Solution open-source pour le partage sécurisé
des données confidentielles dans le cloud.

Cible de Sécurité CSPN

Rédigée par SCILLE[1]

Contact : contact@scille.fr

Version de la cible : 15/04/2024

[1] https://parsec.cloud

1. Identification du produit évalué

1.1. Identification de la cible d'évaluation

Ce document décrit la cible de sécurité de PARSEC, solution open-source pour le partage sécurisé de données confidentielles dans le cloud. Cette cible de sécurité est élaborée en vue d'une évaluation Certification de Sécurité de Premier Niveau (CSPN) par l'Agence Nationale de Sécurité des Systèmes d'Information (ANSSI) ; elle préfigure également la cible de sécurité d'une certification Critères Communs en vue d'une future qualification standard (EAL3+) ou renforcée (EAL4+).

PARSEC est un ensemble de composants logiciels libres disponible sous forme de logiciel desktop pour la gestion collaborative de fichiers. Le partage sécurisé est effectué par point de montage ou via une interface utilisateur dédiée. Le produit sécurise les données sensibles avant qu'elles ne soient stockées sur les clouds publics. Le produit garantit la confidentialité, l'intégrité, l'historisation, le contrôle d'accès, la non-répudiation, l'authenticité et la gestion des opérations concurrentes.

1.2. Identification du produit

Catégorie	Identification
Organisation éditrice	SCILLE SAS
Lien vers l'organisation	https://parsec.cloud/
Nom commercial du produit	PARSEC_STANDARD_V1A
Lien vers le produit	https://parsec.cloud/
Numéro de la version évaluée	3.0.0
Catégorie de produit	Stockage sécurisé
Guide d'installation du serveur de métadonnées	Parsec-v3_0_0-Guide-Administration.pdf
Guide utilisateur	Parsec-v3_0_0-Guide-Utilisation.pdf

2. Argumentaire, description du produit évalué

2.1. Description générale du produit

2.1.1. Contexte

PARSEC est le fruit d'un travail mené en partenariat avec le Laboratoire Bordelais de Recherche en Informatique sur financement du Ministère des Armées via le dispositif de subvention RAPID dédié aux innovations duales.

A l'origine du projet de recherche PARSEC, il y a une vision stratégique de l'évolution du cloud et de l'internet : la sécurité et le partage sécurisé des données sont des enjeux majeurs des années à venir et la seule protection périmétrique du réseau informatique ne permet plus de répondre aux besoins de sécurité des données dont le champ est beaucoup plus vaste : mobilité des acteurs, sécurité dans le cloud, garanties d'intégrité, de confidentialité, de responsabilité des acteurs, etc. Le contrôle du trafic réseau ne garantit plus la confidentialité des données, encore moins l'intégrité et l'authenticité. La stratégie de sécurité doit devenir « *Zero-Trust* » c'est à dire à « *Confiance nulle* » :

1. Tout réseau est par défaut considéré comme hostile ;
2. Les menaces internes et externes sont présentes à tout moment sur le réseau ;
3. Être à l'intérieur d'un réseau interne n'est jamais un gage de confiance absolue ;
4. Chaque terminal, chaque utilisateur et chaque flux réseau doivent être authentifiés et autorisés ;
5. Les politiques de sécurité doivent être dynamiques et définies à partir d'autant de sources de données que possible.

Il y a également la conviction que la « transformation numérique» des organisations se décline en quatre axes stratégiques « *par conception* » :

1. **Sécurité dans le cloud.** Pour pouvoir faire face aux attaques de corruption ou de violation des données, la sécurité informatique dans le cloud doit commencer dès le stade de la conception du système d'information, en partant du principe que la vulnérabilité principale est le poste de travail de l'utilisateur; en d'autres termes que c'est le poste de travail (y compris sa dimension humaine) qui garantit le niveau de sécurité.
2. **Protection des données.** La protection des données résulte du règlement général européen sur la protection des données (RGPD) applicable à compter du 25 mai 2018. Les manquements sont très fortement sanctionnés. Le point fondamental, c'est que le responsable du traitement de la donnée est considéré comme l'acteur responsable, et en tant que tel il lui revient de prendre les mesures pour garantir la protection des données personnelles : c'est le principe *d'accountability***.**
3. **Mobilité.** Les postes de travail sont devenus massivement mobiles. La mobilité impacte l'organisation du travail : les organisations collaboratives à distance entraînent la disparition des frontières physiques de l'entreprise. La conséquence, c'est que les systèmes d'informations doivent être nativement « *responsive* », c'est à dire que leur ergonomie doit s'adapter naturellement au poste de travail, la règle de base de conception des pages étant « *l'expérience utilisateur* » ou UX, ce qui comprend également les normes d'accessibilité.
4. **Collaboration** : le partage de la connaissance crée l'intelligence collective. Tous les collaborateurs doivent avoir à chaque instant une vision cohérente de l'information.
5. **Ergonomie** : En matière de sécurité, le principal risque est le facteur humain. Le système doit être simple à utiliser, idéalement intuitif, faute de quoi il sera contourné.
6. **Agilité.** Une entreprise qui a construit son système d'information par briques applicatives au fil de ses besoins métiers doit gérer un patrimoine de plusieurs centaines voire milliers d'applications indépendantes traitant des données similaires. Face à ce constat, une stratégie gagnante est de migrer ses applications en méthodologie agile, son middleware et son infrastructure sur des solutions web sous licence logiciel libre, et de faire de ses informaticiens des contributeurs actifs.

2.1.2. Cas d'utilisation

PARSEC s'adresse aux organisations qui veulent construire des *enclaves de confiance* sur un nuage (cloud) en s'appuyant sur des infrastructures cloud de moindre niveau de sécurité et de faible coût. PARSEC est une brique supplémentaire qui permet de renforcer le niveau de sécurité d'un groupe de confiance qui respecte déjà les réglementations en vigueur, notamment celles relatives aux postes utilisateurs.

Puisque la compromission d'un seul poste de travail suffit à compromettre l'ensemble du groupe de confiance, la condition nécessaire au déploiement de PARSEC est de disposer d'un ensemble de postes de travail couvert par la réglementation applicable au niveau de protection recherché pour les données.

PARSEC permet le partage sur internet de données sensibles ou confidentielles sur un cloud privé ou public, y compris depuis un accès internet. PARSEC peut s'adresser par exemple aux PME/PMI technologiques, aux start-up, aux cabinets de conseil, aux entités de R&D ou aux professions indépendantes traitant de spécialités sensibles en apportant au profit des données partagées une garantie de confidentialité, d'intégrité, d'authenticité et d'historisation.

2.1.3. Résumé de la solution

PARSEC est composé de trois zones physiques décrites dans la Figure 1.

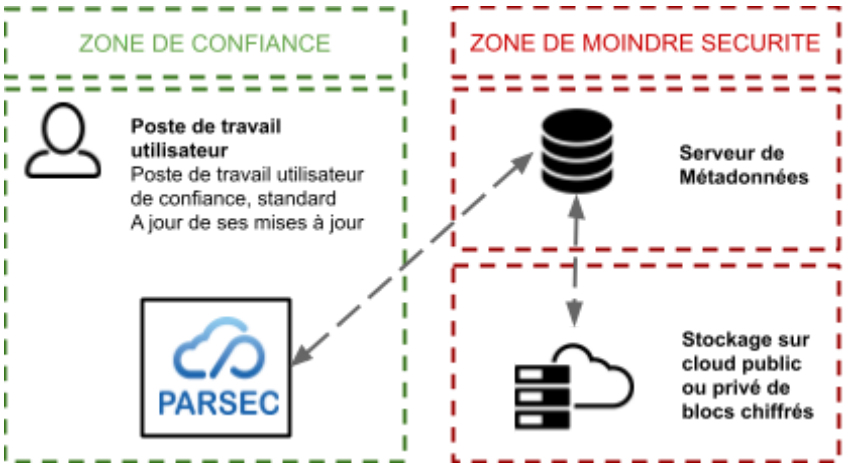

Figure 1 : Les zones physiques de la solution PARSEC.

Poste de travail utilisateur. La *zone de confiance* de PARSEC est limitée au poste de travail de l'utilisateur sur lequel est installé le client logiciel PARSEC. Le poste de travail doit être conforme à la réglementation souhaitée, notamment être à jour des correctifs de sécurité du système d'exploitation et être conforme à la politique de sécurité (PSSI) de l'entreprise, afin d'être considéré comme une zone de confiance. Le client PARSEC est une application lourde qui tourne sous Windows, MacOS et Linux. Seule la version Windows 11 fait l'objet d'une certification CSPN, objet de la présente cible de sécurité. Outre le serveur de métadonnées ci-après, ce sont ces deux logiciels lourds qui sont évalués dans le cadre de la certification. Le client assure les fonctions de mise sous enveloppe chiffrée et de signature des données à protéger et d'une façon générale toutes les fonctions de sécurité portant sur les données sensibles à protéger.

