elle reçoit sur son endpoint (/signsessiondata). L'outil de conformité vérifie (par calcul à partir des données transmises) que la signature reçue en provenance de l'API PSC correspond à celle attendue, validant ainsi la bonne séparation des contextes à partir des éléments : id PS, id LPS, id de session, token API.***

10.1.2. Diagramme de séquence



10.1.3. Résultats attendus

1) Requête de connexion (passant) :

- Le Proxy e-Santé doit **initier une connexion** du PS1 à PSC au nom du LPS1 (cette connexion doit être en succès pour continuer, avec notamment un contrôle du certificat présenté (AC émettrice, Expiration et Révocation)
- Le Proxy e-Santé doit **retourner :**
  - un **identifiant de session unique** à la connexion « PS1+LPS1-Proxy e-Santé » (relié à l'Access Token PSC du PS au niveau du Proxy e-Santé),
  - ainsi que le « **session_state »** PSC du PS

2) Requête de signature (passant) :

- Le LPS1 envoie via Platines une requête de signature sur le end point /send/apipsc/signsessiondata du proxy e-santé.
- Le proxy e-santé envoie une requête sur le end point /token-exchange du serveur d'autorisation de l'API PSC et obtient un access token API.
- Le serveur d'autorisation de l'API PSC vérifie le jeton PSC (AT) via un appel au endpoint d'introspection de PSC,
- Le proxy e-santé envoie les données du LPS au mock service l'API PSC sur le end point /apipsc/signsessiondata.
- Les données retournées au LPS1 par le mock service de l'API PSC doivent contenir **les données envoyées + une signature** par l'API PSC
- Vérifier que les données signées qui transfèrent par le proxy n'ont pas été altérées

Les requêtes envoyées par le proxy e-santé sur le serveur d'autorisation et sur le mock service de l'API PSC sont décrites à la fin du guide au paragraphe 10.6.

3) Requête de récupération des traces (passant) :

- Appel au endpoint /traces afin de vérifier que le proxy collecte bien les traces demandées (plusieurs formats sont supportés : text/plain, application/json, application/xml, application/zip, application/octet-stream)
- Vérifier le format de la réponse : La pièce jointe avec les traces fournit par le proxy doit contenir l'entête "Content-Disposition" à sa réponse

4) Requête de connexion (non passant) :

- Le Proxy e-Santé doit **initier une double connexion** du PS1 à PSC via le LPS1
- Le Proxy e-Santé doit **retourner :**
  - un **code 304**,

5) Requête de connexion (non passant) :

- Le Proxy e-Santé doit **initier une connexion** du PS1 à PSC via le LPS1 avec **un identifiant de LPS invalide**

- Le Proxy e-Santé doit **retourner :**
  - un **code 404**,

ainsi que le message « **User National ID or Software Client ID Not Found** »

- **6) Requête sur /send mais PS déconnecté :**Le Proxy e-Santé doit **initier une connexion** du PS1 à PSC via le LPS1
- Le Proxy e-Santé doit déconnecter le PS
- A la réception de la requête /send, le Proxy e-Santé doit **retourner :**
  - un **code 401**,
  - ainsi que le message « **No session found** »

10.1.4. Scénario de test

- nationalId = idNat d'une e-CPS de test
- clientId = Identifiant du LPS (ans-odc-lps1-edc-bas)
- proxy_session_id = Identifiant de la session Proxy
- session_state = Identifiant de session PSC

Remarque :

Les informations « A Valoriser » (en vert) sont renseignées :

- Par l'utilisateur technique lors de la création de la session de tests. Exemple : "**channel**" : "${value_authentMode}", prend l'une des deux options : **CARD, MOBILE** selon le choix de l'utilisateur.
- Par l'outil de test en utilisant les informations déjà reçues lors des requêtes précédentes. Exemple : "**session_state**" : "${sessionstate}" prend la valeur reçue par la requête /connect

Nom du test	Modèle d'URL	Réponse attendue

Requête de connexion
 	
(Passant)
	Méthode : POST
	
	Requête :
	https://${BASE_URL}/connect
	
	Json body :
	{
	"**nationalId**" : "${value_nationalId}",
	"**bindingMessage**" : "99",
	"**clientID**" : "${value_clientId}",
	"**channel**" : "${value_authentMode}",
	}
	
	En vert = A valoriser
	En bleu = Nom du endpoint
	
	Exemple :
	https://10.3.8.165:8080/connect
		- Code : 200
		
		- Le corps de la réponse (.json) contient :
		- key "**proxy_session_id**" avec une valeur
		- key "**session_state**" avec une valeur

Requête de signature **(Passant)**	**Méthode** : POST **Requête** :	- Code : 200 - Le corps de la réponse (.json) contient :

	https://${BASE_URL}/send/apipsc/ signsessiondata
	
	Json body :
	{
	"nationalId" : "${value_nationalId}",
	"clientID" : "${value_clientId}",
	"proxy_session_id" : "${value_proxysessionid}",
	"session_state" : "${sessionstate}"
	}
	
	En vert = A valoriser
	En bleu = Nom du endpoint
	
	Exemple :
	https://10.3.8.165:8080/send/apipsc/signsessiondata
	
	Json body :
	{
	"nationalId" : "899700539499",
	"clientID" : " ans-odc-lps1-edc-bas",
	"proxy_session_id" :
	"F24A1675730D0BD9D7191CA78592143C",
	"session_state" : "9ee0fd18-4823-4972-b6c3-847bbb3cbe8a"
	}
		- key " **nationalId**" avec une valeur
		- key "**clientId**" avec une valeur
		- key "**proxy_session_id**" avec une valeur
		- key "**session_state**" avec une valeur
		- key "**signature**" avec une valeur
		
		- Les valeurs des variables (« nationalId », « clientId », « Proxy_session_id », « session_state », « signature ») sont identiques à celles envoyées dans la requête

Requête de récupération des traces **(Passant)**	**Méthode :** GET **Requête** : https://${BASE_URL}/traces?start=${value_start}&end=${value_end} Format de la date : AAAA-MM-JJT00:00:00Z En vert = A valoriser En bleu = Nom du endpoint **Exemple :** https://10.3.8.165:8080/traces?start=2024-10-02T09:00:00Z&end=2024-10-02T11:32:00Z **Paramètres** : - start = début de la période de recherche choisie (Obligatoire) - end = fin de la période de recherche choisie (facultatif)	- Code : 200 - La réponse peut être au format texte ou pièce jointe, et doit contenir les informations suivantes : • client_id • id_nat • session_state • timestamp • source_ip • source_port • proxy_session_id • certificats ○ cn ○ ou • error_code • request

Requête de connexion
 	
(Non Passant) (PS1 déjà connecté au Proxy)
	Méthode : POST
	
	Requête : **Envoyer deux fois la requête de connexion pour le PS1 :**
	https://${BASE_URL}/connect
	
	Json body :
	{
	"nationalId" : "${value_nationalId}",
	"bindingMessage" : "99",
	"clientID" : "${value_clientId}",
	"channel" : "${value_authentMode}",
	}
	
	En vert = A valoriser
	En bleu = Nom du endpoint
	
	Exemple :
	https://10.3.8.165:8080/connect
		- Code : **304**

Requête de connexion	**Méthode :** POST	- Code : **404**

(Non Passant) (Identifiant PS invalide)	**Requête** : **Envoyer une requête de connexion avec un identifiant PS invalide :**
	https://${BASE_URL}/connect
	
	Json body :
	{
	"**nationalId**" : "${value_nationalId}",
	"**bindingMessage**" : "99",
	"**clientID**" : "${value_clientId}",
	"**channel**" : "${value_authentMode}",
	}
	
	En vert = A valoriser+
	En bleu = Nom du endpoint
	
	Exemple :
	Json body :
	{
	"**nationalId**" : "${value_nationalId}",
	"**bindingMessage**" : "99",
	"**clientID**" : "**ans-odc-lps3-edc-bas**",
	"**channel**" : "MOBILE",
	}
		- Un message : "**User National ID or Software Client ID Not Found**"

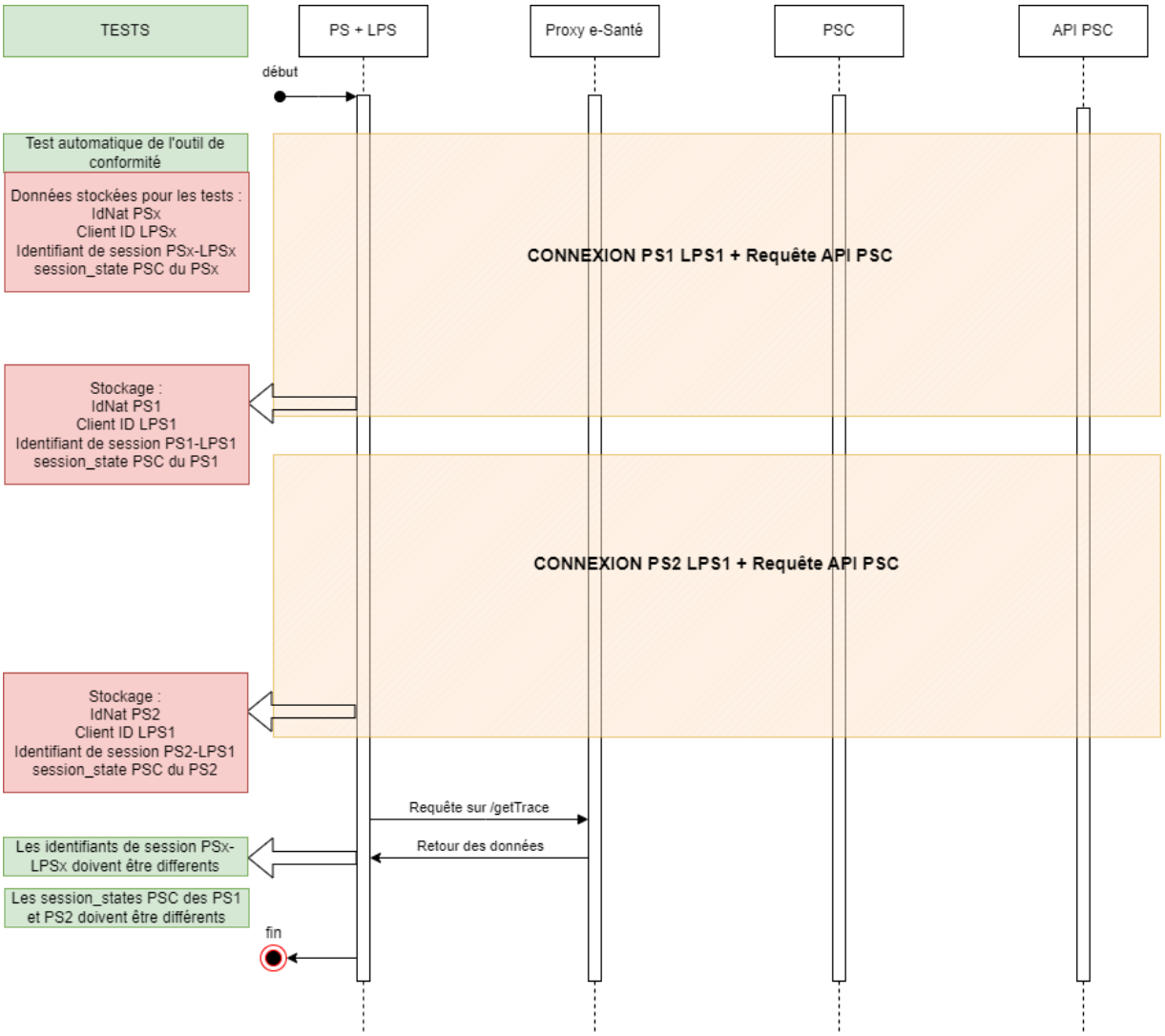
10.2. Scénario 2 : Cas Nominal - Connexion PS1 LPS1 et PS2 LP1 + Requête API PSC

10.2.1. Objectif

Ce scénario de test doit :

- Simuler la connexion du PS1 via le LPS1 au Proxy e-Santé (aucun AT n'est envoyé au Proxy e-Santé)
- Effectuer une requête (1) à l'API PSC Connectée
- Simuler la connexion du PS2 via le LPS1 au Proxy e-Santé (aucun AT n'est envoyé au Proxy e-Santé)
- Effectuer une requête (2) à l'API PSC Connectée
- Récupérer les traces correspondantes

10.2.2. Diagramme de séquence



10.2.3. Résultats attendus

- Le Proxy e-Santé doit initier une connexion du PS1 à PSC (cette connexion doit être en succès pour continuer)
- Le Proxy e-Santé doit retourner un identifiant de session unique à la connexion PS1+LPS1-Proxy e-Santé et relié à l'AT PSC du PS
- Les données retournées doivent contenir les données envoyées + un aléa, le tout signé par l'API PSC
- Le Proxy e-Santé doit initier une connexion du PS2 à PSC (cette connexion doit être en succès pour continuer)
- Le Proxy e-Santé doit retourner un identifiant de session unique à la connexion PS2+LPS1-Proxy e-Santé et relié à l'AT PSC du PS (qui doit être différent de celui du point 2)
- Les données retournées doivent contenir les données envoyées + un aléa, le tout signé par l'API PSC
- Les traces collectées (via le endpoint /traces) doivent contenir les informations attendues

10.2.4. Scénario de test

- nationalId = idNat d'une e-CPS de test

- clientId = Identifiant du LPS (ans-odc-lps1-edc-bas)
- proxy_session_id = Identifiant de la session Proxy
- session_state = Identifiant de session PSC

Nom du test	Modèle d'URL	Réponse attendue

Requête de connexion PS1 LPS1
 	
(Passant)
	Méthode : POST
	
	Requête :
	https://${BASE_URL}/connect
	
	Json body :
	{
	"nationalId" : "${value_nationalId_PS1}",
	"bindingMessage" : "99",
	"clientID" : "${value_clientId}",
	"channel" : "${value_authentMode}",
	}
	
	En vert = A valoriser
	En bleu = Nom du endpoint
	
	Exemple :
	https://10.3.8.165:8080/connect
		- Code : 200
		
		- Le corps de la réponse (.json) contient :
		- key "**proxy_session_id**" avec une valeur
		- key "**session_state**" avec une valeur

Requête de signature
 	
(Passant)
	Méthode : POST
	
	Requête :
	https://${BASE_URL}/send/apipsc/signsessiondata
	
	Json body :
	{
	"nationalId" : "${value_nationalId}",
	"clientID" : "${value_clientId}",
	"proxy_session_id" : "${value_proxysessionid}",
	"session_state" : "${sessionstate}"
	}
	
	En vert = A valoriser
	En bleu = Nom du endpoint
	
	Exemple :
	https://10.3.8.165:8080/send/apipsc/signsessiondata
	
	Json body :
	{
	"nationalId" : "899700539499",
	"clientID" : " ans-odc-lps1-edc-bas",
	"proxy_session_id" :
	"F24A1675730D0BD9D7191CA78592143C",
	"session_state" : "9ee0fd18-4823-4972-b6c3-847bbb3cbe8a"
	}
		- Code : 200
		
		- Le corps de la réponse (.json) contient :
		- key " **nationalId**" avec une valeur
		- key "**clientId**" avec une valeur
		- key "**proxy_session_id**" avec une valeur
		- key "**session_state**" avec une valeur
		- key "**signature**" avec une valeur
		
		- Les valeurs des variables (« nationalId », « clientId », « Proxy_session_id », « session_state », « signature ») sont identiques à celles envoyées dans la requête

Requête de récupération des traces **(Passant)**	**Méthode :** GET **Requête** : https://${BASE_URL}/traces?start=${value_start}&end=${value_end} Format de la date : AAAA-MM-JJT00:00:00Z	- Code : 200 - La réponse peut être au format texte ou pièce jointe, et doit contenir les informations suivantes : • client_id