Serveur de métadonnées. Le serveur de métadonnées chiffrées est hébergé sur le cloud, donc dans une *zone de moindre sécurité*. En tant que tel il est interrogeable depuis n'importe quel poste de travail connecté à l'internet. Ce serveur ne sait rien des données échangées mais en revanche sait quel utilisateur les manipule. En outre, le serveur fournit une couche non cryptographique de contrôle d'accès afin de filtrer l'accès aux données chiffrées et de limiter l'accès en écriture aux utilisateurs avec les droits suffisants.

Stockage cloud. Les blocs chiffrés contenant les données à protéger sont stockés sur le cloud, donc également dans une *zone de moindre sécurité*. Leur accès par les clients se fait via le serveur de métadonnées qui y applique une couche de contrôle d'accès.

2.2. Choix techniques et architecturaux

2.2.1. Le modèle

PARSEC sécurise les données sensibles *avant* qu'elles ne soient stockées sur les clouds publics, en procédant en trois étapes :

- Découpage en blocs des fichiers avant chiffrement;
- Chiffrement de chaque bloc par la clé symétrique du workspace (WS_ENC_LAST_KEY);
- Chiffrement des métadonnées (arborescence, composition des fichiers, information de partage) par la clé symétrique du workspace (WS_ENC_LAST_KEY).

La séparation des acteurs :

- *Utilisateur* : représente une personne physique dans PARSEC. Un utilisateur dispose d'une clé asymétrique (USER_ENC_S_KEY / USER_ENC_P_KEY) lui permettant de chiffrer des données uniquement pour lui tel que son User Manifest (voir ci-dessous).
- *Poste de travail* : support physique (ordinateur bureautique ou portable).
- *Terminal* : c'est par l'intermédiaire d'un terminal (ou *device*) que l'utilisateur accède à PARSEC. Chaque utilisateur a potentiellement plusieurs terminaux (ex: un pour son ordinateur fixe et un autre sur son portable). Chaque terminal possède sa propre clé asymétrique de signature (DEVICE_SIG_S_KEY / DEVICE_SIG_P_KEY) permettant de signer les modifications qu'il a réalisées.

Le modèle de données :

- *File Manifest* : contient la liste des blocs (i.e. morceau de données chiffré) qui le composent ainsi que leur ordre pour reconstituer le fichier.
- *Folder Manifest :* index qui contient un ensemble d'entrées, chaque entrée étant un File Manifest ou un autre Folder Manifest.
- *Workspace Manifest :* index similaire au Folder Manifest, mais faisant office de racine pour l'arborescence du workspace.
- *User Manifest :* index racine propre à chaque utilisateur et contenant des éléments uniquement utilisés par celui-ci (ex: configuration synchronisée entre ses terminaux).

Le modèle de partage :

- *Workspace* : un groupe d'utilisateurs partageant un même espace de confiance. PARSEC effectue le partage de données sensibles via le chiffrement des clés du workspace (WS_ENC_KEYS) par la clé du destinataire du partage (USER_ENC_P_KEY) -- cette étape de chiffrement est répétée autant de fois qu'il y a de destinataires.
- *Organisation* : un ensemble des workspaces et un ensemble d'utilisateurs membres de l'organisation. L'accès à un workspace ne peut être accordé qu'aux membres de l'organisation. Deux organisations distinctes ne peuvent pas accéder au même workspace.

2.2.2. Les composants fonctionnels

Les composants fonctionnels sont décrits dans la Figure 2.

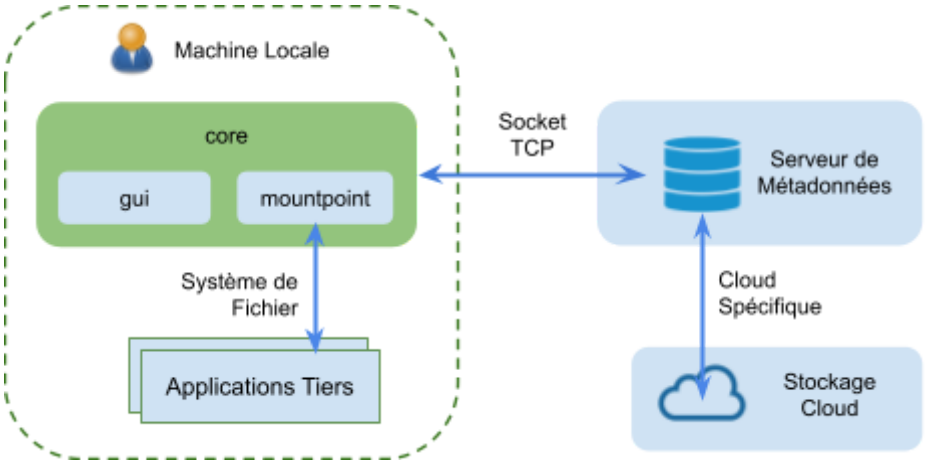

Figure 2. Les Composants Fonctionnels.

Mountpoint. Le composant *mountpoint* est responsable de l'interaction avec le système de fichiers pour communiquer avec les *applications tiers* selon une logique très simple consistant à transmettre toutes les requêtes natives au système de fichiers virtuel défini dans le composant *core*.

GUI. Le composant *gui* gère l'interface utilisateur interagit avec le composant *core* ou directement avec le système de fichiers natif, mais pas avec le composant *mountpoint*. Contrairement au composant *mountpoint*, le *gui* souscrit aux événements exposés par le serveur de métadonnées, tels que les notifications en cas d'actions concurrentes, de suppression d'un groupe etc. cela afin d'en informer l'utilisateur.

CORE. Outre les deux fonctions précédentes, le composant *core* intègre toute la logique du client et contient cinq sous-modules (Figure 3) dont les rôles sont les suivants :

- *Le Système de Fichiers Virtuels* (VFS) reçoit toutes les requêtes relatives au système de fichiers en provenance du composant *mountpoint* et, à cette fin, dispose d'une API qui simule une interface de système de fichiers. Pour optimiser ses performances, ce composant ne cherche pas à pousser les modifications jusqu'au serveur de métadonnées; il se contente de stocker de manière chiffrée les modifications sur le disque dur de la machine locale.

- *Le Synchroniseur* est le composant qui transfère périodiquement les données modifiées stockées sur la machine locale vers le serveur de métadonnées. Il s'occupe d'écouter les notifications du serveur de métadonnées en cas de modification des données par un autre terminal ainsi que de résoudre les conflits de version entre les données locales et celles du serveur de métadonnées. Pour ce faire, le synchroniseur stocke en mémoire les clés de déchiffrement des workspaces actuellement en cours d'utilisation (et ayant été récupérés auprès du serveur).
- *Le Gestionnaire d'Identités* stocke dans la mémoire locale l'identité de l'utilisateur connecté (sous la forme d'une session). La passphrase de l'utilisateur, qui n'est pas stockée, à laquelle on a rajouté un sel, chiffre la clé privée du terminal (DEVICE_SIG_S_KEY), ainsi que celle de l'utilisateur (USER_ENC_S_KEY) qui est partagée entre tous les terminaux de l'utilisateur. La DEVICE_SIG_S_KEY sert à signer une modification, et la USER_ENC_S_KEY sert à déchiffrer les métadonnées personnelles de l'utilisateur.
- *Le Partage* gère les opérations de partage. Il s'occupe de vérifier cryptographiquement les messages et données en provenance du serveur, ainsi que de détecter les changements dans les partages de workspace (afin de notifier l'utilisateur ou bien de lancer automatiquement une rotation de clé).

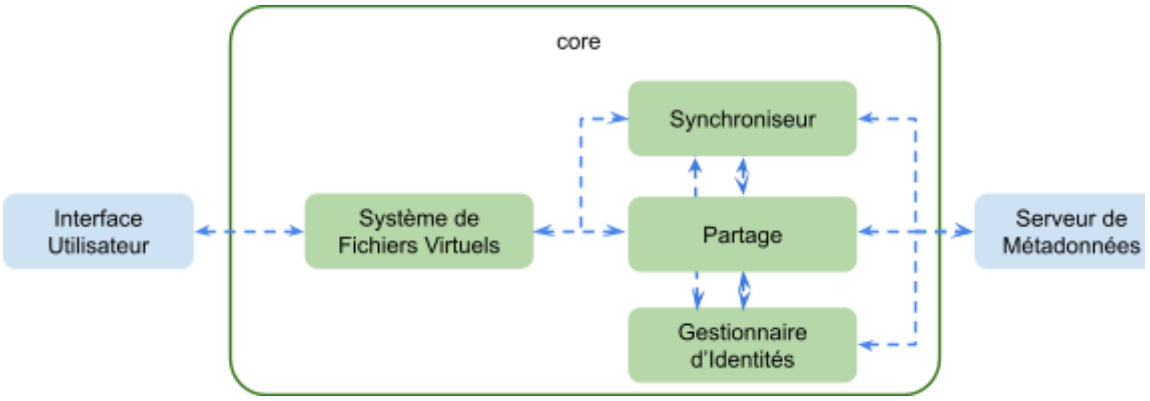

Figure 3. Core sous-modules.

Le serveur de métadonnées. Le serveur de métadonnées est dans un environnement distant et contient trois sous-modules :

- *Le Stockage des données / métadonnées*. Les données des fichiers (les *Blocks*) sont stockés sur un service de stockages objet (AWS S3 ou OpenStack Swift), les métadonnées (les *Vlobs*, pour *Versioned Blobs*) sont stockés dans une base PostgreSQL
- *Les Notifications des Terminaux* s'occupent d'envoyer des notifications aux terminaux connectés lors de modifications des données ou bien de la réception d'un nouveau message
- *Le Gestionnaire de Clés Publiques* contient une correspondance entre l'identité des utilisateurs/terminaux et leur clé publique (USER_ENC_P_KEY et DEVICE_SIG_P_KEY).

2.3. Profils et rôles des acteurs

2.3.1. Gestion des utilisateurs

Il existe trois profils pour les utilisateurs :

1. *Utilisateur Externe* permet:
    - la lecture et optionnellement l'écriture de workspaces qui lui ont été partagés
    - N'a pas accès au détails des utilisateurs présent dans l'organisation
    - la création de ses propres terminaux.
2. *Utilisateur Standard* permet :
    - tout ce qui est permis à un utilisateur *Externe*
    - la création d'un workspace;
    - la gestion de la documentation (création, modification, historique, informations sur l'intégrité);
    - le partage des données au sein d'un workspace;
3. *Administrateur* permet :
    - tout ce qui est permis à un utilisateur *Standard*;

- la création d'autres utilisateurs (*Administrateur*, *Externe* ou *Standard*);
- la suppression de n'importe quel *Utilisateur* quel que soit son profil.
- la modification d'un profil autre que lui-même

2.3.2. Gestion des terminaux

Seul l'utilisateur, quel que soit son profil, peut se créer un un nombre quelconque de terminaux (ou device). Le nombre de terminaux par utilisateur est généralement faible. La suppression d'un seul terminal est impossible : les terminaux partageant la clé privée de l'utilisateur USER_ENC_S_KEY, la compromission d'un de ses terminaux entraîne la compromission de l'ensemble de l'utilisateur qui doit donc être révoqué. Lorsqu'un utilisateur est révoqué, tous ses terminaux sont supprimés.

2.3.3. Gestion des workspaces et des documents

Il existe quatre rôles ayant des droits différents au sein d'un workspace :

1. *Lecteur* : il ne dispose que des accès qu'en lecture.
2. *Contributeur* : il dispose des accès en écriture et en lecture.
3. *Gérant* : il peut donner les droits sauf celui de propriétaire. Il a accès en lecture et en écriture.
4. *Propriétaire* : il peut donner tous les droits y compris celui de propriétaire. Il peut y avoir plusieurs propriétaires. Le créateur du workspace est propriétaire par défaut. Il a accès en lecture et en écriture. Seul un *Propriétaire* peut enclencher une rotation des clés.

2.4. Cinématique d'utilisateur

2.4.1. Création d'une organisation

La création d'une organisation a lieu en deux étapes:

1. Dans un premier temps un administrateur du serveur de métadonnées enregistre le nom de l'organisation et obtient un token d'initialisation de l'organisation qu'il transmet à la personne désignée pour être le premier administrateur de l'organisation.

2. Dans un second temps, l'application crée sur le poste de ce premier administrateur de l'organisation une clé d'organisation (ORG_ROOT_SIG_S_KEY, ORG_ROOT_SIG_P_KEY), une clé de compte d'utilisateur (USER_ENC_S_KEY, USER_ENC_P_KEY) et une clé de terminal (DEVICE_SIG_S_KEY, DEVICE_SIG_P_KEY). L'application certifie les clés publiques de l'utilisateur et de l'appareil avec la clé de signature de l'organisation et les télécharge sur le serveur de métadonnées. De plus, seule la partie publique de la clé racine de l'organisation (ORG_ROOT_SIG_P_KEY) est téléchargée dans le serveur de métadonnées, la partie secrète est intentionnellement oubliée, ce qui la rend irrécupérable.

2.4.2. Création d'un nouvel utilisateur

La création d'un nouvel utilisateur ne peut se faire que par un utilisateur existant, déjà enregistré dans l'organisation et ayant le profil Administrateur.

Considérons le cas où Alice est Administrateur et veut rajouter Bob :

1. Alice signale au serveur de métadonnées que Bob est invité au sein de l'organisation en transmettant son adresse e-mail.
2. Le serveur de métadonnées envoie un e-mail à Bob avec une URL d'invitation qui contient l'ID d'organisation et un identifiant unique du canal d'invitation.
3. Alice et Bob effectuent un échange de clé Diffie-Hellman (DH) [1] authentifié :
    a. Alice et Bob créent des clés asymétriques éphémères et échangent les parties publiques en utilisant le serveur de métadonnées comme canal de transmission pour déduire une clé secrète partagée dans le style de DH (ENROLLMENT_SHARED_KEY).
    b. Pour empêcher un serveur de métadonnées malveillant de modifier le canal DH (attaque *man-in-the-middle*), Alice et Bob authentifient leur clé secrète partagée ENROLLMENT_SHARED_KEY à l'aide du protocole *Short Authentication String* (SAS) [2]. Chaque partie communique verbalement ou via un canal physique de la main à la main un token SAS que son homologue doit valider parmi un ensemble de tokens (conformément aux recommandations de la littérature scientifique [3]).

4. Bob génère ses clés d'utilisateur (USER_ENC_P_KEY, USER_ENC_S_KEY) et de terminal (DEVICE_SIG_P_KEY, DEVICE_SIG_S_KEY) et utilise le canal authentifié pour communiquer leurs parties publiques à Alice.
5. Alice signe ces deux clés à l'aide de sa clé privée (DEVICE_SIG_S_KEY) et télécharge ces clés certifiées sur le serveur de métadonnées

Comme chaque clé d'utilisateur est signée par un terminal enregistré dans l'organisation et celle du premier utilisateur est signée par la clé racine (ORG_ROOT_SIG_S_KEY), en revalidant la chaîne de signatures, un client est en mesure de s'assurer qu'une clé a bien été ajoutée à PARSEC par un terminal légitime et peut donc être considérée comme valide.

Un utilisateur se voit attribuer une adresse email à sa création afin de signifier sa correspondance à une personne physique. Pour une adresse email donnée, il existe au plus un utilisateur non révoqué dans une organisation. De cette façon un utilisateur compromis peut être remplacé au sein de l'organisation (i.e. révocation de l'utilisateur existant puis création d'un nouvel utilisateur avec la même adresse email), tout en permettant aux autres utilisateurs de le retrouver via la même adresse email.

2.4.3. Création d'un nouveau terminal

La création d'un nouveau terminal fonctionne de manière similaire à celle d'un nouvel utilisateur à ceci près que le nouveau terminal n'a pas à créer de clé d'utilisateur (USER_ENC_P_KEY, USER_ENC_S_KEY) mais c'est au terminal existant de lui transmettre cette information de manière sécurisée. Le même mécanisme DH authentifié par SAS est utilisé comme décrit dans 2.4.2. La nouvelle clé de périphérique est certifiée de manière identique en utilisant la clé de signature de terminal existante (DEVICE_SIG_S_KEY) avant d'être mise à jour vers le serveur de métadonnées.

2.4.4. Gestion de la lecture d'un fichier

Le client PARSEC tente de privilégier l'accès local aux données lors de la lecture de fichier. Cela n'est pas toujours possible et la consultation du serveur de métadonnées peut s'avérer obligatoire. La lecture d'un fichier est illustré dans la Figure 4 :

1. Si le client PARSEC ne possède pas le File Manifest en local, il s'authentifie auprès du serveur de métadonnées pour le lui demander ;
2. Le serveur de métadonnées s'assure que le client a le droit d'y accéder et le lui envoie le cas échéant le file manifest chiffré par une des clés du workspace ;
3. Le client PARSEC récupère les clés de chiffrement du workspace (REALM_KEYS_BUNDLE) auquel appartient le fichier. Celles-ci étant fourni sous la forme d'un document (REALM_KEYS_BUNDLE) chiffré par une clé (REALM_KEYS_BUNDLE_KEY) elle-même chiffrée pour chaque utilisateur membre du workspace (REALM_KEYS_BUNDLE_ACCESS) (voir 2.4.5. Gestion des workspaces et du contrôle d'accès).
4. Le client PARSEC déchiffre le file manifest et en vérifie la signature (à noter que la phase de récupération de la clé publique du terminal ayant signé le manifest est analogue au mécanisme présenté dans le chapitre dédié à la gestion des utilisateurs/terminaux) ;
5. Le client PARSEC peut alors retrouver tous les blocs nécessaires à la lecture du fichier. Dans le cas des blocs non présents en local, le client PARSEC les demandes au serveur de métadonnées. Une fois récupérés, le client les déchiffre (en utilisant les clés de chiffrement du workspace) et vérifie leur hash ;
6. Finalement le client peut recombiner les blocs déchiffrés pour former le contenu du fichier demandé.