	En vert = A valoriser En bleu = Nom du endpoint **Exemple :** https://10.3.8.165:8080/traces?start=2024-10-02T09:00:00Z&end=2024-10-02T11:32:00Z **Paramètres** : - start = début de la période de recherche choisie (Obligatoire) - end = fin de la période de recherche choisie (facultatif)	• id_nat • session_state • timestamp • source_ip • source_port • proxy_session_id • certificats ○ cn ○ ou • error_code • request

Requête de connexion PS2 LPS1
 	
(Passant)
	Méthode : POST
	
	Requête :
	https://${BASE_URL}/connect
	
	Json body :
	{
	"nationalId" : "${value_nationalId_PS2}",
	"bindingMessage" : "99",
	"clientID" : "${value_clientId}",
	"channel" : "${value_authentMode}",
	}
	
	En vert = A valoriser
	En bleu = Nom du endpoint
	
	Exemple :
	https://10.3.8.165:8080/connect
		- Code : 200
		
		- Le corps de la réponse (.json) contient :
		- key "**proxy_session_id**" avec une valeur
		- key "**session_state**" avec une valeur

Requête de signature
 	
(Passant)
	Méthode : POST
	
	Requête :
	https://${BASE_URL}/send/apipsc/signsessiondata
	
	Json body :
	{
	"nationalId" : "${value_nationalId}",
	"clientID" : "${value_clientId}",
	"proxy_session_id" : "${value_proxysessionid}",
	"session_state" : "${sessionstate}"
	}
	
	En vert = A valoriser
	En bleu = Nom du endpoint
	
	Exemple :
	https://10.3.8.165:8080/send/apipsc/signsessiondata
	
	Json body :
	{
	"nationalId" : "899700539500",
	"clientID" : " ans-odc-lps1-edc-bas",
	"proxy_session_id" :
	"F24A1675730D0BD9D7191CA78592143C",
	"session_state" : "9ee0fd18-4823-4972-b6c3-847bbb3cbe8a"
	}
		- Code : 200
		
		- Le corps de la réponse (.json) contient :
		- key " **nationalId**" avec une valeur
		- key "**clientId**" avec une valeur
		- key "**proxy_session_id**" avec une valeur
		- key "**session_state**" avec une valeur
		- key "**signature**" avec une valeur
		
		- Les valeurs des variables (« nationalId », « clientId », « Proxy_session_id », « session_state », « signature ») sont identiques à celles envoyées dans la requête

Requête de récupération des traces **(Passant)**	**Méthode :** GET **Requête** : https://${BASE_URL}/traces?start=${value_start}&end=${value_end}	- Code : 200 - La réponse peut être au format texte ou pièce jointe, et doit contenir les informations suivantes :

	Format de la date : AAAA-MM-JJT00:00:00Z En vert = A valoriser En bleu = Nom du endpoint **Exemple :** https://10.3.8.165:8080/traces?start=2024-10-02T09:00:00Z&end=2024-10-02T11:40:00Z **Paramètres** : - start = début de la période de recherche choisie (Obligatoire) - end = fin de la période de recherche choisie (facultatif)	• client_id • id_nat • session_state • timestamp • source_ip • source_port • proxy_session_id • certificats ○ cn ○ ou • error_code request
Déconnexion PS1 & PS2	**Méthode :** DELETE **Requête** : https://${BASE_URL}/disconnect **Http header :** Cokkie : proxy_session_id=${proxy_session_id_value} En vert = A valoriser En bleu = Nom du endpoint **Exemple :** https://10.3.8.165:8080/disconnect	- Code : 200 - 'proxy_session_id' de PS1 et différent que celui de PS2 - 'session_state' value de PS1 et différent que celui de PS2

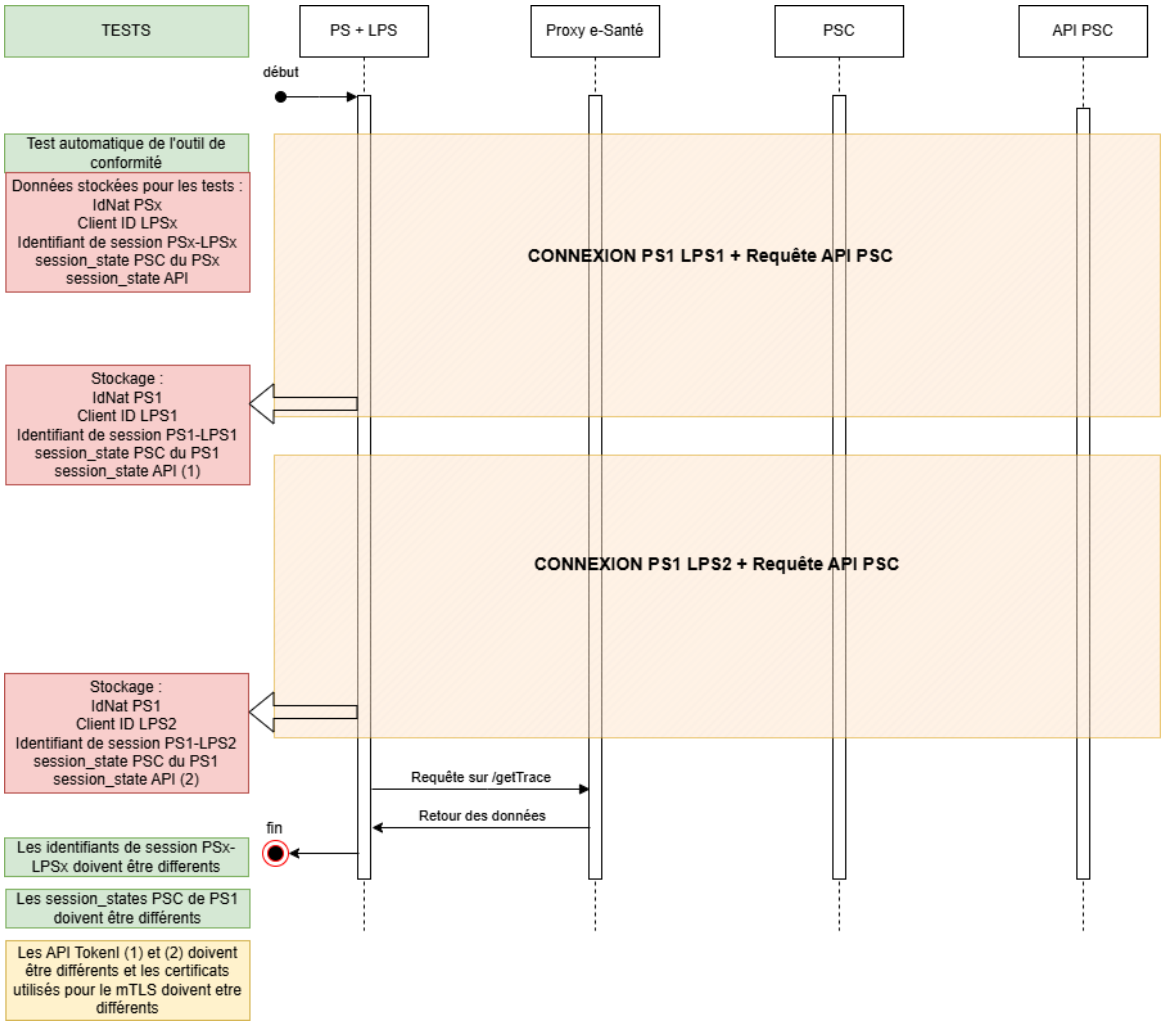
10.3. Scénario 3 : Cas Nominal - Connexion PS1 LPS1 et PS1 LP2 + Requête API PSC

10.3.1. Objectif

Ce scénario de test doit :

- Simuler la connexion du PS1 via le LPS1 au Proxy e-Santé (aucun AT n'est envoyé au Proxy e-Santé)
- Effectuer une requête (1) à l'API PSC Connectée
- Simuler la connexion du PS1 via le LPS2 au Proxy e-Santé (aucun AT n'est envoyé au Proxy e-Santé)
- Effectuer une requête (2) à l'API PSC Connectée
- Récupérer les traces correspondantes

10.3.2. Diagramme de séquence



10.3.3. Résultats attendus

- Le Proxy e-Santé doit initier une connexion du PS1 à PSC (cette connexion doit être en succès pour continuer)
- Le Proxy e-Santé doit retourner un identifiant de session unique à la connexion PS1+LPS1-Proxy e-Santé et relié à l'AT PSC du PS
- Les données retournées doivent contenir les données envoyées + un aléa, le tout signé par l'API PSC
- Le Proxy e-Santé doit initier une connexion du PS1 à PSC (cette connexion doit être en succès pour continuer)
- Le Proxy e-Santé doit retourner un identifiant de session unique à la connexion PS1+LPS2-Proxy e-Santé et relié à l'AT PSC du PS (qui doit être différent de celui du point 2)
- Les données retournées doivent contenir les données envoyées + un aléa, le tout signé par l'API PSC
- Les traces collectées (via le endpoint /traces) doivent contenir les informations attendues

10.3.4. Scénario de test

- nationalId = idNat d'une e-CPS de test
- clientId = Identifiant du LPS ("ans-odc-lps1-edc-bas" ou "ans-odc-lps2-edc-bas")
- proxy_session_id = Identifiant de la session Proxy

- session_state = Identifiant de session PSC

Nom du test	Modèle d'URL	Réponse attendue

Requête de connexion PS1 LPS1
 	
(Passant)
	Méthode : POST
	
	Requête :
	https://${BASE_URL}/connect
	
	Json body :
	{
	"**nationalId**" : "${value_nationalId_PS1}",
	"**bindingMessage**" : "99",
	"**clientID**" : "${value_clientId_**lps1**}",
	"**channel**" : "${value_authentMode}",
	}
	
	En vert = A valoriser
	En bleu = Nom du endpoint
	
	Exemple :
	https://10.3.8.165:8080/connect
		- Code : 200
		
		- Le corps de la réponse (.json) contient :
		- key "**proxy_session_id**" avec une valeur
		- key "**session_state**" avec une valeur

Requête de signature
 	
(Passant)
	Méthode : POST
	
	Requête :
	https://${BASE_URL}/send/apipsc/signsessiondata
	
	Json body :
	{
	"**nationalId**" : "${value_nationalId_PS1}",
	"**clientID**" : "${value_clientId_**lps1**}",
	"**proxy_session_id**" : "${value_proxysessionid}",
	"**session_state**" : "${sessionstate}"
	}
	
	En vert = A valoriser
	En bleu = Nom du endpoint
	
	Exemple :
	https://10.3.8.165:8080/send/apipsc/signsessiondata
	
	Json body :
	{
	"**nationalId**" : "899700539499",
	"**clientID**" : " ans-odc-lps1-edc-bas",
	"**proxy_session_id**" :
	"F24A1675730D0BD9D7191CA78592143C",
	"**session_state**" : "9ee0fd18-4823-4972-b6c3-847bbb3cbe8a"
	}
		- Code : 200
		
		- Le corps de la réponse (.json) contient :
		- key " **nationalId**" avec une valeur
		- key "**clientId**" avec une valeur
		- key "**proxy_session_id**" avec une valeur
		- key "**session_state**" avec une valeur
		- key "**signature**" avec une valeur
		
		- Les valeurs des variables (« nationalId », « clientId », « Proxy_session_id », « session_state », « signature ») sont identiques à celles envoyées dans la requête

Requête de récupération des traces **(Passant)**	**Méthode :** GET **Requête** : https://${BASE_URL}/traces?start=${value_start}&end=${value_end} Format de la date : AAAA-MM-JJT00:00:00Z En vert = A valoriser En bleu = Nom du endpoint	- Code : 200 - La réponse peut être au format texte ou pièce jointe, et doit contenir les informations suivantes : • client_id • id_nat • session_state • timestamp • source_ip

	Exemple : https://10.3.8.165:8080/traces?start=2024-10-02T09:00:00Z&end=2024-10-02T11:32:00Z **Paramètres** : - start = début de la période de recherche choisie (Obligatoire) - end = fin de la période de recherche choisie (facultatif)	• source_port • proxy_session_id • certificats ○ cn ○ ou • error_code • request

Requête de connexion PS1 LPS2
 	
(Passant)
	Méthode : POST
	
	Requête :
	https://${BASE_URL}/connect
	
	Json body :
	{
	"nationalId" : "${value_nationalId_PS1}",
	"bindingMessage" : "99",
	"clientID" : "${value_clientId_LPS2}",
	"channel" : "${value_authentMode}",
	}
	
	En vert = A valoriser
	En bleu = Nom du endpoint
	
	Exemple :
	https://10.3.8.165:8080/connect
		- Code : 200
		
		- Le corps de la réponse (.json) contient :
		- key "**proxy_session_id**" avec une valeur
		- key "**session_state**" avec une valeur

Requête de signature
 	
(Passant)
	Méthode : POST
	
	Requête :
	https://${BASE_URL}/send/apipsc/signsessiondata
	
	Json body :
	{
	"nationalId" : "${value_nationalId_PS1}",
	"clientID" : "${value_clientId_IPS2}",
	"proxy_session_id" : "${value_proxysessionid}",
	"session_state" : "${sessionstate}"
	}
	
	En vert = A valoriser
	En bleu = Nom du endpoint
	
	Exemple :
	https://10.3.8.165:8080/send/apipsc/signsessiondata
	
	Json body :
	{
	"nationalId" : "899700539500",
	"clientID" : " ans-odc-lps2-edc-bas",
	"proxy_session_id" : "dc3d6338-a841-4ccc-9a20-1774e09185bc",
	"session_state" : "b6e716e1-00b7-41ce-a62d-0f653cd57471"
	}
		- Code : 200
		
		- Le corps de la réponse (.json) contient :
		- key " **nationalId**" avec une valeur
		- key "**clientId**" avec une valeur
		- key "**proxy_session_id**" avec une valeur
		- key "**session_state**" avec une valeur
		- key "**signature**" avec une valeur
		
		- Les valeurs des variables (« nationalId », « clientId », « Proxy_session_id », « session_state », « signature ») sont identiques à celles envoyées dans la requête

Requête de récupération des traces **(Passant)**	**Méthode :** GET **Requête** : https://${BASE_URL}/traces?start=${value_start}&end=${value_end} Format de la date : AAAA-MM-JJT00:00:00Z En vert = A valoriser En bleu = Nom du endpoint	- Code : 200 - La réponse peut être au format texte ou pièce jointe, et doit contenir les informations suivantes : • client_id • id_nat • session_state • timestamp

	Exemple : https://10.3.8.165:8080/traces?start=2024-10-02T09:00:00Z&end=2024-10-02T11:40:00Z **Paramètres** : - start = début de la période de recherche choisie (Obligatoire) - end = fin de la période de recherche choisie (facultatif)	• source_ip • source_port • proxy_session_id • certificats ○ cn ○ ou • error_code request
Déconnexion PS1 LPS1 & PS1 LPS2	**Méthode :** DELETE **Requête** : https://${BASE_URL}/disconnect **Http header :** Cokkie : proxy_session_id=${proxy_session_id} En vert = A valoriser En bleu = Nom du endpoint **Exemple :** https://10.3.8.165:8080/disconnect	- Code : 200 - 'proxy_session_id' de PS1 LPS1 et différent que celui de PS1 LPS2 - 'session_state' value de PS1 LPS1 et différent que celui de PS1 LPS2