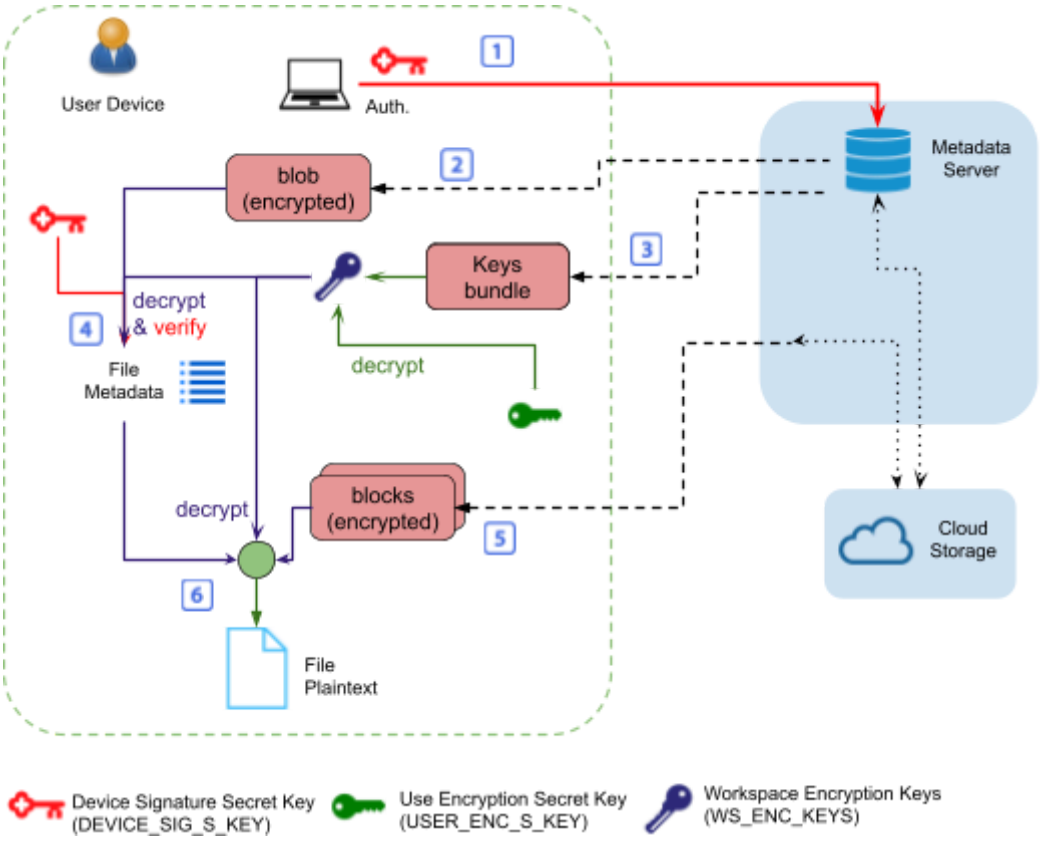

Figure 4 : Authentification et lecture d'un fichier à partir d'un terminal utilisateur

L'utilisateur interagit avec les fichiers en utilisant ses logiciels tiers classiques. Les données sont dans un premier temps stockées chiffrées sur le disque dur de la machine, cela pour des questions de performance et de résilience ainsi que pour permettre de fonctionner en mode hors ligne. Dans un second temps, le client PARSEC envoie les modifications, s'il y en a, au serveur de métadonnées.

L'historisation permet à l'utilisateur de lister toutes les versions de tel fichier particulier, et de restaurer le contenu à une version précédente.

2.4.5. Gestion des workspaces et du contrôle d'accès

Afin de pouvoir stocker des fichiers, l'utilisateur doit d'abord créer un workspace, le déclarer auprès du serveur et configurer sa clé de chiffrement initiale (WS_ENC_KEY).

À partir de là, le partage d'un workspace consiste en deux opérations :

1. Fournir l'information d'accès (WS_ID, ainsi que le rôle de l'utilisateur au sein du workspace). Cela est réalisé par l'upload dans le serveur d'un certificat de rôle (indiquant auteur, ID du workspace, ID de l'utilisateur, et rôle de ce dernier). Ce certificat est par la suite transmis à tous les utilisateurs de l'organisation.
2. Permettre au nouvel utilisateur de déchiffrer les données du workspace. Cela se fait au moyen d'un mécanisme de chiffrement de l'ensemble des clés de chiffrement du workspace sous une forme amalgamée (REALM_KEYS_BUNDLE) pour chaque utilisateur membre (via le REALM_KEYS_BUNDLE_ACCESS). De fait, au moment du partage, un REALM_KEYS_BUNDLE_ACCESS permettant de déchiffrer le dernier REALM_KEYS_BUNDLE est généré pour le nouvel utilisateur et uploadé dans le serveur.

2.5. Séquestre

Le séquestre est un service permettant de récupérer l'intégralité des données d'une organisation. Celui-ci est activable uniquement à la création de l'organisation (la clé d'autorité de séquestre SEQUESTER_AUTHORITY_P_KEY étant signée par ORG_ROOT_SIG_S_KEY).

Par la suite, SEQUESTER_AUTHORITY_S_KEY peut servir à signer des clés de service de séquestre (SEQUESTER_SERVICE_P_KEY) avec lesquels les utilisateurs sont forcés par le serveur à transmettre les clés de chiffrement de leurs workspaces.

Le cas d'usage typique de la fonctionnalité de séquestre est le suivant : un service d'inspection fait une enquête nécessitant d'accéder à toutes les données stockées sur les workspaces de la (ou des) personne(s) impliquée(s) dans l'enquête.

Il est à noter que les parties secrètes SEQUESTER_AUTHORITY_S_KEY ainsi que SEQUESTER_SERVICE_S_KEY sont utilisées uniquement dans un mode hors-ligne (i.e. pas besoin de communication directe avec le serveur de métadonnées) afin de permettre de les stocker dans un lieu sécurisé et de maîtriser au maximum leur environnement d'utilisation.

2.6. Hypothèses sur le fonctionnement et sur l'environnement de PARSEC

2.6.1. Hypothèses sur le poste de travail et le terminal

Les hypothèses concernant le poste de travail (qui est externe au produit) et la dégradation du niveau de sécurité si les hypothèses ne sont pas respectées sont les suivantes :

H_DESKTOP_INTEGRIY Intégrité du poste de travail	Le poste de travail est à jour de ses correctifs de sécurité et configuré conformément aux règles annexées à l'instruction interministérielle 901 : - EXP-POL-COR : définir et mettre en œuvre une politique de suivi et d'application des correctifs de sécurité. Le maintien du niveau de sécurité d'un système d'information impose une gestion organisée et adaptée des mises à jour de sécurité. Un processus de gestion des correctifs propre à chaque système ou applicatif doit être défini et adapté aux contraintes et au niveau d'exposition du système. - EXP-COR-SEC : déploiement des correctifs de sécurité. Les correctifs de sécurité des ressources informatiques locales doivent être déployés par l'équipe locale chargée des SI en s'appuyant sur les préconisations et les outils proposés par les services centraux. Le poste de travail est géré par un utilisateur ou un administrateur bienveillant, le code du client lourd de PARSEC n'a pas été altéré. Les failles "zéro-day" sont hors périmètre de l'évaluation. Le client lourd est mis à jour avec les dernières corrections de sécurité. Les clés de chiffrement et de sécurité sont générés par le système d'exploitation et en conséquence dépendent de la solidité de leur générateur d'aléa.
H_DESKTOP_CONFIDENTIALITY Contrôle du poste de travail en confidentialité	Le poste de travail ne quitte pas la surveillance de l'utilisateur lorsque l'utilisateur est authentifié sur PARSEC. Le poste de travail est obligatoirement mono-utilisateur.
H_DEVICE_AVAILABILITY_1 Disponibilité du terminal	Le terminal est considéré comme disponible. Si cette hypothèse n'est pas tenue, l'utilisateur pourra perdre l'accès à ses fichiers le temps de l'indisponibilité de son poste de travail, à moins qu'il n'ait configuré un autre terminal avec les mêmes droits d'accès ; en cas d'utilisation d'un poste de travail nomade, il peut donc être pertinent de configurer un autre poste avec ses droits. Afin de se prémunir de la perte de ses données en cas de perte d'un poste de travail, chaque utilisateur est supposé disposer d'au moins deux terminaux (*device* au sens de PARSEC).
H_DEVICE_AVAILABILITY_2 Détection d'une compromission	Le vol ou la compromission d'un poste de travail ou d'un terminal une fois détecté et la révocation de l'utilisateur compromis empêche immédiatement l'accès au

	service. Postérieurement, un mécanisme de rotation de clés automatique est déclenché pour chaque propriétaire de workspaces dont l'utilisateur révoqué était membre.
H_DEVICE_AUTHENTICATION Processus de suppression d'un terminal	L'administrateur et/ou l'utilisateur est correctement formé à l'utilisation de la cible d'évaluation (Target of Evaluation - ToE) conformément aux guides utilisateurs. Notamment : lorsqu'un terminal est compromis, la stratégie de mitigation consiste à recréer un nouvel utilisateur, à partager toutes les données avec ce nouvel utilisateur et à supprimer l'ancien utilisateur pour rendre ce dernier inopérant sur le système.
H_DELETED_USER_LOCAL_ACCESS Processus de suppression d'un utilisateur	Lors de la suppression d'un utilisateur, ce dernier, même s'il ne peut plus communiquer avec le serveur de métadonnées ni accéder aux nouvelles données ou modifications des données antérieures, peut continuer à lire les données qu'il a enregistrées (chiffrées ou pas) sur son poste de travail avant sa suppression. Cette hypothèse est identique à l'hypothèse sous-jacente de "*lazy revocation*" [4].