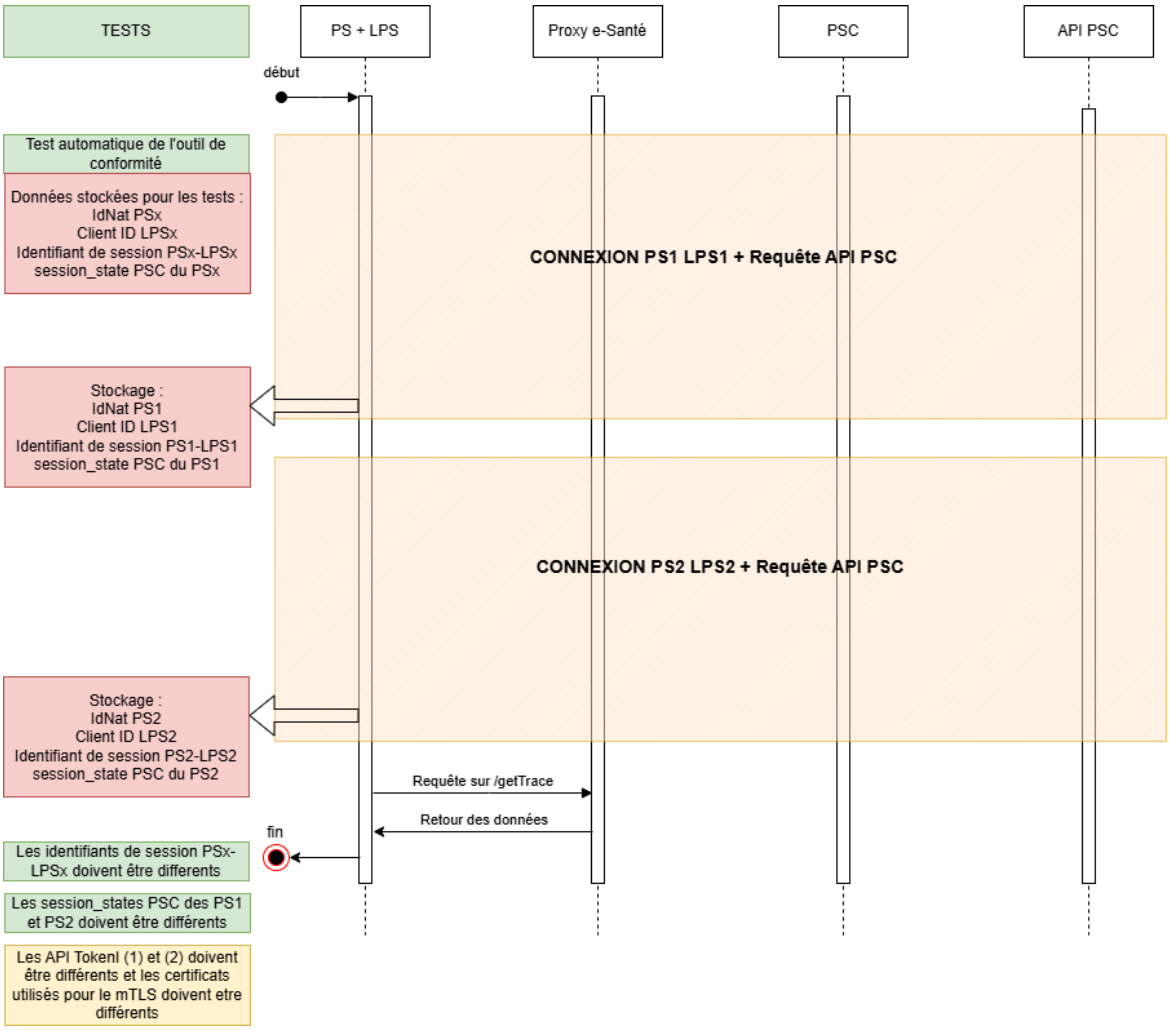
10.4. Scénario 4 : Cas Nominal - Connexion PS1 LPS1 et PS2 LP2 + Requête API PSC

10.4.1. Objectif

Ce scénario de test doit :

- Simuler la connexion du PS1 via le LPS1 au Proxy e-Santé (aucun AT n'est envoyé au Proxy e-Santé)
- Effectuer une requête (1) à l'API PSC Connectée
- Simuler la connexion du PS2 via le LPS2 au Proxy e-Santé (aucun AT n'est envoyé au Proxy e-Santé)
- Effectuer une requête (2) à l'API PSC Connectée
- Récupérer les traces correspondantes

10.4.2. Diagramme de séquence



10.4.3. Résultats attendus

- Le Proxy e-Santé doit initier une connexion du PS1 à PSC (cette connexion doit être en succès pour continuer)
- Le Proxy e-Santé doit retourner un identifiant de session unique à la connexion PS1+LPS1-Proxy e-Santé et relié à l'AT PSC du PS
- Les données retournées doivent contenir les données envoyées + un aléa, le tout signé par l'API PSC
- Le Proxy e-Santé doit initier une connexion du PS2 à PSC (cette connexion doit être en succès pour continuer)
- Le Proxy e-Santé doit retourner un identifiant de session unique à la connexion PS2+LPS2-Proxy e-Santé et relié à l'AT PSC du PS (qui doit être différent de celui du point 2)
- Les données retournées doivent contenir les données envoyées + un aléa, le tout signé par l'API PSC
- Les traces collectées (via le endpoint /traces) doivent contenir les informations attendues

10.4.4. Scénario de test

- nationalId = idNat d'une e-CPS de test
- clientId = Identifiant du LPS ("ans-odc-lps1-edc-bas" ou "ans-odc-lps2-edc-bas")
- proxy_session_id = Identifiant de la session Proxy

- session_state = Identifiant de session PSC

Nom du test	Modèle d'URL	Réponse attendue

Requête de connexion PS1 LPS1
 	
(Passant)
	Méthode : POST
	
	Requête :
	https://${BASE_URL}/connect
	
	Json body :
	{
	"**nationalId**" : "${value_nationalId_PS1}",
	"**bindingMessage**" : "99",
	"**clientID**" : "${value_clientId_**lps1**}",
	"**channel**" : "${value_authentMode}",
	}
	
	En vert = A valoriser
	En bleu = Nom du endpoint
	
	Exemple :
	https://10.3.8.165:8080/connect
		- Code : 200
		
		- Le corps de la réponse (.json) contient :
		- key "**proxy_session_id**" avec une valeur
		- key "**session_state**" avec une valeur

Requête de signature
 	
(Passant)
	Méthode : POST
	
	Requête :
	https://${BASE_URL}/send/apipsc/signsessiondata
	
	Json body :
	{
	"**nationalId**" : "${value_nationalId_PS1}",
	"**clientID**" : "${value_clientId_**lps1**}",
	"**proxy_session_id**" : "${value_proxysessionid}",
	"**session_state**" : "${sessionstate}"
	}
	
	En vert = A valoriser
	En bleu = Nom du endpoint
	
	Exemple :
	https://10.3.8.165:8080/send/apipsc/signsessiondata
	
	Json body :
	{
	"**nationalId**" : "899700539499",
	"**clientID**" : " ans-odc-lps1-edc-bas",
	"**proxy_session_id**" :
	"F24A1675730D0BD9D7191CA78592143C",
	"**session_state**" : "9ee0fd18-4823-4972-b6c3-847bbb3cbe8a"
	}
		- Code : 200
		
		- Le corps de la réponse (.json) contient :
		- key " **nationalId**" avec une valeur
		- key "**clientId**" avec une valeur
		- key "**proxy_session_id**" avec une valeur
		- key "**session_state**" avec une valeur
		- key "**signature**" avec une valeur
		
		- Les valeurs des variables (« nationalId », « clientId », « Proxy_session_id », « session_state », « signature ») sont identiques à celles envoyées dans la requête

Requête de récupération des traces **(Passant)**	**Méthode :** GET **Requête** : https://${BASE_URL}/traces?start=${value_start}&end=${value_end} Format de la date : AAAA-MM-JJT00:00:00Z En vert = A valoriser En bleu = Nom du endpoint	- Code : 200 - La réponse peut être au format texte ou pièce jointe, et doit contenir les informations suivantes : • client_id • id_nat • session_state • timestamp • source_ip

	Exemple : https://10.3.8.165:8080/traces?start=2024-10-02T09:00:00Z&end=2024-10-02T11:32:00Z **Paramètres** : - start = début de la période de recherche choisie (Obligatoire) - end = fin de la période de recherche choisie (facultatif)	• source_port • proxy_session_id • certificats ○ cn ○ ou • error_code • request

Requête de connexion PS2 LPS2
 	
(Passant)
	Méthode : POST
	
	Requête :
	https://${BASE_URL}/connect
	
	Json body :
	{
	"nationalId" : "${value_nationalId_PS2}",
	"bindingMessage" : "99",
	"clientID" : "${value_clientId_LPS2}",
	"channel" : "${value_authentMode}",
	}
	
	En vert = A valoriser
	En bleu = Nom du endpoint
	
	Exemple :
	https://10.3.8.165:8080/connect
		- Code : 200
		
		- Le corps de la réponse (.json) contient :
		- key "**proxy_session_id**" avec une valeur
		- key "**session_state**" avec une valeur

Requête de signature
 	
(Passant)
	Méthode : POST
	
	Requête :
	https://${BASE_URL}/send/apipsc/signsessiondata
	
	Json body :
	{
	"nationalId" : "${value_nationalId_PS2}",
	"clientID" : "${value_clientId_IPS2}",
	"proxy_session_id" : "${value_proxysessionid}",
	"session_state" : "${sessionstate}"
	}
	
	En vert = A valoriser
	En bleu = Nom du endpoint
	
	Exemple :
	https://10.3.8.165:8080/send/apipsc/signsessiondata
	
	Json body :
	{
	"nationalId" : "899700539500",
	"clientID" : " ans-odc-lps2-edc-bas",
	"proxy_session_id" : "dc3d6338-a841-4ccc-9a20-1774e09185bc",
	"session_state" : "b6e716e1-00b7-41ce-a62d-0f653cd57471"
	}
		- Code : 200
		
		- Le corps de la réponse (.json) contient :
		- key " **nationalId**" avec une valeur
		- key "**clientId**" avec une valeur
		- key "**proxy_session_id**" avec une valeur
		- key "**session_state**" avec une valeur
		- key "**signature**" avec une valeur
		
		- Les valeurs des variables (« nationalId », « clientId », « Proxy_session_id », « session_state », « signature ») sont identiques à celles envoyées dans la requête

Requête de récupération des traces **(Passant)**	**Méthode :** GET **Requête** : https://${BASE_URL}/traces?start=${value_start}&end=${value_end} Format de la date : AAAA-MM-JJT00:00:00Z En vert = A valoriser En bleu = Nom du endpoint	- Code : 200 - La réponse peut être au format texte ou pièce jointe, et doit contenir les informations suivantes : • client_id • id_nat • session_state • timestamp

	Exemple : https://10.3.8.165:8080/traces?start=2024-10-02T09:00:00Z&end=2024-10-02T11:40:00Z **Paramètres** : - start = début de la période de recherche choisie (Obligatoire) - end = fin de la période de recherche choisie (facultatif)	• source_ip • source_port • proxy_session_id • certificats ○ cn ○ ou • error_code request
Déconnexion PS1 LPS1 & PS2 LPS2	**Méthode :** DELETE **Requête** : https://${BASE_URL}/disconnect **Http header :** Cokkie : proxy_session_id=${proxy_session_id} En vert = A valoriser En bleu = Nom du endpoint **Exemple :** https://10.3.8.165:8080/disconnect	- Code : 200 - 'proxy_session_id' de PS1 LPS1 et différent que celui de PS2 LPS2 - 'session_state' value de PS1 LPS1 et différent que celui de PS2 LPS2

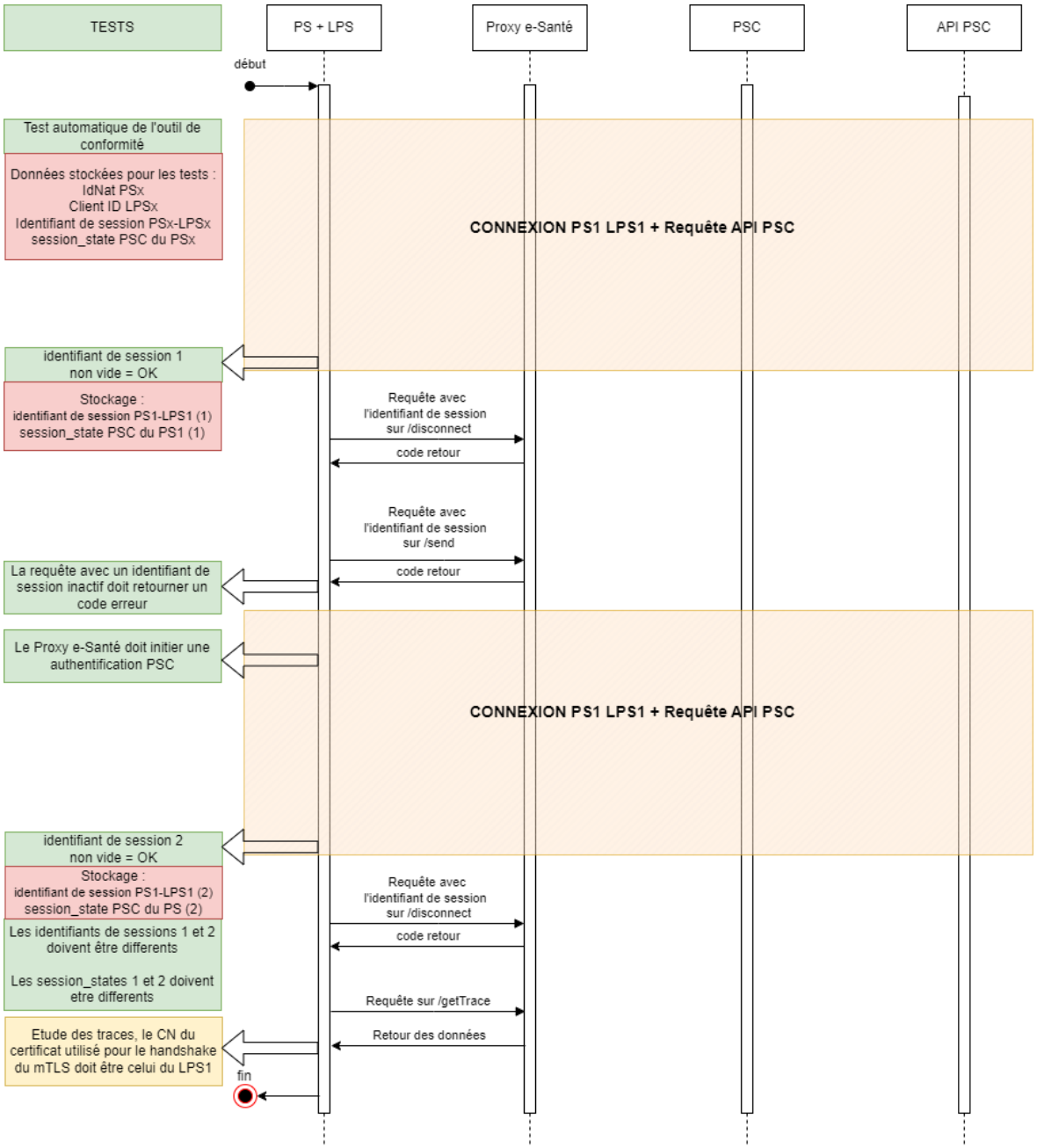
10.5. Scénario 5 : Cas Nominal - Connexion PS1 LPS1 + Requête API PSC + déconnexion

10.5.1. Objectif

Ce scénario de test doit :

- Simuler la connexion du PS1 via le LPS1 au Proxy e-Santé (aucun AT n'est envoyé au Proxy e-Santé)
- Effectuer une requête (1) à l'API PSC Connectée
- Déconnecter le PS1 sur le endpoint /disconnect (désactiver de la session PS1+LPS1)
- Effectuer une requête (1) à l'API PSC Connectée (identique à celle du point 2)
- Simuler la connexion du PS1 via le LPS1 au Proxy e-Santé (aucun AT n'est envoyé au Proxy e-Santé)
- Effectuer une requête (2) à l'API PSC Connectée
- Récupérer les traces correspondantes