2.6.2. Hypothèses sur les utilisateurs

H_USER_CREATION_TOKEN Transmission physique du token	Lors de la création d'un nouvel utilisateur par un administrateur, il est supposé que les tokens utilisés par la méthode *Short Authenticated Strings* [2] (SAS) (voir 2.4.2.) sont envoyés verbalement ou par un **canal physique de la main à la main**.
H_PASSPHRASE Complexité de la passphrase	La passphrase permet de déchiffrer les clés privées de terminal et d'utilisateur (USER_ENC_S_KEY et DEVICE_SIG_S_KEY) sur le poste de celui-ci. Elle est demandée à l'utilisateur pour s'authentifier. La passphrase permet également de déchiffrer les clés symétriques (USER_MAN_KEY et LOCAL_ENC_KEY). Lors de la saisie de la passphrase par un nouvel utilisateur, il existe une alerte sur sa faiblesse éventuelle s'appuyant sur une bibliothèque de vérification de la complexité de la passphrase. Il sera déconseillé à un utilisateur de choisir une passphrase non conforme aux recommandations de l'ANSSI[2]. Nous faisons le choix de recommander une passphrase de 16 caractères dans un alphabet de 36 symboles (taille de clé équivalente 82) considérée par l'ANSSI de force moyenne. De manière alternative, les clés privées de terminal et d'utilisateur peuvent être protégées par le keyring du système d'exploitation, permettant ainsi de se passer de passphrase du point de vue de l'application. Dans cette configuration, la sécurisation des clés se base sur H_DESKTOP_INTEGRIY et H_DESKTOP_CONFIDENTIALITY.

2.6.3. Hypothèses sur le serveur de métadonnées

H_SERVER_INTEGRITY Intégrité du serveur de métadonnées	Le serveur de métadonnées est durci selon les recommandations de l'ANSSI (instruction interministérielle II901 par exemple).
H_SERVER_DDOS Protection contre le déni de service	L'environnement d'hébergement assure la disponibilité des métadonnées chiffrées dans le cas où l'interface extérieure du serveur de métadonnées ou de stockage dans le cloud est attaquée par déni de Service.

[2] https://www.ssi.gouv.fr/administration/precautions-elementaires/calculer-la-force-dun-mot-de-passe/

2.6.4. Hypothèses sur le stockage des blocs chiffrés

H_STOCKAGE_STANDARD Délégation du serveur de stockage	Les services cloud de stockage des blocs chiffrés sont commerciaux, de type Amazon Web Services S3. Les blocs chiffrés sont considérés comme accessibles en lecture.

2.6.5. Hypothèses sur la communication entre le serveur de métadonnées et le logiciel client lourd

H_TLS Communication chiffrée	La communication sécurisée entre le client lourd et le serveur de métadonnées est assurée par le protocole TLS.

2.6.6. Hypothèses sur la génération des clés de séquestre

H_GENERATION_SEQUESTER_KEY Génération des clés de séquestre	La génération des clés de séquestre (SEQUESTER_AUTHORITY_S_KEY et SEQUESTER_SERVICE_S_KEY) se fait via une commande tiers sans lien avec PARSEC (e.g. *openssl*). Il est attendu que cette génération se fasse en respectant les bonnes pratiques, en particulier en matière de génération d'entropie. Les clés générées doivent être des clés RSA d'une taille de 4096 bits.

2.6.7. Hypothèses sur le stockage des clés de séquestre

H_STOCKAGE_SEQUESTER_KEY Stockage des clés de séquestre	Le stockage des clés de séquestre (SEQUESTER_AUTHORITY_S_KEY et SEQUESTER_SERVICE_S_KEY) se fait dans un environnement sécurisé, hors-ligne et à froid. Plusieurs copies de ces clés sont stockées dans des endroits séparés.

3. Description de l'environnement technique

3.1. Système d'exploitation compatible pour le poste de travail

La certification est faite pour le client lourd PARSEC tournant sur Windows 11[3].

3.2. Matériel compatible ou dédié pour le serveur de métadonnées

Le serveur de métadonnées est évalué sur un serveur dédié tournant sur Ubuntu 22.04[4]. La configuration hardware est :

- Intel 64 bits bi-processeurs quadri-cœurs,
- 8 GB de RAM,
- 50 GB de disque dur,
- 2 Gigabit Ethernet.

3.3. Environnement cloud de stockage

Le système fonctionne sur tout environnement compatible Amazon S3 (stockage d'objets).

[3] Le client peut fonctionner sur des systèmes plus anciens mais ce fonctionnement dégradé n'est pas couvert par l'évaluation de sécurité.

[4] Le logiciel du serveur de métadonnées PARSEC est déployé sur un serveur dédié "on premise" conformément au Guide d'Administration du serveur de Métadonnées.

4. Description des biens sensibles à protéger par le produit évalué

Les biens sensibles à protéger par PARSEC sont :

Légende : C = Confidentialité, **I** = Intégrité, **A** = Authenticité et/ou Non-répudiation, **H** = Historisation.

Code	Bien sensible à protéger	C	I	A	H
B_FIC	Le contenu des **fichiers**.	X	X	X	X
B_MTD	Les **métadonnées** des fichiers (nom, arborescence, horodatage).	X	X	X	
B_UK	La **clé utilisateur** (USER_ENC_S_KEY) permettant de reconstruire en clair les données stockées sous forme chiffrée sur le cloud. Si la clé est modifiée cela n'a aucune conséquence pour le système.	X	X		
B_DK	La **clé du terminal** (DEVICE_SIG_S_KEY) permettant de signer les documents et les actions. Elle est à l'origine de la non-répudiation.	X	X		
B_TKN	Les **tokens** générés lors d'une invitation d'un utilisateur en utilisant la méthode SAS (*Short Authenticated Strings*) [2] lors de l'authentification du mécanisme d'échange de clés Diffie-Hellman (DH) [1].	X			
B_PASS	La **passphrase** utilisée pour l'authentification d'un utilisateur	X	X		
B_UA	Le compte **utilisateur** (user account), i.e., USER_ENC_P_KEY signée par un terminal tiers.		X		
B_DA	Le compte **device** ou terminal (device account), i.e. DEVICE_SIG_P_KEY.		X		
B_ACC	La liste des identifiants utilisateur (certificats) autorisés à **accéder** au un workspace.		X	X	X
B_UMK	La **clé du user manifest** USER_MAN_KEY pour chiffrer le "user manifest".	X	X		
B_WSK	Les **clés du workspace** WS_ENC_KEYS pour chiffrer ses données et métadonnées.	X	X		
B_RKBK	La **clé de chiffrement d'un keys bundle** REALM_KEYS_BUNDLE_KEY				
B_LK	La **clé locale** symétrique servant à chiffrer les blocs et les manifestes sur le poste client.	X	X		
B_ORK	La **clé publique racine de l'organisation** ORG_ROOT_SIG_P_KEY utilisée par l'administrateur.		X	X	
B_TKA	Le **token d'administration du serveur de métadonnées** PARSEC_ADMINISTRATION_TOKEN.	X		X	
B_ESK	La **clé d'enrôlement** ENROLLMENT_SHARED_KEY décrite dans "2.4.2. Création d'un nouvel utilisateur". Permet de créer un canal de communication sécurisé entre les deux parties de l'enrôlement	X	X		
B_SVK	La **clé de signature du séquestre (**SEQUESTER_AUTHORITY_S_KEY)	X	X		
B_SEK	La **clé de déchiffrement de séquestre (**SEQUESTER_SERVICE_S_KEY)	X	X		

Etat des biens à protéger :