10.5.2. Diagramme de séquence



10.5.3. Résultats attendus

- Le Proxy e-Santé doit initier une connexion du PS1 à PSC (cette connexion doit être en succès pour continuer)
- Le Proxy e-Santé doit retourner un identifiant de session unique à la connexion PS1+LPS1-Proxy e-Santé et relié à l'AT PSC du PS
- Les données retournées doivent contenir les données envoyées + un aléa, le tout signé par l'API PSC
- Le PS1 + LPS1 est déconnecté du Proxy e-Santé
- L'identifiant de session doit être inactif
- La requête envoyée sur le endpoint /send avec l'id de session doit retourner un code erreur
- Le Proxy e-Santé doit initier une connexion du PS1 à PSC (cette connexion doit être en succès pour continuer)

- Le Proxy e-Santé doit retourner un identifiant de session unique à la connexion PS1+LPS1-Proxy e-Santé et relié à l'AT PSC du PS (qui doit être différent de celui du point 2)
- Les données retournées doivent contenir les données envoyées + un aléa, le tout signé par l'API PSC
- Les traces collectées (via le endpoint /traces) doivent contenir les informations attendues
- Les identifiants de session PS1 + LPS des points 2 et 8 doivent être différents

10.5.4. Scénario de test

- nationalId = idNat d'une e-CPS de test
- clientId = Identifiant du LPS (ans-odc-lps1-edc-bas)
- proxy_session_id = Identifiant de la session Proxy
- session_state = Identifiant de session PSC

Nom du test	Modèle d'URL	Réponse attendue

Requête de connexion PS1 LPS1
 	
(Passant)
	Méthode : POST
	
	Requête :
	https://${BASE_URL}/connect
	
	Json body :
	{
	"**nationalId**" : "${value_nationalId_PS1}",
	"**bindingMessage**" : "99",
	"**clientID**" : "${value_clientId_**lps1**}",
	"**channel**" : "${value_authentMode}",
	}
	
	En vert = A valoriser
	En bleu = Nom du endpoint
	
	Exemple :
	https://10.3.8.165:8080/connect
		- Code : 200
		
		- Le corps de la réponse (.json) contient :
		- key "**proxy_session_id**" avec une valeur
		- key "**session_state**" avec une valeur

Requête de signature
 	
(Passant)
	Méthode : POST
	
	Requête :
	https://${BASE_URL}/send/apipsc/signsessiondata
	
	Json body :
	{
	"**nationalId**" : "${value_nationalId_PS1}",
	"**clientID**" : "${value_clientId_**lps1**}",
	"**proxy_session_id**" : "${value_proxysessionid}",
	"**session_state**" : "${sessionstate}"
	}
	
	En vert = A valoriser
	En bleu = Nom du endpoint
	
	Exemple :
	https://10.3.8.165:8080/send/apipsc/signsessiondata
	
	Json body :
	{
	"**nationalId**" : "899700539499",
	"**clientID**" : " ans-odc-lps1-edc-bas",
		- Code : 200
		
		- Le corps de la réponse (.json) contient :
		- key " **nationalId**" avec une valeur
		- key "**clientId**" avec une valeur
		- key "**proxy_session_id**" avec une valeur
		- key "**session_state**" avec une valeur
		- key "**signature**" avec une valeur
		
		- Les valeurs des variables (« nationalId », « clientId », « Proxy_session_id », « session_state », « signature ») sont identiques à celles envoyées dans la requête

	"**proxy_session_id**" : "F24A1675730D0BD9D7191CA78592143C", "**session_state**" : "9ee0fd18-4823-4972-b6c3-847bbb3cbe8a" }	
Déconnexion PS1 LPS1	**Méthode :** DELETE **Requête** : https://${BASE_URL}/disconnect **Http header :** Cokkie : proxy_session_id=${proxy_session_id} En vert = A valoriser En bleu = Nom du endpoint **Exemple :** https://10.3.8.165:8080/disconnect	- Code : 200

Requête de signature
 	
(Non Passant)
	Méthode : POST
	
	Requête :
	https://${BASE_URL}/send/apipsc/signsessiondata
	
	Json body :
	{
	"**nationalId**" : "${value_nationalId_PS1}",
	"**clientID**" : "${value_clientId_**lps1**}",
	"**proxy_session_id**" : "${value_proxysessionid}",
	"**session_state**" : "${sessionstate}"
	}
	
	En vert = A valoriser
	En bleu = Nom du endpoint
	
	Exemple :
	https://10.3.8.165:8080/send/apipsc/signsessiondata
	
	Json body :
	{
	"**nationalId**" : "899700539499",
	"**clientID**" : " ans-odc-lps1-edc-bas",
	"**proxy_session_id**" :
	"F24A1675730D0BD9D7191CA78592143C",
	"**session_state**" : "9ee0fd18-4823-4972-b6c3-847bbb3cbe8a"
	}
		- Code : 401
		
		- Un message : "**User National ID or Software Client ID Not Found**"

Requête de récupération des traces **(Passant)**	**Méthode :** GET **Requête** : https://${BASE_URL}/traces?start=${value_start}&end=${value_end} Format de la date : AAAA-MM-JJT00:00:00Z En vert = A valoriser En bleu = Nom du endpoint **Exemple :** https://10.3.8.165:8080/traces?start=2024-10-02T09:00:00Z&end=2024-10-02T11:32:00Z **Paramètres** : - start = début de la période de recherche choisie (Obligatoire) - end = fin de la période de recherche choisie (facultatif)	- Code : 200 - La réponse peut être au format texte ou pièce jointe, et doit contenir les informations suivantes : • client_id • id_nat • session_state • timestamp • source_ip • source_port • proxy_session_id • certificats ○ cn ○ ou • error_code • request

Requête de connexion PS1 LPS1
 	
(Passant)
	Méthode : POST
	
	Requête :
	https://${BASE_URL}/connect
	
	Json body :
	{
	"nationalId" : "${value_nationalId_PS1}",
	"bindingMessage" : "99",
	"clientID" : "${value_clientId_LPS1}",
	"channel" : "${value_authentMode}",
	}
	
	En vert = A valoriser
	En bleu = Nom du endpoint
	
	Exemple :
	https://10.3.8.165:8080/connect
		- Code : 200
		
		- Le corps de la réponse (.json) contient :
		- key "**proxy_session_id**" avec une valeur. Cette valeur est différente que la première requête de « connexion »
		- key "**session_state**" avec une valeur. Cette valeur est différente que la première requête de « connexion »

Requête de signature
 	
(Passant)
	Méthode : POST
	
	Requête :
	https://${BASE_URL}/send/apipsc/signsessiondata
	
	Json body :
	{
	"nationalId" : "${value_nationalId_PS1}",
	"clientID" : "${value_clientId_lPS1}",
	"proxy_session_id" : "${value_proxysessionid}",
	"session_state" : "${sessionstate}"
	}
	
	En vert = A valoriser
	En bleu = Nom du endpoint
	
	Exemple :
	https://10.3.8.165:8080/send/apipsc/signsessiondata
	
	Json body :
	{
	"nationalId" : "899700539500",
	"clientID" : " ans-odc-lps1-edc-bas",
	"proxy_session_id" : "dc3d6338-a841-4ccc-9a20-1774e09185bc",
	"session_state" : "b6e716e1-00b7-41ce-a62d-0f653cd57471"
	}
		- Code : 200
		
		- Le corps de la réponse (.json) contient :
		- key " **nationalId**" avec une valeur
		- key "**clientId**" avec une valeur
		- key "**proxy_session_id**" avec une valeur
		- key "**session_state**" avec une valeur
		- key "**signature**" avec une valeur
		
		- Les valeurs des variables (« nationalId », « clientId », « Proxy_session_id », « session_state », « signature ») sont identiques à celles envoyées dans la requête

Déconnexion PS1 LPS1	**Méthode :** DELETE **Requête** : https://${BASE_URL}/disconnect **Http header :** Cokkie : proxy_session_id=${proxy_session_id} En vert = A valoriser En bleu = Nom du endpoint **Exemple :** https://10.3.8.165:8080/disconnect	- Code : 200 - 'proxy_session_id' valeurs sont différents entre les 2 sessions - 'session_state' valeurs sont différent entre les 2 sessions
Requête de récupération des traces	**Méthode :** GET **Requête** : https://${BASE_URL}/traces?start=${value_start}&end=${value_end}	- Code : 200 - La réponse peut être au format texte ou pièce jointe, et doit

(Passant)	Format de la date : AAAA-MM-JJT00:00:00Z En vert = A valoriser En bleu = Nom du endpoint **Exemple :** https://10.3.8.165:8080/traces?start=2024-10-02T09:00:00Z&end=2024-10-02T11:32:00Z **Paramètres** : - start = début de la période de recherche choisie (Obligatoire) - end = fin de la période de recherche choisie (facultatif)	contenir les informations suivantes : • client_id • id_nat • session_state • timestamp • source_ip • source_port • proxy_session_id • certificats ○ cn ○ ou • error_code • request

10.6. Requêtes envoyées par le proxy e-santé sur l'API PSC de l'outil conformité

Dans les scénarii précédents, on décrit une requête POST envoyée sur le endpoint https://${BASE_URL}/send/apipsc/signsessiondata du proxy e-santé.

Le proxy e-santé a ensuite la charge de relayer cette requête de signature au mock service de l'API PSC de l'outil de conformité.

Le proxy effectue 2 requêtes sur l'API PSC :

1- **Une requête d'authentification sur le endpoint token-exchange du serveur d'autorisation.**

a- URL de la requête : https://auth.server.api.edc-psc.esante.gouv.fr/realms/signsessiondata/protocol/openid-connect/token

b- Paramètres à envoyer dans le body de la requête : ces paramètres sont conformes à la RFC 8693 et au volet transport du CI-SIS :
grant_type:urn:ietf:params:oauth:grant-type:token-exchange
subject_token:{{psc_token}}
subject_token_type:urn:ietf:params:oauth:token-type:access_token
client_id:ans-odc-lps1-edc-bas
subject_issuer:psc

c- Réponse de la requête :

```
{
        "access_token": {{access_token_api}},
        "expires_in": 14400,
        "refresh_expires_in": 14400,
       "refresh_token": {{refresh_token_api}},
        "token_type": "Bearer",
        "not-before-policy": 0,
        "session_state": "21d178d2-77fe-4a98-8734-d8d8df2115e0",
        "scope": "email profile"
    }
```

Remarque :

La variable {{psc_token}} qui valorise l'attribut **subject_token** contient **l'access token Pro Santé Connect** que le proxy e-Santé va échanger contre un access token API.

Le serveur d'autorisation étant implémenté dans une instance de Keycloak, la configuration de Keycloak exige que l'on indique l'émetteur du subject_token. Il s'agit de Pro Santé Connect : cf **subject_issuer:psc**

2- **Une requête métier de signature sur le mock service de l'API PSC de l'outil de conformité**

URL du mock service :
https://[identifiant_du_mock_service].mockservice.platines.esante.gouv.fr/mockservice/apipsc/signsessiondata

Le serveur d'autorisation exige de la part du proxy e-Santé une **authentification mTLS** avec le même certificat client qui a servi à réaliser l'authentification du professionnel de santé de test.

Rappel de l'exigence sur le DN du certificat : **CN = ans-odc-lps1-edc-bas, OU=<identifiant_structure_de_test>**

a- Exemple : https:// fdfe7fcd-9039-44a7-8c2d-1bea4dfd5ef3.mockservice.platines.esante.gouv.fr/mockservice/apipsc/signsessiondata

Nota Bene : la valeur de l'identifiant du_mock_service change pour chaque mock service créé sur la plateforme Platines.

b- Paramètres de la requête sur le mock service :
Header :

```
Authorization: Bearer {{api_token}}
Content-Type: application/json
User-Agent: PostmanRuntime/7.43.0
Accept: */*
Cache-Control: no-cache
Postman-Token: f983cafe-8b12-4c99-8815-143f8d92b37f
Host: fdfe7fcd-9039-44a7-8c2d-1bea4dfd5ef3.mockservice.platines.esante.gouv.fr
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
Content-Length: 178
```

Body :
{"nationalId":"899700506928","clientID":"ans-odc-lps1-edc-bas","proxy_session_id": "6eab7863-bc77-47c6-b6e9-5a30d5b9277a","session_state": "c9e32db5-cad8-455e-8f05-c9ab84bcc2f4"}

c- Réponse de la requête :

```
{
   "nationalId": "899700506928",
   "clientID": "ans-odc-lps1-edc-bas",
    "proxy_session_id": "6eab7863-bc77-47c6-b6e9-5a30d5b9277a",
   "session_state": "c9e32db5-cad8-455e-8f05-c9ab84bcc2f4",
   "signature": "/AabMWYqr4HPesjYzq25M2Zao0bVnIMQPM+PFccB7q4="
}
```

La réponse avec la valeur de la signature doit être transmise par le proxy e-Santé au LPS simulé dans Platines.

Remarque :

Le mock service exige de la part du proxy e-Santé une **authentification mTLS** avec le même certificat client qui a servi à réaliser l'authentification du professionnel de santé de test et l'authentification du proxy e-Santé sur le serveur d'autorisation.
Rappel du format du DN du certificat **CN = ans-odc-lps1-edc-bas, OU=<identifiant_structure_de_test>**

Conformément au volet transport du CI-SIS, le mock service contrôle cette exigence sur l'utilisation du même certificat comme suit :

1- **Le mock service extrait le hash du certificat utilisé pour réaliser la requête /token-exchange depuis le claim «** cnf » de l'access token API.

"cnf": {
"x5t#S256":
"MDExNjE5YmZmYzE0Y2Q1OGExMmZlMTkyNGI2N2U2ZTI3ZjkzNDQ1Njg3MjFiNzk4MGFmMDFkODkwZDA5ZTE0NQ==
"
},

2- Le mock calcule la valeur du hash du certificat utilisé pour s'authentifier sur le mock service et compare cette valeur à celle récupérée dans le claim « cnf » de l'access token API.

11. ANNEXES :

11.1. Exemple d'access token API décodé :


```
{
  "exp": 1737745735,
  "iat": 1737731335,
  "jti": "86bb1abd-3376-489d-af58-9368bf9500ad",
  "iss": "https://auth.server.api.edc-psc.esante.gouv.fr/realms/signsessiondata",
  "aud": "account",
  "sub": "f73a6cb8-e47d-4a91-bd1b-698baa4377c5",
  "typ": "Bearer",
  "azp": "ans-odc-lps1-edc-bas",
  "session_state": "21d178d2-77fe-4a98-8734-d8d8df2115e0",
  "acr": "1",
  "allowed-origins": [
    "/*"
  ],
  "realm_access": {
    "roles": [
      "offline_access",
      "uma_authorization",
      "default-roles-signsessiondata"
    ]
  },
  "resource_access": {
    "account": {
      "roles": [
        "manage-account",
        "manage-account-links",
        "view-profile"
      ]
    }
  },
  "scope": "email profile",
  "sid": "21d178d2-77fe-4a98-8734-d8d8df2115e0",
  "email_verified": false,
  "name": "KIT DOC0050692",
  "SubjectNameID": "899700506928",
  "cnf": {
    "x5t#S256": "MDExNjE5YmZmYzE0Y2Q1OGExMmZlMTkyNGI2N2U2ZTI3ZjkzNDQ1Njg3MjFiNzk4MGFmMDFkODkwZDA5ZTE0NQ=="
  },
  "preferred_username": "899700506928",
  "given_name": "KIT",
  "family_name": "DOC0050692"
}
```