Code	Bien sensible à protéger	Etat	Localisation
B_FIC	Les **fichiers et les données**	Repos	Poste client et service de stockage : découpés en blocs et chiffrés par la clé symétrique du workspace (WS_ENC_LAST_KEY) pour chaque bloc.
B_MTD	Les **métadonnées**	Transit Repos	Serveur de métadonnées et poste client, chiffrés par une clé symétrique (WS_ENC_KEY).
B_UK	La **clé utilisateur** (USER_ENC_S_KEY Curve25519)	Transit Repos	Au repos sur le poste client (Stockée sous forme chiffrée avec B_PASS). En transit lors de l'enrôlement d'un nouveau Device (chiffrée avec ENROLLMENT_SHARED_KEY).
B_DK	La **clé du terminal** (DEVICE_SIG_S_KEY Ed25519)	Repos	Poste client (Stockée sous forme chiffrée avec B_PASS).
B_TKN	Le **token d'enrôlement** (Short Authenticated Strings)	Transit	Poste client.
B_PASS	La **passphrase**	Repos	Poste client.
B_UA	Le compte **utilisateur** (user Account)	Repos Transit	Serveur de métadonnées Poste client (stocké sous forme chiffré avec une clé symétrique)
B_DA	Le compte **device** ou terminal	Repos Transit	Serveur de métadonnées Poste client (stocké sous forme chiffré avec une clé symétrique)
B_ACC	La liste des utilisateurs autorisés à **accéder** au workspace.	Repos Transit	Serveur de métadonnées et poste client.
B_UMK	La **clé de chiffrement du manifest utilisateur** (USER_MAN_KEY XSalsa20/Poly1305 MAC)	Repos Transit	Au repos sur le poste client (Stockée sous forme chiffrée avec B_PASS) En transit lors de l'enrôlement d'un nouveau Device (chiffrée avec ENROLLMENT_SHARED_KEY)
B_WSK	Les **clés du workspace** (WS_ENC_KEYS XSalsa20/Poly1305 MAC) pour chiffrer ses données et métadonnées.	Repos Transit	Au repos sur le serveur de métadonnées (stockées sous la forme de keys bundle chiffré avec B_RKBK) En transit sur le poste client lorsque celui-ci a besoin de lire/modifier les données du workspace.
B_RKBK	La **clé de chiffrement d'un keys bundle** (WS_ENC_KEYS XSalsa20/Poly1305 MAC)	Repos Transit	Au repos sur le serveur de métadonnées (stockée chiffrée avec B_UK. En transit sur le poste client lorsque celui-ci a besoin d'obtenir B_WSK.
B_LK	La **clé de chiffrement local** (LOCAL_ENC_KEY XSalsa20/Poly1305 MAC)	Repos	Poste client. Stockée sous forme chiffrée avec B_PASS.
B_ORK	La **partie publique de la clé racine de l'organisation** (ORG_ROOT_SIG_P_KEY Ed25519)	Repos Transit	Serveur de métadonnées Poste client stockée sous forme chiffrée avec B_PASS.

			En transit lors de l'enrôlement d'un nouveau User/Device (chiffrée avec ENROLLMENT_SHARED_KEY).
B_TKA	Le **token d'administration** du serveur de métadonnées	Repos	Serveur de métadonnées
B_ESK	La **clé d'enrôlement**	Repos	Poste client
B_SVK	La **clé de signature du séquestre** (SEQUESTER_AUTHORITY_S_KEY)	Repos	Stockage à froid dans un environnement sécurisé (e.g. clé usb dans un coffre fort)
B_SEK	La **clé de déchiffrement de séquestre** (SEQUESTER_SERVICE_S_KEY)	Repos	Stockage à froid dans un environnement sécurisé (e.g. clé usb dans un coffre fort)

5. Profil des agents menaçants et menaces

5.1. Agents menaçants (qui fait l'attaque)

S'agissant de logiciel open source, nous considérons comme attaquant tout agent, humain ou logiciel, qui a pris connaissance des fonctionnalités et du code source du produit[5].

L'attaquant va chercher à récupérer des informations confidentielles ou mettre en défaut le fonctionnement du groupe de confiance.

- Attaquant sur le cloud ayant des droits sur le stockage dans le cloud des blocs chiffrés
- Attaquant de type "Man in the Middle" entre le client et le serveur de métadonnées.
- Attaquant sur le cloud ayant des droits sur le serveur de métadonnées et/ou sur la base de métadonnées.
- Attaquant disposant du contrôle d'un poste de travail d'un utilisateur ayant des droits sur un workspace.
- Attaquant disposant du contrôle d'un poste de travail d'un utilisateur supprimé.
- Attaquant disposant des clés de séquestre.

5.2. Chemins d'attaque

Les chemins d'attaque pourraient être les suivants :

- Confidentialité des données : compromission des WS_ENC_LAST_KEY ou conduite d'une attaque par force brute ou cryptanalyse (attaque de texte clair connue)
- Intégrité des données : collision des fonctions de hachage cryptographique
- Non répudiation et authenticité : falsification de signature numérique
- Audit et historisation : envoi d'informations périmées (c'est à dire diffusion aux utilisateurs de contenu obsolète)
- Compromission du terminal : conduite d'une attaque par force brute pour trouver la clé de l'utilisateur, protégé par une passphrase durcie.
- Compromission du terminal de séquestre : par vol de matériel client

[5] S'agissant d'un logiciel libre, tout expert en sécurité informatique est un attaquant potentiel.

5.3. MENACES SUR LES BIENS SENSIBLES

Code	Menace	Bien sensible à protéger																
		B_FIC	B_MTD	B_UK	B_DK	B_TKN	B_PASS	B_UA	B_DA	B_ACC	B_UMK	B_WSK	B_RKBK	B_LK	B_ORK	B_TKA	B_ESK	B_SVK	B_SEK
M1	Altération (corruption) des métadonnées	X	X					X	X										
M2	Altération (corruption) des blocs chiffrés stockés	X																	
M3	Altération (corruption) du client logiciel PARSEC sur le poste de travail de l'utilisateur / Compromission d'un poste de travail (ransomware, virus)	X	X	X	X		X			X	X	X	X	X					
M4	Altération de l'historique des fichiers		X																
M5	Compromission du secret "utilisateur"	X	X	X							X	X	X						
M6	Compromission du secret "terminal"				X			X	X	X									
M7	Violation de données utilisateur	X	X																
M8	Bypass de la connexion (Man-In-the-Middle)	X	X	X	X			X	X			X	X		X		X		
M9	Bypass de la vérification de signature d'un document	X	X					X	X										
M10	Déni de service sur le serveur de métadonnées	X	X									X	X						
M11	Violation de données par contournement du serveur de métadonnées et lecture directe des blocs chiffrés	X																	
M12	Perte d'un poste de travail utilisateur			X	X						X			X					
M13	Violation de données par écoute des flux	X	X									X	X						
M14	Destruction de la base de données du serveur de métadonnées	X	X					X	X	X		X	X						
M15	Ingénierie sociale (vol d'information de connexion)					X	X									X	X		
M16	Compromission du secret de séquestre																	X	X

6. Description des fonctions de sécurité du produit

Les fonctions de sécurité cryptographiques s'appuient intégralement sur la bibliothèque *libsodium*[6].

6.1. F1_CONFIDENTIALITY : Confidentialité des données

La confidentialité des données est assurée exclusivement par le poste client. Les postes des clients actifs sont la seule entité de confiance.

Sans l'activation du séquestre à la création de l'organisation, seul le client connaît les clés de chiffrement. Ni l'administrateur, ni une quelconque autre autorité administrative ne peut accéder aux clés de chiffrement.

Dans le cas d'une organisation avec le mode séquestre, les possesseur des clés de séquestre sont en mesure d'accéder aux clés de chiffrement des données et métadonnées. Toutefois, même dans ce mode, les clés personnelles (USER_ENC_S_KEY et DEVICE_SIG_S_KEY) restent uniquement connues du poste client (i.e. les possesseurs des clés de séquestre ne sont jamais en mesure de falsifier des signatures ou d'usurper l'identité du client).

La fonction de sécurité assure :

- la confidentialité des métadonnées sur le serveur de métadonnées (XSalsa20)
- la confidentialité des blocs de données tels que stockés sur le cloud (XSalsa20)

La bibliothèque cryptographique utilisée est *libsodium*. Les mécanismes détaillés sont précisés dans la spécification cryptographique PARSEC.

Les biens à protéger sont : **B_FIC, B_MTD, B_UK, B_DK, B_TKN, B_PASS, B_UMK, B_LK.**

6.2. F2_INTEGRITY : Intégrité des données

PARSEC garantit l'intégrité des fichiers stockés dans un workspace. Chaque métadonnée, signée par la DEVICE_SIG_S_KEY (Ed25519), a une empreinte unique, qui permet de détecter toute modification. Les empreintes des blocs constitutifs d'un fichier, générés par un mécanisme de hachage sont stockées dans les métadonnées afin d'en garantir l'intégrité.

Les biens à protéger sont : **B_FIC, B_MTD, B_UA, B_DA, B_ACC.**

6.3. F3_AUTHENTICITY : Non répudiation et Authenticité

Les fichiers sont signés et authentifiés par la DEVICE_SIG_S_KEY (Ed25519) qui est une clé dédiée à la signature. Il est possible de connaître l'identité de l'utilisateur qui a modifié les données. Et un utilisateur qui a modifié les données ne peut pas nier que c'était lui.

PARSEC procède à plusieurs vérifications :

- au niveau du client PARSEC : comparaison que la date enregistrée au niveau du serveur de métadonnées est la même que celle accessible via le manifest (qui n'est accessible que par sa USER_ENC_S_KEY)
- au niveau du serveur de métadonnées PARSEC : vérification que le terminal qui a signé la donnée est autorisé à accéder au workspace.

Si l'une des deux vérifications susmentionnées est négative, les actions suivantes sont prises :

- filtrage des documents compromis : le client PARSEC n'affichera que les documents contrôlés "Valide" (PARSEC ne procède à aucune suppression).
- émission d'une notification d'alerte avec possibilité d'auditer le fichier suspect.

Les biens à protéger sont : **B_FIC, B_MTD, B_ACC.**

[6] https://doc.libsodium.org/ libsodium, contrairement à OpenSSL, masque l'utilisation des composants cryptographiques de bas niveau ce qui permet une implémentation sans risque.

6.4. F4_USER_AUTHENTICATION : Authentification des Utilisateurs

Sur le poste de travail de l'utilisateur, le déchiffrement des USER_ENC_S_KEY et DEVICE_SIG_S_KEY passe par défaut par l'entrée d'une passphrase (Argon2id + XSalsa20). Le chiffrement intègre une somme de contrôle (hachage Poly1305 MAC) permettant de détecter la modification des clés chiffrées.

La DEVICE_SIG_S_KEY à s'authentifier auprès du serveur de métadonnées. La USER_ENC_S_KEY sert à déchiffrer les données propres à cet utilisateur. Si l'authentification auprès du serveur de métadonnées n'est pas possible, l'accès aux documents qui ne sont pas stockés dans le cache local de la machine sera impossible.

Les biens à protéger sont : **B_UK, B_DK, B_TKN, B_PASS, B_UA, B_ACC.**

6.5. F5_USER_CHAIN : chaîne de confiance

Tous les utilisateurs et leurs appareils faisant partie d'une organisation, sont liés par une chaîne de signature à l'administrateur qui les a créés selon la cinématique utilisateur décrite dans la section 2.4.

PARSEC procède à la vérification suivante en amont de la production de la donnée : si un attaquant essaie d'uploader vers le serveur de métadonnées des certificats de utilisateur/terminal signés avec une clé invalide ou un certificat antidaté (vu que le certificat n'est pas chiffré, il est vérifiable par le serveur), le serveur les refuse pour cause de tentative d'un attaquant d'envoyer des données corrompues au serveur.

Les biens à protéger sont : **B_UA, B_DK, B_ORK.**

6.6. F6_DEVICE_NON_REPUDIATION : Contrôle du terminal en responsabilité

La solution PARSEC garantit la non répudiation pour des postes de travail partagés entre plusieurs utilisateurs, sous réserve que la session PARSEC de chaque utilisateur soit close après chaque utilisation. Chaque modification au sein d'un workspace est signée par la DEVICE_SIG_S_KEY (Ed25519) de l'utilisateur.
Les mécanismes détaillés sont précisés dans la spécification cryptographique PARSEC.

Les biens à protéger sont : **B_MTD**.

6.7. F7_DEVICE_AUTHENTICATION : Authentification des terminaux

L'utilisateur peut créer de nouveaux terminaux. Le nouveau terminal créé dispose d'une clé (DEVICE_SIG_S_KEY - Ed25519) qui lui est propre et lui permet de s'authentifier auprès du serveur de métadonnées ainsi que de signer les données qu'il produit. Tous les terminaux d'un utilisateur partagent la clé USER_ENC_S_KEY. Un terminal créé ne peut pas être supprimé. Un terminal compromis nécessite la suppression de l'utilisateur concerné par l'administrateur.

Les biens à protéger sont : **B_MTD, B_FIC, B_ACC.**

6.8. F8_ENROLLMENT_TRANSMISSION : Transmission des informations privée lors de la création d'un nouveau terminal ou d'un nouvel utilisateur

Lors de la création d'un nouveau terminal ou utilisateur, ce dernier doit recevoir la clé publique de l'organisation ORG_ROOT_SIG_P_KEY. En outre, lors de la création d'un nouveau terminal, le terminal existant doit réussir à transmettre à ce dernier la USER_ENC_S_KEY de manière sécurisée. La transmission de ces informations est effectuée par le canal sécurisé authentifié obtenu en utilisant l'échange de clés DH avec une méthode SAS (Short Authentication String) décrite aux paragraphes 2.4.2 & 2.4.3.

Les biens à protéger sont : **B_UK, B_TKN, B_ESK.**

6.9. F9_ACCESS_CONTROL : Sécurité de contrôle d'accès

Lors du partage d'un workspace avec un utilisateur, le serveur de métadonnées enregistre le droit d'accès à ce workspace par l'utilisateur, signé par la clé de terminal (DEVICE_SIG_S_KEY) de l'utilisateur à l'origine du partage. Il connaît ainsi tous les droits des utilisateurs.

L'utilisateur, lui, reçoit un REALM_KEYS_BUNDLE_ACCESS lui permettant de retrouver les clés symétriques de workspace (WS_ENC_KEY) qui lui permettra de déchiffrer les métadonnées. Le serveur de métadonnées ne connaît pas les clés de workspace.

Cette fonction empêche donc un attaquant d'accéder aux biens sensibles s'il n'est pas connu par le serveur de métadonnées, et s'il arrive à prendre le contrôle du serveur de métadonnées, il ne pourra pas les déchiffrer s'il n'est pas en possession des clés symétriques de workspace (WS_ENC_KEY).

La suppression d'un utilisateur entraîne l'impossibilité pour les terminaux de ce dernier de se connecter au serveur de métadonnées et donc de créer de nouvelles données ou de créer de nouveaux utilisateurs. Tout ce qu'il a fait avant la date de suppression reste valide.

En outre, REALM_KEYS_BUNDLE_ACCESS et REALM_KEYS_BUNDLE ne sont jamais stockés de manière persistante sur le poste client. Ces éléments sont récupérés auprès du serveur de manière paresseuse et gardés dans la mémoire vive de la machine. De fait, la révocation d'un utilisateur permet de lui couper l'accès aux clés symétriques de workspace (WS_ENC_KEY) auquel il avait précédemment accès.

Les biens à protéger sont : **B_FIC, B_MTD, B_ACC.**

6.10. F10_WS_KEY_ROTATE : Rotation de la clé de chiffrement d'un workspace

Le propriétaire d'un workspace dispose de la connaissance des utilisateurs illégitimes, c'est-à-dire les utilisateurs précédemment révoqués par l'administrateur ou les utilisateurs préalablement retirés du workspace.

Cette fonction permet au terminal Parsec de déclencher automatiquement la rotation de la clé de chiffrement du workspace : une nouvelle clé est générée est sera désormais utilisée pour toutes les nouvelles données et métadonnées du workspace ce qui garantit que les utilisateurs devenus illégitimes ne sont plus en capacité de déchiffrer les documents du workspace postérieures à sa révocation.

Les documents antérieurs à sa révocation ne sont pas rechiffrés car on considère que l'utilisateur y a déjà eu accès, ils restent néanmoins protégés, non cryptographiquement, par F9_ACCESS_CONTROL et F11_WS_KEY_TRANSMISSION qui empêchent l'utilisateur révoqué d'accéder à la fois aux métadonnées et données chiffrées, ainsi qu'aux clés de déchiffrement du workspace.

Les biens à protéger sont : **B_FIC**, **B_MTD.**

6.11. F11_WS_KEY_TRANSMISSION : Transmission des clés symétriques de workspace (WS_ENC_KEYS) lors de du partage d'un workspace avec un utilisateur

Une nouvelle clé symétrique de workspace est générée à chaque rotation de clé (c.f. F10_WS_KEY_ROTATE). L'ensemble des clés d'un workspace est stocké dans REALM_KEYS_BUNDLE, lui-même chiffré par une REALM_KEYS_BUNDLE_ACCESS et stocké côté serveur.

La REALM_KEYS_BUNDLE_KEY est quant à elle chiffrée pour chaque utilisateur ayant accès au workspace avec sa clé de chiffrement asymétrique (USER_ENC_P_KEY) pour former un REALM_KEYS_BUNDLE_ACCESS et stockée côté serveur.

En cas de rotation des clés du workspace, un nouveau REALM_KEYS_BUNDLE est généré (contenant une clé de plus que le REALM_KEYS_BUNDLE précédant) et les REALM_KEYS_BUNDLE_ACCESS de chaque membre du workspace sont régénérés.

Il est à noter que le terminal d'un utilisateur ne stocke jamais sur disque les REALM_KEYS_BUNDLE / REALM_KEYS_BUNDLE_ACCESS. Ces éléments sont demandés au serveur de manière paresseuse (i.e. quand le terminal a besoin d'envoyer ou de recevoir des données/métadonnées au serveur) puis gardés en mémoire vive.

Les biens à protéger sont : **B_FIC**, **B_MTD**.

6.12. F12_USER_ROLE_MANAGEMENT : Gestion des utilisateurs et de leurs droits

Les administrateurs se différencient des utilisateurs normaux par le fait que leur signature est certifiée. Les signatures d'administrateur (DEVICE_SIG_P / S_KEY) sont attestées par une vérification récursive des signatures-mères jusqu'à la racine de l'organisation. Chaque fois qu'un administrateur effectue des opérations spécifiques, il télécharge un certificat signé sur le serveur de métadonnées. Le serveur de métadonnées vérifie la signature du certificat (en remontant la chaîne) et autorise l'opération si la signature est attestée. Chaque fois que des utilisateurs normaux récupèrent le contenu signé par les administrateurs à partir du serveur de métadonnées, ils appliquent le même mécanisme de vérification de la chaîne que la signature.

Les rôles lecteur, auteur, contributeur et propriétaire sont appliqués de la même manière que le mécanisme de validation de la signature de l'administrateur.

Les biens à protéger sont : **B_FIC**, **B_MTD, B_ACC**.

6.13. F13_SERVER_ADMINISTRATION : Administration du serveur de métadonnées

L'administration du serveur de métadonnées, conforme aux règles de l'art de l'administration d'un serveur, permet de procéder à la création initiale de l'organisation. Cette fonctionnalité est protégée par un token d'accès et n'implique pas de mécanismes cryptographiques.

Les biens à protéger sont : **B_TKA**.

6.14. F14_SEQUESTER_ADMINISTRATION : Administration du système de séquestre

L'administration du séquestre permet de mettre en place des clés de séquestre afin de récupérer les clés de chiffrement des workspaces (WS_ENC_KEYS) pour déchiffrer l'ensemble des données et métadonnées.

Les parties privées des clés de séquestre sont protégées par un stockage adapté décrit dans l'hypothèse H_STOCKAGE_SEQUESTER_KEY.

Les biens à protéger sont : **B_SVK**, **B_SEK**.

7. Couverture des besoins de sécurité

Le tableau ci-dessous indique comment les menaces sont couvertes, soit par les fonctions de sécurité, soit par les hypothèses.

Fonction de Sécurité | Menaces	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12	M13	M14	M15	M16
F1_CONFIDENTIALITY : Confidentialité des données							M7	M8			M11		M13			
F2_INTEGRITY : Intégrité des données	M1	M2	M3						M9							
F3_AUTHENTICITY : Non répudiation et Authenticité	M1	M2														
F4_USER_AUTHENTICATION : Authentification des Utilisateurs	M1	M2			M5	M6	M7	M8				M12			M15	
F5_USER_CHAIN : chaîne de confiance	M1		M3					M8				M12			M15	
F6_DEVICE_NON_REPUDIATION Contrôle du terminal en responsabilité	M1				M5	M6	M7	M8				M12			M15	
F7_DEVICE_AUTHENTICATION : Authentification des terminaux					M5	M6									M15	
F8_ENROLLMENT_TRANSMISSION : Transmission des informations privée lors de la création d'un nouveau terminal ou d'un nouvel utilisateur					M5								M13			
F9_ACCESS_CONTROL : Sécurité de contrôle d'accès					M5	M6		M8								
F10_WS_KEY_ROTATE : Rotation de la clé de chiffrement d'un workspace			M3		M5	M6						M12				
F11_WS_KEY_TRANSMISSION : Transmission de la clé symétrique de workspace (WS_ENC_KEYS) lors de du partage d'un workspace avec un utilisateur											M11		M13			
F12_USER_ROLE_MANAGEMENT : Gestion des utilisateurs et de leurs droits							M7		M9							
F13_SERVER_ADMINISTRATION : Administration du serveur de métadonnées															M15	
F14_SEQUESTER_ADMINISTRATION : Administration du système de séquestre																M16
Hypothèse | Menaces	**M1**	**M2**	**M3**	**M4**	**M5**	**M6**	**M7**	**M8**	**M9**	**M10**	**M11**	**M12**	**M13**	**M14**	**M15**	**M16**
H_DESKTOP_INTEGRIY Intégrité du poste de travail	M1	M2	M3		M5	M6	M7	M8								
H_DESKTOP_CONFIDENTIALITY Contrôle du poste de travail en confidentialité	M1	M2			M5	M6	M7	M8							M15	
H_DEVICE_AVAILABILTY_1 Disponibilité du terminal												M12				
H_DEVICE_AVAILABILY_2 Détection immédiate d'une compromission	M1	M2					M7	M8				M12				
H_DEVICE_AUTHENTICATION Processus de suppression d'un terminal	M1					M6	M7	M8				M12				
H_DELETED_USER_LOCAL_ACCESS Processus de suppression d'un utilisateur	M1				M5		M7	M8								
H_USER_CREATION_TOKEN Transmission physique du token					M5		M7	M8							M15	

H_PASSPHRASE : Complexité de la passphrase imposée								M8							M15	
H_SERVER_INTEGRITY Intégrité du serveur de métadonnées	M1			M4										M14		
H_SERVER_DDOS Protection contre le déni de service										M10						
H_STOCKAGE_STANDARD Délégation du serveur de stockage		M2														
H_SSL Communication chiffrée								M8					M13			
H_GENERATION_SEQUESTER_KEY Génération des clés de séquestre																M16
H_STOCKAGE_SEQUESTER_KEY Stockage des clés de séquestre																M16
Hypothèse | Menaces	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12	M13	M14	M15	M16

8. Glossaire

Sigle ou acronyme	Définition
Base de métadonnées	Base de données relationnelle utilisée pour stocker les métadonnées chiffrées (il s'agit d'une base PostgreSQL).
Bootstrap ou amorçage	Initialisation du compte utilisateur par la création ou récupération de l'index.
DEVICE_SIG_P_KEY	Clé publique de signature du terminal (device).
DEVICE_SIG_S_KEY	Clé privée (secret key) de signature du terminal (device). Correspond à B_DK.
ENROLLMENT_SHARED_KEY	Clé symétrique permettant de créer un canal de communication sécurisé entre les deux parties de l'enrôlement. Correspond à B_ESK.
File Manifest	Index qui contient la liste des blocs, l'index de clé de workspace utilisé pour chiffrer chaque bloc, ainsi que l'ordre permettant le recomposer le fichier.
Folder Manifest	Index qui contient un ensemble d'entrées, chaque entrée étant un File Manifest ou un autre Folder Manifest.
LOCAL_ENC_KEY	Clé Locale symétrique servant à chiffrer les blocs et les manifestes sur le poste client. Correspond à B_LK.
Organisation	Un ensemble d'utilisateurs et d'espaces de travail partagés (workspaces), équivalent à une institution, une entreprise ou une association.
ORG_ID	Le nom unique de l'organisation.
ORG_ROOT_SIG_P_KEY	Clé publique de la clé de signature racine de l'organisation.
ORG_ROOT_SIG_S_KEY	Clé secrète de la clé de signature racine de l'organisation.
Poste de travail	Support physique (ordinateur, fixe ou portable) sur lequel PARSEC est installé. "Desktop" en anglais.
Serveur de métadonnées	Interface logicielle qui sert les données chiffrées stockées dans la base de métadonnées.
Stockage objet	Service de stockage compatible du standard de fait S3 (Simple Storage Service) : un site d'hébergement de fichiers proposé par Amazon Web Services.

Terminal	Support logique PARSEC installé sur un poste de travail permettant à l'utilisateur de s'authentifier auprès du serveur de métadonnées. Plusieurs terminaux peuvent être installés sur un même poste de travail. "Device" en anglais.
USER_ENC_P_KEY	Clé publique de chiffrement de l'utilisateur.
USER_ENC_S_KEY	Clé privée (secret key) de chiffrement de l'utilisateur. Correspond à B_UK.
USER_MAN_KEY	Clé de chiffrement du manifest utilisateur
VLOB	Versioned (Binary) Large OBject.
Workspace	Espace de travail (groupe de fichiers) partagé entre plusieurs utilisateurs partageant le même niveau de confiance. Appelée aussi "enclave de confiance", "bulle de confiance", "bulle de sécurité", "espace de travail", "enclave sécurisée" ou "workspace" (en anglais).
WS_ID	Identifiant du workspace.
WS_ENC_KEYS	Clés de chiffrement symétrique du workspace.
WS_ENC_LAST_KEY	Dernière clé de chiffrement symétrique du workspace créée. C'est cette clé qui est utilisée pour chiffrer de nouvelle données et métadonnées.
REALM_KEYS_BUNDLE	Document contenant une ou plusieurs clé WS_ENC_KEYS associé à un index chiffre avec REALM_KEYS_BUNDLE_KEY
REALM_KEYS_BUNDLE_KEY	Clé symétrique utilisée pour chiffrer REALM_KEYS_BUNDLE
REALM_KEYS_BUNDLE_ACCESS	Document à destination de chaque membre d'un workspace chiffre avec leur clé USER_ENC_P_KEY contenant la clé REALM_KEYS_BUNDLE_KEY
SEQUESTER_AUTHORITY_P_KEY	Clé publique de signature de l'autorité de séquestre, cette clé est elle-même signée par la clé racine de l'organisation.
SEQUESTER_AUTHORITY_S_KEY	Clé secrète de signature de l'autorité de séquestre, cette clé sert à signer les services de séquestre.
SEQUESTER_SERVICE_P_KEY	Clé publique de chiffrement d'un service de séquestre. Cette clé est utilisée par les utilisateurs pour chiffrer une copie des nouvelles données et métadonnées des workspaces.
SEQUESTER_SERVICE_S_KEY	Clé secrète de chiffrement d'un service de séquestre.

9. Références

[1] Diffie, Whitfield, and Martin Hellman. "New directions in cryptography." *IEEE transactions on Information Theory* 22.6 (1976): 644-654.

[2] Vaudenay, Serge. "Secure communications over insecure channels based on short authenticated strings." *Annual International Cryptology Conference*. Springer, Berlin, Heidelberg, 2005.

[3] Koh, John S., Steven M. Bellovin, and Jason Nieh. "Why Joanie Can Encrypt: Easy Email Encryption with Easy Key Management." *Proceedings of the Fourteenth EuroSys Conference 2019*. 2019.

[4] Kallahalla, Mahesh, et al. "Plutus: Scalable Secure File Sharing on Untrusted Storage." *Fast*. Vol. 3. 2003.